Document identifier: S32G2RM
Rev. 2 DraftF, Sep 2020

S32G2 Reference Manual

Supports S32G274A, S32G254A, S32G233A, S32G234M

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

COMPANY CONFIDENTIAL

Contents

Chapter 1 About This Manual 91
1.1 Audience 91
1.2 Organization 91
1.2.1 Attachments 91
1.3 Module descriptions 91
1.3.1 Chip-specific information that clarifies content in the same chapter 92
1.3.2 Chip-specific information that refers to a different chapter 93
1.4 Register descriptions 94
1.5 Conventions 95
1.5.1 Notes and cautions 95
1.5.2 Numbering systems 95
1.5.3 Typographic notation 95
1.5.4 Special terms 96
1.6 Editorial changes 97

Chapter 2 Introduction 98
2.1 Overview 98
2.2 Applications 98
2.3 Block diagram 98
2.4 Module functional categories 99
2.5 Feature comparison 101

Chapter 3 Memory Map 105
3.1 Introduction 105

Chapter 4 Signal Description 106
4.1 IOMUX and signal details 106

Chapter 5 Arm Modules 107
5.1 Core complex platform components 107
5.2 Cortex-A53 cluster complex 107
5.2.1 Arm Cortex-A53 MPCore 107
5.2.2 Arm CoreLink Generic Interrupt Controller (GIC-500) 109
5.3 Operation details 112
5.3.1 Operation mode 112
5.3.2 Booting Cortex-A53 cores in 32-bit mode 112
5.3.3 Cluster clock gating 112
5.3.4 Cortex-A53 system counter 113
5.3.5 Cortex-A53 system error interrupt 113
5.3.6 Cortex-A53 wake-up from WFE 113
5.3.7 Cortex-A53 PMUEVENT bus signals observation 113
5.3.7.1 PMUEVENT observer register descriptions 114
5.3.7.1.1 PMUEVENTOBSERVER memory map 114
5.3.7.1.2 PMUEVENT Observer Control Register (CTL0 - CTL29) 114
5.3.7.1.3 PMUEVENT Observer Status Register (STAT0 - STAT29) 116
5.3.8 Cortex-A53/GIC500 lockstep operation 116
5.3.9 Cortex-M7 lockstep operation 119

5.4 Cortex-M7 overview....120
5.4.1 Cortex-M7 core introduction....120
5.4.1.1 Buses, interconnects, and interfaces....120
5.4.2 Debug facilities....121

Chapter 6 NoC Subsystem....122
6.1 Introduction....122
6.2 Functional description....122
6.2.1 Observability....122
6.2.1.1 Concepts....122
6.2.1.1.1 Packet and transaction probes....122
6.2.1.1.2 Observers....123
6.2.1.2 Operation....123
6.2.1.2.1 Logging errors....123
6.2.1.2.2 Tracing packets....124
6.2.1.2.3 Collecting statistics....127
6.2.1.2.4 Transaction profiling operation....128
6.2.1.2.5 Activating observer debug output....129
6.2.1.3 Technical reference....129
6.2.1.3.1 Observation probes....129
6.2.1.3.2 Observers....130
6.2.2 QoS....138
6.2.2.1 Concepts....138
6.2.2.1.1 QoS generators....138
6.2.2.2 Programming....139
6.2.2.2.1 Programming QoS generators....139
6.2.3 Resilience features....142
6.2.3.1 Fault aggregation and reporting....142
6.2.3.1.1 Unit fault aggregation and reporting....142
6.2.3.1.2 Fault controller....143

Chapter 7 Ncore....147
7.1 Chip-specific Ncore information....147
7.1.1 Ncore interface units....147
7.2 Introduction....147
7.3 Features....147
7.4 Block diagram....148
7.5 Signal description....148
7.6 Functional description....148
7.6.1 Caches and cache coherency....148
7.6.2 Ncore agents....148
7.6.3 Ncore interfaces....149
7.6.3.1 Coherent agent interface (CAI)....149
7.6.3.2 Coherent memory interface (CMI)....150
7.6.3.3 Non coherent bridge (NCB)....150
7.6.3.4 Directory....150
7.6.3.4.1 Snoop filter....150
7.6.4 Transport interconnect....151
7.6.5 Storage protection....151
7.6.5.1 Error types....152
7.6.6 Resiliency....152
7.6.6.1 Component logic duplication....153
7.6.6.2 Fault controller....154

7.6.6.3 BIST sequence 154
7.6.7 Coherent read/write transactions 155
7.6.7.1 Coherent read transactions 155
7.6.7.2 Coherent write transactions 155
7.7 Reset and initialization 155
7.7.1 CAI initialization 156
7.7.2 CMI initialization 156
7.7.3 NCB initialization 156
7.7.4 Directory initialization 156
7.7.5 Discovery routine 157
7.8 Maintenance operations 157
7.8.1 Snoop filter maintenance 158
7.9 Power management 159
7.9.1 CAI power management 160
7.9.1.1 Isolated power state 160
7.9.2 NCB power management 161
7.9.3 CMI power management 161
7.9.4 Directory power management 162
7.10 Memory map and register definition 162
7.10.1 Coherent agent interface unit (CAIU) register descriptions 162
7.10.1.1 CAIU memory map 162
7.10.1.2 CAIU Transaction Control (CAIUTC) 163
7.10.1.3 CAIU Transaction Activity (CAIUTA) 163
7.10.1.4 CAIU Correctable Error Control (CAIUCEC) 164
7.10.1.5 CAIU Correctable Error Status (CAIUCES) 165
7.10.1.6 CAIU Correctable Error Location Register 0 (CAIUCELR0) 167
7.10.1.7 CAIU Correctable Error Location Register 1 (CAIUCELR1) 168
7.10.1.8 CAIU Correctable Error Status Alias (CAIUCESA) 168
7.10.1.9 CAIU Uncorrectable Error Control (CAIUUEC) 170
7.10.1.10 CAIU Uncorrectable Error Status (CAIUUES) 171
7.10.1.11 CAIU Uncorrectable Error Location Register 0 (CAIUUELR0) 172
7.10.1.12 CAIU Uncorrectable Error Location Register 1 (CAIUUELR1) 173
7.10.1.13 CAIU Uncorrectable Error Status Alias (CAIUUESA) 174
7.10.1.14 CAIU Identification (CAIUID) 175
7.10.2 Non-coherent bridge unit (NCBU) register descriptions 177
7.10.2.1 NCBU memory map 177
7.10.2.2 NCBU Transaction Control (NCBUTC) 177
7.10.2.3 NCBU Transaction Activity (NCBUTA) 178
7.10.2.4 NCBU Correctable Error Control (NCBUCEC) 179
7.10.2.5 NCBU Correctable Error Status (NCBUCES) 180
7.10.2.6 NCBU Correctable Error Location Register 0 (NCBUCELR0) 182
7.10.2.7 NCBU Correctable Error Location Register 1 (NCBUCELR1) 183
7.10.2.8 NCBU Correctable Error Status Alias (NCBUCESA) 183
7.10.2.9 NCBU Uncorrectable Error Control (NCBUUEC) 185
7.10.2.10 NCBU Uncorrectable Error Status (NCBUUES) 186
7.10.2.11 NCBU Uncorrectable Error Location Register 0 (NCBUUELR0) 187
7.10.2.12 NCBU Uncorrectable Error Location Register 1 (NCBUUELR1) 188
7.10.2.13 NCBU Uncorrectable Error Status Alias (NCBUUESA) 189
7.10.2.14 NCBU Identification Register (NCBUID) 190
7.10.3 Directory unit (DIRU) register descriptions 192
7.10.3.1 DIRU memory map 192
7.10.3.2 DIRU Transaction Activity (DIRUTA) 192
7.10.3.3 DIRU Snoop Filtering Enable (DIRUSFE) 193
7.10.3.4 DIRU Caching Agent Snoop Enable (DIRUCASE0) 194
7.10.3.5 DIRU Caching Agent Snoop Activity (DIRUCASA0) 195

7.10.3.6 DIRU Snoop Filter Maintenance Control (DIRUSFMC)..... 195
7.10.3.7 DIRU Snoop Filter Maintenance Activity (DIRUSFMA)..... 197
7.10.3.8 DIRU Snoop Filter Maintenance Location Register 0 (DIRUSFMLR0)..... 197
7.10.3.9 DIRU Snoop Filter Maintenance Location Register 1 (DIRUSFMLR1)..... 198
7.10.3.10 DIRU Correctable Error Control (DIRUCEC)..... 199
7.10.3.11 DIRU Correctable Error Status (DIRUCES)..... 200
7.10.3.12 DIRU Correctable Error Location Register 0 (DIRUCELR0)..... 202
7.10.3.13 DIRU Correctable Error Location Register 1 (DIRUCELR1)..... 203
7.10.3.14 DIRU Correctable Error Status Alias (DIRUCESA)..... 204
7.10.3.15 DIRU Uncorrectable Error Control (DIRUUEC)..... 205
7.10.3.16 DIRU Uncorrectable Error Status (DIRUUES)..... 206
7.10.3.17 DIRU Uncorrectable Error Location Register 0 (DIRUUELR0)..... 208
7.10.3.18 DIRU Uncorrectable Error Location Register 1 (DIRUUELR1)..... 209
7.10.3.19 DIRU Uncorrectable Error Status Alias (DIRUUESA)..... 210
7.10.3.20 DIRU Identification Register (DIRUID)..... 212
7.10.4 Coherent memory interface unit (CMIU) register descriptions..... 212
7.10.4.1 CMIU memory map..... 212
7.10.4.2 CMIU Transaction Activity (CMIUTA)..... 213
7.10.4.3 CMIU Correctable Error Control (CMIUCEC)..... 214
7.10.4.4 CMIU Correctable Error Status (CMIUCES)..... 215
7.10.4.5 CMIU Correctable Error Location Register 0 (CMIUCELR0)..... 216
7.10.4.6 CMIU Correctable Error Location Register 1 (CMIUCELR1)..... 217
7.10.4.7 CMIU Correctable Error Status Alias (CMIUCESA)..... 218
7.10.4.8 CMIU Uncorrectable Error Control (CMIUUEC)..... 219
7.10.4.9 CMIU Uncorrectable Error Status (CMIUUES)..... 220
7.10.4.10 CMIU Uncorrectable Error Location Register 0 (CMIUUELR0)..... 222
7.10.4.11 CMIU Uncorrectable Error Location Register 1 (CMIUUELR1)..... 223
7.10.4.12 CMIU Uncorrectable Error Status Alias (CMIUUESA)..... 223
7.10.4.13 CMIU Identification Register (CMIUID)..... 225
7.10.5 Coherent subsystem (CSR) register descriptions..... 226
7.10.5.1 CSR memory map..... 226
7.10.5.2 Coherent Subsystem ACE DVM Snoop Enable (CSADSE0)..... 227
7.10.5.3 Coherent Subsystem ACE DVM Snoop Activity (CSADSA)..... 227
7.10.5.4 Coherent Subsystem Correctable Error Interrupt Status (CSCEIS0 - CSCEIS3)..... 228
7.10.5.5 Coherent Subsystem Correctable Error Interrupt Status (CSCEIS4 - CSCEIS6)..... 229
7.10.5.6 Coherent Subsystem Uncorrectable Error Interrupt Status (CSUEIS0 - CSUEIS3)..... 230
7.10.5.7 Coherent Subsystem Uncorrectable Error Interrupt Status (CSUEIS4 - CSUEIS6)..... 231
7.10.5.8 Coherent Subsystem Snoop Filter Identification (CSSFIDR0)..... 232
7.10.5.9 Coherent Subsystem Snoop Filter Identification (CSSFIDR1 - CSSFIDR31)..... 233
7.10.5.10 Coherent Subsystem Unit Identification (CSUID)..... 234
7.10.5.11 Coherent Subsystem Identification (CSID)..... 236
7.10.6 CCTI fault controller register descriptions..... 236
7.10.6.1 CCTI_fault_ctrl memory map..... 237
7.10.6.2 Core ID (COREID)..... 237
7.10.6.3 Revision ID (REVISIONID)..... 238
7.10.6.4 Mission Fault 0 (MISSION_FAULT0)..... 239
7.10.6.5 Mission Fault 1 (MISSION_FAULT1)..... 240
7.10.6.6 Mission Fault 2 (MISSION_FAULT2)..... 241
7.10.6.7 Mission Fault 3 (MISSION_FAULT3)..... 241
7.10.6.8 Latent Fault 0 (LATENT_FAULT0)..... 242
7.10.6.9 Latent Fault 1 (LATENT_FAULT1)..... 243
7.10.6.10 Latent Fault 2 (LATENT_FAULT2)..... 244
7.10.6.11 Latent Fault 3 (LATENT_FAULT3)..... 245
7.10.6.12 Faults (FAULTS)..... 245
7.10.6.13 Interrupt Enable (INTEN)..... 246

7.10.6.14 Interrupt Clear (INTCLR)....247
7.10.6.15 BIST Control (BISTCTL)....248
7.10.6.16 BIST Done (BIST_DONE)....249
7.10.6.17 BIST Timeout 1 (BIST_TO1)....250
7.10.6.18 BIST Timeout 2 (BIST_TO2)....251
7.10.7 Functional safety controller (FSC) register descriptions....251
7.10.7.1 FSC memory map....251
7.10.7.2 FSC BIST Control (SCBISTC)....252
7.10.7.3 FSC BIST Activity (SCBISTA)....252
7.10.7.4 FSC Correctable Errors Threshold (SCCETH)....254
7.10.7.5 FSC Latent Fault (SCLF0)....254
7.10.7.6 FSC Mission Fault (SCMF0)....255
7.10.7.7 FSC Correctable Error Threshold Fault (SCCETHF0)....256

Chapter 8 Miscellaneous System Control Module (MSCM)....258
8.1 Introduction....258
8.1.1 Features....258
8.2 Memory map and register definition....258
8.2.1 Core configuration registers....258
8.2.2 Shared peripheral interrupt (SPI) routing....259
8.2.3 MSCM register descriptions....259
8.2.3.1 MSCM memory map....259
8.2.3.2 Processor X Type Register (CPXTYPE)....264
8.2.3.3 Processor X Number Register (CPXNUM)....265
8.2.3.4 Processor X Revision Register (CPXREV)....266
8.2.3.5 Processor X Configuration 0 Register (CPXCFG0)....267
8.2.3.6 Processor X Configuration 1 Register (CPXCFG1)....269
8.2.3.7 Processor X Configuration 2 Register (CPXCFG2)....270
8.2.3.8 Processor X Configuration 3 Register (CPXCFG3)....272
8.2.3.9 Processor Type Register (CP0TYPE - CP6TYPE)....274
8.2.3.10 Processor Number Register (CP0NUM - CP6NUM)....276
8.2.3.11 Processor Count Register (CP0REV - CP6REV)....277
8.2.3.12 Processor Configuration 0 Register (CP0CFG0 - CP6CFG0)....279
8.2.3.13 Processor Configuration 1 Register (CP0CFG1 - CP6CFG1)....281
8.2.3.14 Processor Configuration 2 Register (CP0CFG2 - CP6CFG2)....283
8.2.3.15 Processor Configuration 3 Register (CP0CFG3 - CP6CFG3)....285
8.2.3.16 Interrupt Router CPa Interruptb Status Register (IRCP0ISR0 - IRCP6ISR2)....288
8.2.3.17 Interrupt Router CPa Interruptb Generation Register (IRCP0IGR0 - IRCP6IGR2)....290
8.2.3.18 Interrupt Router CPa Interruptb Status Register (IRCP0ISR3 - IRCP6ISR4)....291
8.2.3.19 Interrupt Router CPn Interruptx Generation Register (IRCP0IGR3 - IRCP6IGR3)....294
8.2.3.20 Interrupt Router CPn Interruptx Generation Register (IRCP0IGR4 - IRCP6IGR4)....296
8.2.3.21 Interrupt Router Configuration Register (IRCPCFG)....298
8.2.3.22 Interrupt Router Non-Maskable Interrupt Control Register (IRNMIC)....300
8.2.3.23 Interrupt Router Shared Peripheral Routing Control Register (IRSPRC0 - IRSPRC239)....301
8.3 Functional description....302
8.3.1 MSI routing....303
8.3.1.1 Core-to-core MSIs....304
8.3.1.2 PCIe-to-core MSIs....304
8.3.2 Interrupt steering and semaphores....304
8.3.2.1 Interrupt handling overview....304
8.3.2.2 MSCM interrupt router functional description....305

Chapter 9 Miscellaneous Control Module (MCM)....306

9.1 Introduction....306
9.1.1 Features....306
9.2 Memory map/register descriptions....306
9.2.1 MCM register descriptions....306
9.2.1.1 MCM memory map....306
9.2.1.2 SoC-defined Platform Revision (PLREV)....307
9.2.1.3 Processor Core Type (PCT)....307
9.2.1.4 Core Platform Control (CPCR)....308
9.2.1.5 Interrupt Status and Control (ISCR)....309
9.2.1.6 Local Memory Descriptor 0 (LMEM_DESC_0)....312
9.2.1.7 Local Memory Descriptor a (LMEM_DESC_1 - LMEM_DESC_2)....315
9.2.1.8 Local Memory Descriptor 3 (LMEM_DESC_3)....317
9.2.1.9 Local Memory Descriptor 4 (LMEM_DESC_4)....320
9.3 Functional description....322
9.3.1 Interrupts....322
9.4 Glossary....322

Chapter 10 A53 Cluster General Purpose Registers (A53_GPR)....323
10.1 Introduction....323
10.2 Technical reference....323
10.3 A53_CLUSTER_GPR register descriptions....323
10.3.1 A53_CLUSTER_GPR memory map....323
10.3.2 GPR00 (GPR00)....324
10.3.3 GPR01 (GPR01)....326
10.3.4 GPR06 (GPR06)....328
10.3.5 GPR07 (GPR07)....329
10.3.6 GPR08 (GPR08)....330
10.3.7 GPR09 (GPR09)....333
10.3.8 Cluster0 Core0 CPUMERRSR High (GPR10)....334
10.3.9 Cluster0 Core0 CPUMERRSR Low (GPR11)....335
10.3.10 Cluster0 Core1 CPUMERRSR High (GPR12)....336
10.3.11 Cluster0 Core1 CPUMERRSR Low (GPR13)....336
10.3.12 Cluster1 Core0 CPUMERRSR High (GPR14)....337
10.3.13 Cluster1 Core0 CPUMERRSR Low (GPR15)....338
10.3.14 Cluster1 Core1 CPUMERRSR High (GPR16)....339
10.3.15 Cluster1 Core1 CPUMERRSR Low (GPR17)....339
10.3.16 Cluster0 L2MERRSR High (GPR18)....340
10.3.17 Cluster0 L2MERRSR Low (GPR19)....341
10.3.18 Cluster1 L2MERRSR High (GPR20)....342
10.3.19 GPR21 (GPR21)....342

Chapter 11 Cortex-M7 Cluster General Purpose Registers (CM7_GPR)....344
11.1 Introduction....344
11.2 CM7_CLUSTER_GPR register descriptions....344
11.2.1 CM7_CLUSTER_GPR memory map....344
11.2.2 Cortex-M7 GPR0 (CORTEXM7_GPR0)....344

Chapter 12 Enhanced Direct Memory Access (eDMA)....346
12.1 Chip-specific eDMA information....346
12.1.1 eDMA instances....346
12.1.2 ECC support....346
12.1.3 eDMA bandwidth management....346

12.2 Introduction....346
12.2.1 eDMA system block diagram....347
12.2.2 Block parts....347
12.2.3 Features....348
12.3 Modes of operation....349
12.4 Memory map/register definition....349
12.4.1 DMA MP register descriptions....350
12.4.1.1 MP memory map....350
12.4.1.2 Management Page Control (CSR)....351
12.4.1.3 Management Page Error Status (ES)....354
12.4.1.4 Management Page Interrupt Request Status (INT)....356
12.4.1.5 Management Page Hardware Request Status (HRS)....357
12.4.1.6 Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI)....358
12.4.2 DMA TCD register descriptions....359
12.4.2.1 TCD memory map....359
12.4.2.2 Channel Control and Status (CH0_CSR - CH31_CSR)....360
12.4.2.3 Channel Error Status (CH0_ES - CH31_ES)....362
12.4.2.4 Channel Interrupt Status (CH0_INT - CH31_INT)....364
12.4.2.5 Channel System Bus (CH0_SBR - CH31_SBR)....365
12.4.2.6 Channel Priority (CH0_PRI - CH31_PRI)....367
12.4.2.7 TCD Source Address (TCD0_SADDR - TCD31_SADDR)....368
12.4.2.8 TCD Signed Source Address Offset (TCD0_SOFF - TCD31_SOFF)....369
12.4.2.9 TCD Transfer Attributes (TCD0_ATTR - TCD31_ATTR)....370
12.4.2.10 TCD Transfer Size Without Minor Loop Offsets (TCD0_NBYTES_MLOFFNO - TCD31_NBYTES_MLOFFNO)....371
12.4.2.11 TCD Transfer Size with Minor Loop Offsets (TCD0_NBYTES_MLOFFYES - TCD31_NBYTES_MLOFFYES)....372
12.4.2.12 TCD Last Source Address Adjustment / Store DADDR Address (TCD0_SLAST_SDA - TCD31_SLAST_SDA)....374
12.4.2.13 TCD Destination Address (TCD0_DADDR - TCD31_DADDR)....375
12.4.2.14 TCD Signed Destination Address Offset (TCD0_DOFF - TCD31_DOFF)....376
12.4.2.15 TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD31_CITER_ELINKNO)....377
12.4.2.16 TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD31_CITER_ELINKYES)....378
12.4.2.17 TCD Last Destination Address Adjustment / Scatter Gather Address (TCD0_DLAST_SGA - TCD31_DLAST_SGA)....379
12.4.2.18 TCD Control and Status (TCD0_CSR - TCD31_CSR)....380
12.4.2.19 TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD31_BITER_ELINKNO)....383
12.4.2.20 TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD31_BITER_ELINKYES)....384
12.5 Functional description....385
12.5.1 eDMA basic data flow....386
12.5.2 Fault reporting and handling....388
12.5.3 Channel preemption....389
12.6 Initialization/application information....389
12.6.1 eDMA initialization....389
12.6.2 eDMA arbitration....391
12.6.3 Programming errors....393
12.6.4 Arbitration mode considerations....393
12.6.4.1 Fixed group arbitration, fixed channel arbitration....393
12.6.4.2 Fixed group arbitration, round-robin channel arbitration....393
12.6.5 Performing DMA transfers....393
12.6.5.1 Single request....393

12.6.5.2 Multiple requests....394
12.6.5.3 Using the modulo feature....395
12.6.6 Monitoring transfer descriptor status....396
12.6.6.1 Testing for minor loop completion....396
12.6.6.2 Reading the transfer descriptors of active channels....397
12.6.6.3 Checking channel preemption status....397
12.6.7 Channel linking....397
12.6.8 Dynamic programming....398
12.6.8.1 Dynamically changing the channel priority....398
12.6.8.2 Dynamic channel linking....398
12.6.8.3 Dynamic scatter/gather....399
12.6.8.3.1 Method 1 (channel not using major loop channel linking)....399
12.6.8.3.2 Method 2 (channel using major loop channel linking)....400
12.6.9 Lockstep....400
12.6.10 Suspend/resume a DMA channel with active hardware service requests....401
12.6.10.1 Suspend an active DMA channel....401
12.6.10.2 Resume a DMA channel....402
12.7 Glossary....402

Chapter 13 DMA CRC Module (DMA_CRC)....403
13.1 Introduction....403
13.1.1 Block diagram....403
13.1.2 Features....403
13.1.3 Mode of operation....404
13.2 DMA_CRC register descriptions....404
13.2.1 DMA_CRC memory map....404
13.2.2 Global Enable CRC Register (GEC)....405
13.2.3 CRC Control Register (CTL0 - CTL7)....406
13.2.4 Initial CRC Value Register (ICRC0 - ICRC7)....409
13.2.5 Final CRC Value Register (FCRC0 - FCRC7)....410
13.3 Functional description....410

Chapter 14 Direct Memory Access Multiplexer (DMAMUX)....412
14.1 Chip-specific DMAMUX information....412
14.1.1 DMAMUX-to-eDMA channel mapping....412
14.2 Introduction....412
14.2.1 Overview....412
14.2.2 Features....413
14.2.3 Modes of operation....413
14.3 Memory map/register definition....413
14.3.1 DMAMUX register descriptions....413
14.3.1.1 DMAMUX memory map....413
14.3.1.2 Channel Configuration register (CHCFG0 - CHCFG15)....414
14.4 Functional description....415
14.4.1 DMA channels with periodic triggering capability....416
14.4.2 DMA channels with no triggering capability....417
14.4.3 Always-enabled DMA sources....417
14.5 Initialization/application information....418
14.5.1 Reset....418
14.5.2 Enabling and configuring sources....418

Chapter 15 Extended Resource Domain Controller (XRDC)....422

15.1 Chip-specific XRDC_0 information.....422
15.1.1 XRDC instances.....422
15.1.2 XRDC_0 submodule configurations.....422
15.1.2.1 MDAC configurations.....422
15.1.2.2 MRC configurations.....423
15.1.2.3 PAC configurations.....424
15.1.3 XRDC_0 MRCs and SRAM interleaving.....425
15.1.4 XRDC_1 Introduction.....425
15.2 Introduction.....426
15.2.1 Features.....426
15.2.2 Block diagram.....427
15.2.3 Modes of operation.....428
15.3 External signal description.....428
15.4 Memory map register definition.....428
15.4.1 XRDC_0 register descriptions.....429
15.4.1.1 XRDC_0 memory map.....429
15.4.1.2 Control (CR).....468
15.4.1.3 Hardware Configuration 0 (HWCFG0).....469
15.4.1.4 Hardware Configuration 1 (HWCFG1).....470
15.4.1.5 Hardware Configuration 2 (HWCFG2).....471
15.4.1.6 Master Domain Assignment Configuration (MDACFG0 - MDACFG21).....472
15.4.1.7 Memory Region Configuration (MRCFG0 - MRCFG13).....474
15.4.1.8 Domain Error Location (DERRLOC0 - DERRLOC7).....475
15.4.1.9 Domain Error Word0 (DERR_W0_0 - DERR_W0_19).....477
15.4.1.10 Domain Error Word1 (DERR_W1_0 - DERR_W1_19).....478
15.4.1.11 Domain Error Word2 (DERR_W2_0 - DERR_W2_19).....481
15.4.1.12 Domain Error Word3 (DERR_W3_0 - DERR_W3_19).....482
15.4.1.13 Process Identifier (PID0 - PID18).....484
15.4.1.14 Master Domain Assignment (MDA_W0_0_DFMT0 - MDA_W7_18_DFMT0).....487
15.4.1.15 Master Domain Assignment (MDA_W0_2_DFMT1 - MDA_W0_21_DFMT1).....492
15.4.1.16 Peripheral Domain Access Control (PDAC_W0_0 - PDAC_W0_408).....494
15.4.1.17 Peripheral Domain Access Control (PDAC_W1_0 - PDAC_W1_408).....496
15.4.1.18 Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_223).....497
15.4.1.19 Memory Region Descriptor (MRGD_W1_0 - MRGD_W1_223).....498
15.4.1.20 Memory Region Descriptor (MRGD_W2_0 - MRGD_W2_223).....499
15.4.1.21 Memory Region Descriptor (MRGD_W3_0 - MRGD_W3_223).....501
15.4.2 XRDC_1 register descriptions.....502
15.4.2.1 XRDC_1 memory map.....502
15.4.2.2 Control (CR).....509
15.4.2.3 Hardware Configuration 0 (HWCFG0).....511
15.4.2.4 Hardware Configuration 1 (HWCFG1).....512
15.4.2.5 Hardware Configuration 2 (HWCFG2).....513
15.4.2.6 Master Domain Assignment Configuration (MDACFG0 - MDACFG7).....514
15.4.2.7 Memory Region Configuration (MRCFG0 - MRCFG5).....515
15.4.2.8 Domain Error Location (DERRLOC0 - DERRLOC7).....516
15.4.2.9 Domain Error Word0 (DERR_W0_0 - DERR_W0_16).....518
15.4.2.10 Domain Error Word1 (DERR_W1_0 - DERR_W1_16).....519
15.4.2.11 Domain Error Word2 (DERR_W2_0 - DERR_W2_16).....521
15.4.2.12 Domain Error Word3 (DERR_W3_0 - DERR_W3_16).....522
15.4.2.13 Master Domain Assignment (MDA_W0_0_DFMT1 - MDA_W0_7_DFMT1).....524
15.4.2.14 Peripheral Domain Access Control (PDAC_W0_0 - PDAC_W0_15).....526
15.4.2.15 Peripheral Domain Access Control (PDAC_W1_0 - PDAC_W1_15).....528
15.4.2.16 Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_83).....529
15.4.2.17 Memory Region Descriptor (MRGD_W1_0 - MRGD_W1_83).....530
15.4.2.18 Memory Region Descriptor (MRGD_W2_0 - MRGD_W2_83).....531

15.4.2.19 Memory Region Descriptor (MRGD_W3_0 - MRGD_W3_83)....533
15.5 Functional description....534
15.5.1 Manager (MGR)....534
15.5.2 Master domain assignment controller (MDAC)....534
15.5.3 D*x*ACP evaluation....535
15.5.4 Hardware semaphores and dynamic access rights....536
15.5.5 Memory region controller (MRC)....536
15.5.5.1 Memory region descriptor hit determination....537
15.5.5.2 Memory region D*x*ACP evaluation....537
15.5.5.3 MRC C code description....538
15.5.6 Peripheral access controller (PAC)....540
15.6 Initialization information....541
15.7 Application information....542
15.7.1 Master domain assignments....542
15.7.2 Cache coherency interconnect (CCI)....542
15.7.3 Memory region descriptor management....544
15.7.4 Domain error capture management....544
15.8 Glossary....545

Chapter 16 System Integration Unit Lite2 (SIUL2)....546
16.1 Chip-specific SIUL2 information....546
16.1.1 SIUL2 memory map....546
16.1.2 Register configuration....546
16.1.3 Interrupts and DMA requests....546
16.1.4 MIDR1 and MIDR2 Reset Values....546
16.2 Introduction....547
16.2.1 Overview....547
16.2.2 Features....548
16.3 SIUL2_0 register descriptions....548
16.3.1 SIUL2_0 memory map....549
16.3.2 SIUL2 MCU ID Register #1 (MIDR1)....559
16.3.3 SIUL2 MCU ID Register #2 (MIDR2)....560
16.3.4 SIUL2 DMA/Interrupt Status Flag Register0 (DISR0)....563
16.3.5 SIUL2 DMA/Interrupt Request Enable Register0 (DIRER0)....567
16.3.6 SIUL2 DMA/Interrupt Request Select Register0 (DIRSR0)....571
16.3.7 SIUL2 Interrupt Rising-Edge Event Enable Register 0 (IREER0)....575
16.3.8 SIUL2 Interrupt Falling-Edge Event Enable Register 0 (IFEER0)....578
16.3.9 SIUL2 Interrupt Filter Enable Register 0 (IFER0)....581
16.3.10 SIUL2 Interrupt Filter Maximum Counter Register (IFMCR0 - IFMCR11)....585
16.3.11 SIUL2 Interrupt Filter Clock Prescaler Register (IFCPR)....586
16.3.12 SIUL2 Multiplexed Signal Configuration Register (MSCR0 - MSCR101)....586
16.3.13 SIUL2 Input Multiplexed Signal Configuration Register (IMCR0 - IMCR83)....590
16.3.14 SIUL2 GPIO Pad Data Output Register (GPDO0 - GPDO101)....594
16.3.15 SIUL2 GPIO Pad Data Input Register (GPDI0 - GPDI101)....598
16.3.16 SIUL2 Parallel GPIO Pad Data Out Register (PGPDO0 - PGPDO5)....602
16.3.17 SIUL2 Parallel GPIO Pad Data Out Register (PGPDO6)....604
16.3.18 SIUL2 Parallel GPIO Pad Data In Register (PGPDI0 - PGPDI5)....606
16.3.19 SIUL2 Parallel GPIO Pad Data In Register (PGPDI6)....608
16.3.20 SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO0 - MPGPDO5)....610
16.3.21 SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO6)....615
16.4 SIUL2_1 register descriptions....618
16.4.1 SIUL2_1 memory map....619
16.4.2 SIUL2 MCU ID Register #1 (MIDR1)....633
16.4.3 SIUL2 MCU ID Register #2 (MIDR2)....635

16.4.4 SIUL2 DMA/Interrupt Status Flag Register0 (DISR0)..... 637
16.4.5 SIUL2 DMA/Interrupt Request Enable Register0 (DIRER0)..... 643
16.4.6 SIUL2 DMA/Interrupt Request Select Register0 (DIRSR0)..... 648
16.4.7 SIUL2 Interrupt Rising-Edge Event Enable Register 0 (IREER0)..... 654
16.4.8 SIUL2 Interrupt Falling-Edge Event Enable Register 0 (IFEER0)..... 658
16.4.9 SIUL2 Interrupt Filter Enable Register 0 (IFER0)..... 661
16.4.10 SIUL2 Interrupt Filter Maximum Counter Register (IFMCR0 - IFMCR31)..... 665
16.4.11 SIUL2 Interrupt Filter Clock Prescaler Register (IFCPR)..... 666
16.4.12 SIUL2 Multiplexed Signal Configuration Register (MSCR112 - MSCR190)..... 667
16.4.13 SIUL2 Input Multiplexed Signal Configuration Register (IMCR119 - IMCR495)..... 672
16.4.14 SIUL2 GPIO Pad Data Output Register (GPDO115 - GPDO188)..... 679
16.4.15 SIUL2 GPIO Pad Data Input Register (GPDI115 - GPDI188)..... 682
16.4.16 SIUL2 Parallel GPIO Pad Data Out Register (PGPDO7)..... 684
16.4.17 SIUL2 Parallel GPIO Pad Data Out Register (PGPDO9)..... 686
16.4.18 SIUL2 Parallel GPIO Pad Data Out Register (PGPDO11)..... 688
16.4.19 SIUL2 Parallel GPIO Pad Data Out Register (PGPDO10)..... 690
16.4.20 SIUL2 Parallel GPIO Pad Data In Register (PGPDI7)..... 692
16.4.21 SIUL2 Parallel GPIO Pad Data In Register (PGPDI9)..... 694
16.4.22 SIUL2 Parallel GPIO Pad Data In Register (PGPDI11)..... 696
16.4.23 SIUL2 Parallel GPIO Pad Data In Register (PGPDI10)..... 698
16.4.24 SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO7)..... 700
16.4.25 SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO9 - MPGPDO10)..... 704
16.4.26 SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO11)..... 708
16.5 Functional description..... 712
16.5.1 Pad Control..... 712
16.5.2 General Purpose Input and Output pads (GPIO)..... 712
16.5.3 External interrupts/DMA requests (REQ Pins)..... 713
16.5.3.1 External interrupt initialization..... 713
16.5.3.2 External interrupt management..... 714
16.5.3.3 External interrupt request..... 714
16.5.3.4 DMA requests..... 715

Chapter 17 Semaphores2 (SEMA42)..... 716
17.1 Introduction..... 716
17.1.1 Multi-core programming: software gates..... 716
17.1.2 Features..... 716
17.2 Memory map/register definition..... 717
17.2.1 SEMA42 register descriptions..... 717
17.2.1.1 SEMA42 memory map..... 717
17.2.1.2 Gate Register (GATE0 - GATE15)..... 718
17.2.1.3 Reset Gate Read (RSTGT_R)..... 719
17.2.1.4 Reset Gate Write (RSTGT_W)..... 721
17.3 Functional description..... 722
17.4 Glossary..... 723

Chapter 18 System Resource Controller (SRC)..... 724
18.1 Introduction..... 724
18.2 SRC register descriptions..... 724
18.2.1 SRC memory map..... 724
18.2.2 Software-Triggered Faults (SW_NCF)..... 724
18.2.3 GMAC Control (GMAC_0_CTRL_STS)..... 726
18.2.4 HSE Cortex-M7 Status (HSE_CM7_STATUS_REG)..... 727
18.2.5 CMU Status 1 (CMU_STATUS_REG1)..... 728

18.2.6 CMUs Status 2 (CMU_STATUS_REG2)....732
18.2.7 FCCU EOUT Override Clear (FCCU_EOUT_OVERRIDE_CLEAR_REG)....733
18.2.8 GPR21 (GPR21)....734
18.2.9 Debug Control (DEBUG_CONTROL)....736
18.2.10 Timestamp Control (TIMESTAMP_CONTROL_REGISTER)....738
18.2.11 FlexRay OS Tick Input Select (FLEXRAY_OS_TICK_INPUT_SELECT_REG)....739

Chapter 19 Main General Purpose Registers (Main GPRs)....741
19.1 Introduction....741
19.2 Memory map and register definition....741
19.2.1 S32G_GPR register descriptions....741
19.2.1.1 S32G_MAIN_GPR memory map....741
19.2.1.2 PFE Port Coherency Enable (PFE_COH_EN)....741
19.2.1.3 PFE EMAC Interface Mode (PFE_EMACX_INTF_SEL)....743
19.2.1.4 PFE EMACX Power Control (PFE_PWR_CTRL)....744
19.2.1.5 LLCE Subsystem Status (LLCE_STAT)....745
19.2.1.6 LLCE Power Control (LLCE_CTRL)....746
19.2.1.7 DDR Urgent Control (DDR_URGENT_CTRL)....747
19.2.1.8 FlexTimer Global Load Control (FLXTIM_CTRL)....749
19.2.1.9 FlexTimer LDOK Status (FLXTIM_STAT)....749
19.2.1.10 Top CMU_FC Status (CMU_STAT)....750
19.2.1.11 Accelerator NoC NoPendingTrans Status (NOC_NOPEND_TRANS)....752
19.2.1.12 SerDes RD/WD Toggle Control (PCIE_TOGGLE)....753
19.2.1.13 SerDes Toggle Done Status (PCIE_TOGGLEDONE_STAT)....754
19.2.1.14 Generic Control 0 (GENCTRL0)....755
19.2.1.15 Generic Control 1 (GENCTRL1)....756
19.2.1.16 Generic Status 0 (GENSTAT0)....757

Chapter 20 Standby General Purpose Registers (Stby GPRs)....759
20.1 Introduction....759
20.2 S32G_STDBY_GPR register descriptions....759
20.2.1 S32G_STDBY_GPR memory map....759
20.2.2 WKUP Pad Pullup/Pulldown Select (WKUP_PUS)....759

Chapter 21 Register Protection (REG_PROT)....761
21.1 Introduction....761
21.2 Features....761
21.3 Modes of operation....761
21.4 External signal description....761
21.5 Memory map and register definition....761
21.5.1 Memory map....763
21.5.2 Module registers (MR)....763
21.5.3 MR and set soft lock bit (LMR)....763
21.6 REG_PROT register descriptions....763
21.6.1 REG_PROT memory map....764
21.6.2 Soft Lock Bit Register (SLBRn)....764
21.6.3 Global Configuration Register (GCR)....766
21.7 Functional description....767
21.7.1 General....767
21.7.2 Change lock settings....767
21.7.2.1 Change lock settings directly via area #3....767
21.7.2.2 Enable locking via mirror module space (area #2)....769

21.7.2.3 Write protection for locking bits....769
21.7.3 Access errors....770
21.7.4 Sequence....770
21.8 Initialization/application information....770
21.8.1 Reset....770

Chapter 22 Wakeup Unit (WKPU)....771
22.1 Chip-specific WKPU information....771
22.1.1 WKPU configuration....771
22.1.2 Chip-specific NMI Configuration Register (NCR)....771
22.1.3 Interrupts....771
22.1.4 NMI on Standby exit....771
22.1.5 Asynchronous wake-up sources for various modes....772
22.1.6 Pull control of wake-up pads....772
22.1.7 Enabling wakeup pads....772
22.1.8 Wakeup/Interrupt Pull Enable Register (WIPUER_WIPDER)....772
22.2 Introduction....773
22.3 Features....773
22.4 WKPU memory map and registers....774
22.4.1 WKPU register descriptions....774
22.4.1.1 WKPU memory map....774
22.4.1.2 NMI Status Flag Register (NSR)....774
22.4.1.3 NMI Configuration Register (NCR)....776
22.4.1.4 Wakeup Boot Mode Select Register (WBMSR)....778
22.4.1.5 Wakeup/Interrupt Status Flag Register (WISR)....778
22.4.1.6 Interrupt Request Enable Register (IRER)....779
22.4.1.7 Wakeup Request Enable Register (WRER)....780
22.4.1.8 Wakeup/Interrupt Rising-Edge Event Enable Register (WIREER)....781
22.4.1.9 Wakeup/Interrupt Falling-Edge Event Enable Register (WIFEER)....782
22.4.1.10 Wakeup/Interrupt Filter Enable Register (WIFER)....783
22.4.1.11 Wakeup/Interrupt Pull Enable Register (WIPUER_WIPDER)....784
22.5 Functional description....785
22.5.1 Non-maskable interrupts....785
22.5.1.1 NMI management....786
22.6 External wakeups interrupts....787
22.6.1 External interrupt management....787
22.7 On-chip wakeups....788
22.8 Initialization information....788
22.8.1 Glitch filter and pad configuration....788
22.8.2 Non-maskable interrupts....788
22.9 Glossary....788

Chapter 23 Clocking....789
23.1 Introduction....789
23.1.1 Features....789
23.1.2 Overview....789
23.1.2.1 Core-related clock overview....790
23.1.2.2 Peripheral-related clock overview....790
23.1.2.3 DDR PLL clock overview....791
23.1.2.4 Accelerator PLL clock overview....791
23.1.2.5 Aurora clocking overview....791
23.1.2.6 CLKOUT*n* overview....792
23.1.2.6.1 CLKOUT*n* clock configuration....792

23.1.2.7 Other clocks....793
23.1.2.7.1 PFE_MAC_0 clocking overview....793
23.1.2.7.2 PFE_MAC_1 clocking overview....794
23.1.2.7.3 PFE_MAC_2 clocking overview....794
23.1.2.7.4 PFE clocking....795
23.2 Clock sources....795
23.2.1 Introduction....795
23.2.2 Input clock pins....796
23.2.3 Output clock pins....796
23.2.4 FIRC....796
23.2.5 SIRC....797
23.2.6 FXOSC....797
23.2.6.1 FXOSC register interface....797
23.2.7 PLL....797
23.2.7.1 PLL input clocks....798
23.2.7.2 PLL output clocks....798
23.2.7.3 PLL register interface....798
23.2.7.4 DFS....798
23.3 Clock generation....799
23.3.1 Clock source mapping on MC_CGM clock selectors....799
23.3.2 MC_CGM_*n* registers....801
23.3.2.1 MC_CGM_0 registers relationship to output clocks....801
23.3.2.2 MC_CGM_1 register relationship to output clocks....803
23.3.2.3 MC_CGM_2 registers relationship to output clocks....803
23.3.2.4 MC_CGM_5 registers relationship to output clocks....805
23.3.3 Application clock source options....805
23.3.4 System clock frequency limitations....807
23.3.5 Default clock configuration....807
23.3.6 Standby clock configuration....808
23.4 Software resettable domain....808
23.5 Clock on/off procedures....808
23.5.1 Clock-on....809
23.5.1.1 PLL SSCG Setup....810
23.5.2 Clock-off....810
23.5.3 PLL turn on sequence....810
23.5.4 Turning on the CLKOUT*n* signals....811
23.5.5 Turning off the CLKOUT*n* signals....811
23.6 Clock gating....811
23.7 Module clocking....811
23.7.1 Safety modules....812
23.7.1.1 ERM....812
23.7.1.2 FCCU....813
23.7.1.3 Redundancy control and checker unit (RCCU)....813
23.7.1.4 SBSW....813
23.7.1.5 STCU2....813
23.7.2 Communication modules....814
23.7.2.1 SerDes....814
23.7.2.2 Ethernet (GMAC_0)....815
23.7.2.2.1 Ethernet (GMAC_0) RGMII configuration....816
23.7.2.2.2 Ethernet (GMAC_0) RMII configuration....817
23.7.2.2.3 Ethernet (GMAC_0) MII configuration....818
23.7.2.2.4 Ethernet (GMAC_0) SGMII configuration....819
23.7.2.3 Ethernet router (PFE) clocking....820
23.7.2.3.1 PFE_MAC_0 clocking overview....822
23.7.2.3.2 PFE_MAC_1 clocking overview....826

23.7.2.3.3 PFE_MAC_2 clocking overview....830
23.7.2.4 LLCE subsystem clocking overview....834
23.7.2.5 Serial peripheral interface (SPI) clocking....835
23.7.2.6 FlexCAN clocking....836
23.7.2.7 FlexRay communication controller clocking....837
23.7.2.8 I^2C clocking....837
23.7.2.9 LINFlexD clocking....839
23.7.2.10 Universal serial bus (USB) clocking....839
23.7.3 Timer modules....839
23.7.3.1 FTM clocking....840
23.7.3.2 RTC clocking....841
23.7.3.3 SWT clocking....842
23.7.4 Generic modules....842
23.7.4.1 SIUL2 clocking....843
23.7.5 Analog modules....843
23.7.5.1 CTU clocking....844
23.7.5.2 SAR_ADC clocking....844
23.7.6 Debug modules....844
23.7.6.1 Debug trace unit clocking....844
23.7.7 Memory-controller modules....845
23.7.7.1 DDR controller clocking....845
23.7.7.2 SRAM controller clocking....845
23.7.7.3 uSDHC clocking....846
23.7.7.4 QuadSPI clocking....846
23.7.8 Security modules....846
23.7.8.1 Hardware Security Engine (HSE_H)....847
23.7.9 Clocking modules....847
23.7.10 Reset modules....848
23.7.10.1 RDC clocking....848
23.7.11 Power-modes modules....849
23.7.11.1 PMC clocking....849
23.7.12 Core platform modules....849
23.7.12.1 Cortex-A53 cluster clocking....850
23.7.12.2 Cortex-M7 clocking....851
23.7.12.3 eDMA clocking....851
23.7.12.4 Arm generic interrupt controller (GIC500) clocking....851
23.8 Clock monitoring....851
23.8.1 Introduction....851
23.8.2 Clock input sources....852
23.8.3 Enabling and disabling CMU_FC_0 and CMU_FC_5....854

Chapter 24 Clock Generation Module (MC_CGM)....855
24.1 Chip-specific MC_CGM information....855
24.2 Chip-specific MC_CGM_2 information....855
24.2.1 MC_CGM feature configuration....855
24.3 Introduction....855
24.4 Features....856
24.5 MC_CGM_0 register descriptions....856
24.5.1 MC_CGM_0 memory map....857
24.5.2 PCFS Step Duration (PCFS_SDUR)....859
24.5.3 PCFS Divider Change 12 Register (PCFS_DIVC12)....860
24.5.4 PCFS Divider End 12 Register (PCFS_DIVE12)....861
24.5.5 PCFS Divider Start 12 Register (PCFS_DIVS12)....862
24.5.6 Clock Mux 0 Select Control Register (MUX_0_CSC)....862

24.5.7 Clock Mux 0 Select Status Register (MUX_0_CSS)..... 864
24.5.8 Clock Mux 0 Divider 0 Control Register (MUX_0_DC_0)..... 866
24.5.9 Clock Mux 0 Divider 1 Control Register (MUX_0_DC_1)..... 867
24.5.10 Clock Mux 0 Divider Update Status Register (MUX_0_DIV_UPD_STAT).....868
24.5.11 Clock Mux 1 Select Control Register (MUX_1_CSC).....869
24.5.12 Clock Mux 1 Select Status Register (MUX_1_CSS)..... 871
24.5.13 Clock Mux 1 Divider 0 Control Register (MUX_1_DC_0)..... 872
24.5.14 Clock Mux 1 Divider Update Status Register (MUX_1_DIV_UPD_STAT).....873
24.5.15 Clock Mux 2 Select Control Register (MUX_2_CSC).....874
24.5.16 Clock Mux 2 Select Status Register (MUX_2_CSS)..... 875
24.5.17 Clock Mux 2 Divider 0 Control Register (MUX_2_DC_0)..... 877
24.5.18 Clock Mux 2 Divider Update Status Register (MUX_2_DIV_UPD_STAT).....878
24.5.19 Clock Mux 3 Select Control Register (MUX_3_CSC).....879
24.5.20 Clock Mux 3 Select Status Register (MUX_3_CSS)..... 880
24.5.21 Clock Mux 3 Divider 0 Control Register (MUX_3_DC_0)..... 882
24.5.22 Clock Mux 3 Divider Update Status Register (MUX_3_DIV_UPD_STAT).....883
24.5.23 Clock Mux 4 Select Control Register (MUX_4_CSC).....884
24.5.24 Clock Mux 4 Select Status Register (MUX_4_CSS)..... 885
24.5.25 Clock Mux 4 Divider 0 Control Register (MUX_4_DC_0)..... 887
24.5.26 Clock Mux 4 Divider Update Status Register (MUX_4_DIV_UPD_STAT).....888
24.5.27 Clock Mux 5 Select Control Register (MUX_5_CSC).....889
24.5.28 Clock Mux 5 Select Status Register (MUX_5_CSS)..... 891
24.5.29 Clock Mux 5 Divider 0 Control Register (MUX_5_DC_0)..... 893
24.5.30 Clock Mux 5 Divider Update Status Register (MUX_5_DIV_UPD_STAT).....894
24.5.31 Clock Mux 6 Select Control Register (MUX_6_CSC).....895
24.5.32 Clock Mux 6 Select Status Register (MUX_6_CSS)..... 896
24.5.33 Clock Mux 6 Divider 0 Control Register (MUX_6_DC_0)..... 898
24.5.34 Clock Mux 6 Divider Update Status Register (MUX_6_DIV_UPD_STAT).....899
24.5.35 Clock Mux 7 Select Control Register (MUX_7_CSC).....900
24.5.36 Clock Mux 7 Select Status Register (MUX_7_CSS)..... 901
24.5.37 Clock Mux 8 Select Control Register (MUX_8_CSC).....903
24.5.38 Clock Mux 8 Select Status Register (MUX_8_CSS)..... 905
24.5.39 Clock Mux 9 Select Control Register (MUX_9_CSC).....907
24.5.40 Clock Mux 9 Select Status Register (MUX_9_CSS)..... 908
24.5.41 Clock Mux 9 Divider 0 Control Register (MUX_9_DC_0)..... 910
24.5.42 Clock Mux 9 Divider Update Status Register (MUX_9_DIV_UPD_STAT).....911
24.5.43 Clock Mux 10 Select Control Register (MUX_10_CSC).....912
24.5.44 Clock Mux 10 Select Status Register (MUX_10_CSS)..... 913
24.5.45 Clock Mux 10 Divider 0 Control Register (MUX_10_DC_0)..... 915
24.5.46 Clock Mux 10 Divider Update Status Register (MUX_10_DIV_UPD_STAT).....916
24.5.47 Clock Mux 11 Select Control Register (MUX_11_CSC).....917
24.5.48 Clock Mux 11 Select Status Register (MUX_11_CSS)..... 919
24.5.49 Clock Mux 12 Select Control Register (MUX_12_CSC).....921
24.5.50 Clock Mux 12 Select Status Register (MUX_12_CSS)..... 922
24.5.51 Clock Mux 12 Divider 0 Control Register (MUX_12_DC_0)..... 924
24.5.52 Clock Mux 12 Divider Update Status Register (MUX_12_DIV_UPD_STAT).....925
24.5.53 Clock Mux 14 Select Control Register (MUX_14_CSC).....926
24.5.54 Clock Mux 14 Select Status Register (MUX_14_CSS)..... 927
24.5.55 Clock Mux 14 Divider 0 Control Register (MUX_14_DC_0)..... 929
24.5.56 Clock Mux 14 Divider Update Status Register (MUX_14_DIV_UPD_STAT).....930
24.5.57 Clock Mux 15 Select Control Register (MUX_15_CSC).....931
24.5.58 Clock Mux 15 Select Status Register (MUX_15_CSS)..... 932
24.5.59 Clock Mux 15 Divider 0 Control Register (MUX_15_DC_0)..... 934
24.5.60 Clock Mux 15 Divider Update Status Register (MUX_15_DIV_UPD_STAT).....935
24.5.61 Clock Mux 16 Select Control Register (MUX_16_CSC).....936

24.5.62 Clock Mux 16 Select Status Register (MUX_16_CSS)..... 938
24.6 MC_CGM_1 register descriptions.....940
24.6.1 MC_CGM_1 memory map.....940
24.6.2 PCFS Step Duration (PCFS_SDUR).....940
24.6.3 PCFS Divider Change 4 Register (PCFS_DIVC4).....941
24.6.4 PCFS Divider End 4 Register (PCFS_DIVE4)..... 942
24.6.5 PCFS Divider Start 4 Register (PCFS_DIVS4)..... 943
24.6.6 Clock Mux 0 Select Control Register (MUX_0_CSC).....944
24.6.7 Clock Mux 0 Select Status Register (MUX_0_CSS)..... 945
24.7 MC_CGM_2 register descriptions.....948
24.7.1 MC_CGM_2 memory map.....948
24.7.2 PCFS Step Duration (PCFS_SDUR).....950
24.7.3 PCFS Divider Change 33 Register (PCFS_DIVC33).....951
24.7.4 PCFS Divider End 33 Register (PCFS_DIVE33)..... 952
24.7.5 PCFS Divider Start 33 Register (PCFS_DIVS33)..... 953
24.7.6 Clock Mux 0 Select Control Register (MUX_0_CSC).....953
24.7.7 Clock Mux 0 Select Status Register (MUX_0_CSS)..... 955
24.7.8 Clock Mux 0 Divider 0 Control Register (MUX_0_DC_0)..... 957
24.7.9 Clock Mux 0 Divider Trigger Control Register (MUX_0_DIV_TRIG_CTRL)..... 958
24.7.10 Clock Mux 0 Divider Trigger Register (MUX_0_DIV_TRIG)..... 959
24.7.11 Clock Mux 0 Divider Update Status Register (MUX_0_DIV_UPD_STAT).....960
24.7.12 Clock Mux 1 Select Control Register (MUX_1_CSC).....961
24.7.13 Clock Mux 1 Select Status Register (MUX_1_CSS)..... 962
24.7.14 Clock Mux 1 Divider 0 Control Register (MUX_1_DC_0)..... 964
24.7.15 Clock Mux 1 Divider 1 Control Register (MUX_1_DC_1)..... 965
24.7.16 Clock Mux 1 Divider Trigger Control Register (MUX_1_DIV_TRIG_CTRL)..... 966
24.7.17 Clock Mux 1 Divider Trigger Register (MUX_1_DIV_TRIG)..... 967
24.7.18 Clock Mux 1 Divider Update Status Register (MUX_1_DIV_UPD_STAT).....968
24.7.19 Clock Mux 2 Select Control Register (MUX_2_CSC).....969
24.7.20 Clock Mux 2 Select Status Register (MUX_2_CSS)..... 971
24.7.21 Clock Mux 2 Divider 0 Control Register (MUX_2_DC_0)..... 973
24.7.22 Clock Mux 2 Divider 1 Control Register (MUX_2_DC_1)..... 974
24.7.23 Clock Mux 2 Divider Trigger Control Register (MUX_2_DIV_TRIG_CTRL)..... 975
24.7.24 Clock Mux 2 Divider Trigger Register (MUX_2_DIV_TRIG)..... 976
24.7.25 Clock Mux 2 Divider Update Status Register (MUX_2_DIV_UPD_STAT).....977
24.7.26 Clock Mux 3 Select Control Register (MUX_3_CSC).....978
24.7.27 Clock Mux 3 Select Status Register (MUX_3_CSS)..... 979
24.7.28 Clock Mux 3 Divider 0 Control Register (MUX_3_DC_0)..... 981
24.7.29 Clock Mux 3 Divider 1 Control Register (MUX_3_DC_1)..... 982
24.7.30 Clock Mux 3 Divider Trigger Control Register (MUX_3_DIV_TRIG_CTRL)..... 983
24.7.31 Clock Mux 3 Divider Trigger Register (MUX_3_DIV_TRIG)..... 984
24.7.32 Clock Mux 3 Divider Update Status Register (MUX_3_DIV_UPD_STAT).....985
24.7.33 Clock Mux 4 Select Control Register (MUX_4_CSC).....986
24.7.34 Clock Mux 4 Select Status Register (MUX_4_CSS)..... 988
24.7.35 Clock Mux 5 Select Control Register (MUX_5_CSC).....990
24.7.36 Clock Mux 5 Select Status Register (MUX_5_CSS)..... 991
24.7.37 Clock Mux 6 Select Control Register (MUX_6_CSC).....993
24.7.38 Clock Mux 6 Select Status Register (MUX_6_CSS)..... 995
24.7.39 Clock Mux 7 Select Control Register (MUX_7_CSC).....997
24.7.40 Clock Mux 7 Select Status Register (MUX_7_CSS)..... 998
24.7.41 Clock Mux 7 Divider 0 Control Register (MUX_7_DC_0)..... 1000
24.7.42 Clock Mux 7 Divider Trigger Control Register (MUX_7_DIV_TRIG_CTRL)..... 1001
24.7.43 Clock Mux 7 Divider Trigger Register (MUX_7_DIV_TRIG)..... 1002
24.7.44 Clock Mux 7 Divider Update Status Register (MUX_7_DIV_UPD_STAT).....1003
24.7.45 Clock Mux 8 Select Control Register (MUX_8_CSC).....1004

24.7.46 Clock Mux 8 Select Status Register (MUX_8_CSS)..... 1005
24.7.47 Clock Mux 8 Divider 0 Control Register (MUX_8_DC_0)..... 1007
24.7.48 Clock Mux 8 Divider Trigger Control Register (MUX_8_DIV_TRIG_CTRL)..... 1008
24.7.49 Clock Mux 8 Divider Trigger Register (MUX_8_DIV_TRIG)..... 1009
24.7.50 Clock Mux 8 Divider Update Status Register (MUX_8_DIV_UPD_STAT)..... 1010
24.7.51 Clock Mux 9 Select Control Register (MUX_9_CSC)..... 1011
24.7.52 Clock Mux 9 Select Status Register (MUX_9_CSS)..... 1012
24.7.53 Clock Mux 9 Divider 0 Control Register (MUX_9_DC_0)..... 1014
24.7.54 Clock Mux 9 Divider Trigger Control Register (MUX_9_DIV_TRIG_CTRL)..... 1015
24.7.55 Clock Mux 9 Divider Trigger Register (MUX_9_DIV_TRIG)..... 1016
24.7.56 Clock Mux 9 Divider Update Status Register (MUX_9_DIV_UPD_STAT)..... 1017
24.8 MC_CGM_5 register descriptions..... 1018
24.8.1 MC_CGM_5 memory map..... 1018
24.8.2 Clock Mux 0 Select Control Register (MUX_0_CSC)..... 1018
24.8.3 Clock Mux 0 Select Status Register (MUX_0_CSS)..... 1019
24.9 Functional description..... 1021
24.9.1 Clock selection multiplexer..... 1021
24.9.1.1 Hardware-controlled clock multiplexer..... 1021
24.9.1.2 Software-controlled clock multiplexer..... 1023
24.9.2 PCFS..... 1025
24.9.2.1 PCFS control..... 1027
24.9.2.2 Clock source powerup and selection..... 1027
24.9.2.3 Clock source powerdown and deselection..... 1028
24.9.2.4 Clock switch with load change..... 1029
24.9.3 Clock dividers..... 1030
24.9.3.1 50% clock divider..... 1030
24.9.3.2 Clock dividers update..... 1031
24.10 Glossary..... 1031

Chapter 25 Digital Frequency Synthesizer (DFS)..... 1032
25.1 Introduction..... 1032
25.1.1 Block diagram..... 1032
25.1.2 Features..... 1033
25.1.3 Modes of operation..... 1034
25.2 Initialization information..... 1034
25.3 DFS register descriptions..... 1035
25.3.1 DFS memory map..... 1035
25.3.2 Port Status Register (PORTSR)..... 1036
25.3.3 Port Loss of Lock Status (PORTLOLSR)..... 1036
25.3.4 Port Reset (PORTRESET)..... 1037
25.3.5 Control (CTL)..... 1038
25.3.6 Divider for Port n (DVPORT0 - DVPORT5)..... 1039
25.4 Functional description..... 1040
25.4.1 Port divider configuration..... 1040

Chapter 26 PLL Digital Interface (PLLDIG)..... 1041
26.1 Introduction..... 1041
26.1.1 Features..... 1041
26.1.2 Block diagram..... 1041
26.2 Modes of operation..... 1042
26.3 ACCEL_PLL register descriptions..... 1042
26.3.1 ACCEL_PLL memory map..... 1042
26.3.2 PLL Control Register (PLLCR)..... 1042

26.3.3 PLL Status Register (PLLSR)..... 1043
26.3.4 PLL Divider (PLLDV)..... 1044
26.3.5 PLL Frequency Modulation (PLLFM)..... 1046
26.3.6 PLL Fractional Divider (PLLFD)..... 1047
26.3.7 PLL Clock Multiplexer (PLLCLKMUX)..... 1049
26.3.8 PLL Output Divider (PLLODIV_0 - PLLODIV_1)..... 1050
26.4 CORE_PLL register descriptions..... 1051
26.4.1 CORE_PLL memory map..... 1051
26.4.2 PLL Control Register (PLLCR)..... 1051
26.4.3 PLL Status Register (PLLSR)..... 1052
26.4.4 PLL Divider (PLLDV)..... 1053
26.4.5 PLL Frequency Modulation (PLLFM)..... 1055
26.4.6 PLL Fractional Divider (PLLFD)..... 1056
26.4.7 PLL Clock Multiplexer (PLLCLKMUX)..... 1058
26.4.8 PLL Output Divider (PLLODIV_0 - PLLODIV_1)..... 1059
26.5 DDR_PLL register descriptions..... 1060
26.5.1 DDR_PLL memory map..... 1060
26.5.2 PLL Control Register (PLLCR)..... 1060
26.5.3 PLL Status Register (PLLSR)..... 1061
26.5.4 PLL Divider (PLLDV)..... 1062
26.5.5 PLL Frequency Modulation (PLLFM)..... 1064
26.5.6 PLL Fractional Divider (PLLFD)..... 1065
26.5.7 PLL Clock Multiplexer (PLLCLKMUX)..... 1067
26.5.8 PLL Output Divider (PLLODIV_0)..... 1068
26.6 PERIPH_PLL register descriptions..... 1069
26.6.1 PERIPH_PLL memory map..... 1069
26.6.2 PLL Control Register (PLLCR)..... 1069
26.6.3 PLL Status Register (PLLSR)..... 1070
26.6.4 PLL Divider (PLLDV)..... 1071
26.6.5 PLL Fractional Divider (PLLFD)..... 1073
26.6.6 PLL Clock Multiplexer (PLLCLKMUX)..... 1074
26.6.7 PLL Output Divider (PLLODIV_0 - PLLODIV_7)..... 1075
26.7 Functional description..... 1076
26.7.1 Input clock frequency..... 1076
26.7.2 Clock configuration..... 1076
26.7.3 LOL..... 1077
26.7.4 Frequency modulation..... 1078
26.7.4.1 Frequency modulation programming..... 1078
26.8 Initialization information..... 1079

Chapter 27 Fast Crystal Oscillator Digital Controller (FXOSC)..... 1080
27.1 Introduction..... 1080
27.1.1 Overview..... 1080
27.1.2 Features..... 1080
27.2 FXOSC register descriptions..... 1080
27.2.1 FXOSC memory map..... 1080
27.2.2 FXOSC Control Register (CTRL)..... 1080
27.2.3 Oscillator Status Register (STAT)..... 1083
27.3 Functional description..... 1084
27.3.1 Clock generation in Crystal mode..... 1084
27.3.2 Clock generation in Single-Input Bypass mode..... 1084
27.3.3 Clock generation in Differential Input Bypass mode..... 1084
27.4 Initialization information..... 1084

Chapter 28 Reset....1086
28.1 Introduction....1086
28.2 Reset features....1087
28.3 High-level reset sequence....1087
28.4 Reset sources....1088
28.5 Reset sources descriptions....1090
28.5.1 POR sources....1090
28.5.1.1 On-chip POR....1090
28.5.1.2 Critical LVDs....1090
28.5.1.3 External POR_B pin....1090
28.5.1.4 POR watchdog timer....1091
28.5.2 Destructive reset sources....1091
28.5.2.1 SPD fail....1091
28.5.2.2 FCCU failure to react....1092
28.5.2.3 STCU2 unrecoverable fault....1092
28.5.2.4 Functional reset escalation to destructive reset....1092
28.5.2.5 FXOSC failure....1092
28.5.2.6 CORE_PLL loss of lock....1092
28.5.2.7 PERIPH_PLL loss of lock....1092
28.5.2.8 DDR_PLL loss of lock....1092
28.5.2.9 ACCEL_PLL loss of lock....1093
28.5.2.10 XBAR_DIV3 clock failure....1093
28.5.2.11 Life-cycle error....1093
28.5.2.12 Software destructive reset....1093
28.5.2.13 Debug destructive reset....1093
28.5.2.14 HSE_H SWT timeout....1093
28.5.2.15 HSE SNVS tamper detected....1093
28.5.3 Functional reset sources....1094
28.5.3.1 FCCU reset reaction....1094
28.5.3.2 Self-test done....1094
28.5.3.3 SWT0 timeout....1094
28.5.3.4 HSE memory ECC error....1094
28.5.3.5 HSE boot error....1094
28.5.3.6 HSE core lock....1094
28.5.3.7 Software functional reset....1094
28.5.3.8 Debug functional reset....1094
28.5.4 Non-MC_RGM and indirect reset sources....1095
28.5.4.1 JCOMP pin....1095
28.5.5 Destructive reset escalation to stay in reset....1095
28.5.6 RESET_B....1095
28.6 Reset process....1096
28.6.1 Overview....1096
28.6.2 Reset process sequence....1096
28.6.2.1 POR sequence....1096
28.6.2.2 Destructive reset sequence....1096
28.6.2.3 Functional reset sequence....1097
28.6.2.4 Standby reset sequence....1099
28.7 Chip status on reset exit....1100
28.8 Module status during reset process....1103
28.9 Determining reset cause on reset exit....1107
28.10 Reset-related I/Os....1107
28.11 Pad states during reset sequence and immediately after reset exit....1108
28.12 Software resettable domains....1108
28.12.1 Introduction....1108

28.12.2 List of software reset domains/partitions on the chip..... 1109
28.12.2.1 Cortex-A53 software-resettable domain/partition.....1109
28.12.3 State of the software reset domains/partitions after power-up/destructive/functional reset sequence.....1110
28.12.4 Software reset partition turn-on flowchart..... 1111
28.12.5 Software reset partition turn-off flowchart..... 1111
28.12.6 PFE software reset partition turn-off flowchart..... 1112
28.12.7 Core turn-on sequence..... 1113
28.12.8 Core turn-off sequence..... 1114
28.12.9 Self-triggered reset (warm reset) sequence for Cortex-M7 application core.....1115
28.12.10 Cortex-A53 core warm reset sequence.....1116
28.12.11 Software peripheral resets..... 1117
28.13 Reset Domain Controller (RDC) register descriptions.....1117
28.13.1 RDC memory map..... 1117
28.13.2 Software Reset Domain 1 Control (RD1_CTRL_REG)..... 1118
28.13.3 Software Reset Domain 2 Control (RD2_CTRL_REG)..... 1119
28.13.4 Software Reset Domain 3 Control (RD3_CTRL_REG)..... 1120
28.13.5 Software Reset Domain 1 Status (RD1_STAT_REG).....1121
28.13.6 Software Reset Domain 2 Status (RD2_STAT_REG).....1122
28.13.7 Software Reset Domain 3 Status (RD3_STAT_REG).....1123

Chapter 29 Reset Generation Module (MC_RGM)..... 1124
29.1 Chip-specific MC_RGM information..... 1124
29.1.1 Reset sources..... 1124
29.1.1.1 Destructive reset sources..... 1124
29.1.1.2 Functional reset sources.....1124
29.1.2 Peripheral numbers.....1124
29.1.3 Clear functional reset counter during initial application sequence..... 1124
29.1.4 Limitations on individual peripheral resets..... 1124
29.2 Introduction.....1124
29.3 Features..... 1125
29.4 Reset sources.....1125
29.5 External signal description.....1126
29.6 RESET_B pin assertion and pin safe state control.....1126
29.7 MC_RGM register descriptions.....1126
29.7.1 MC RGM Register Map memory map.....1126
29.7.2 Destructive Event Status Register (DES).....1127
29.7.3 Functional /External Reset Status Register (FES)..... 1131
29.7.4 Functional Event Reset Disable Register (FERD)..... 1134
29.7.5 Functional Reset Escalation Counter Register (FREC)..... 1137
29.7.6 Functional Reset Escalation Threshold Register (FRET)..... 1138
29.7.7 Destructive Reset Escalation Threshold Register (DRET).....1139
29.7.8 External Reset Control Register (ERCTRL).....1140
29.7.9 Reset During Standby Status Register (RDSS)..... 1141
29.7.10 Peripheral Reset (PRST0_0).....1142
29.7.11 Peripheral Reset (PRST1_0).....1145
29.7.12 Peripheral Reset (PRST2_0).....1148
29.7.13 Peripheral Reset (PRST3_0).....1151
29.7.14 Peripheral Reset (PRST4_0).....1154
29.7.15 Peripheral Reset (PRST5_0).....1157
29.7.16 Peripheral Reset (PRST6_0).....1160
29.7.17 Peripheral Reset (PRST7_0).....1163
29.7.18 Peripheral Reset Status Register (PSTAT0_0)..... 1166
29.7.19 Peripheral Reset Status Register (PSTAT1_0)..... 1170

29.7.20 Peripheral Reset Status Register (PSTAT2_0)....1173
29.7.21 Peripheral Reset Status Register (PSTAT3_0)....1176
29.7.22 Peripheral Reset Status Register (PSTAT4_0)....1179
29.7.23 Peripheral Reset Status Register (PSTAT5_0)....1182
29.7.24 Peripheral Reset Status Register (PSTAT6_0)....1185
29.7.25 Peripheral Reset Status Register (PSTAT7_0)....1188
29.8 Functional description....1191
29.8.1 Reset state machine....1191
29.8.1.1 Power-on reset sequence....1192
29.8.1.2 Destructive reset sequence....1192
29.8.1.3 Functional reset sequence....1193
29.8.1.3.1 Functional reset entry sequence....1193
29.8.1.3.2 Functional reset exit sequence....1193
29.8.1.4 Idle phase....1193
29.8.2 Destructive resets....1193
29.8.3 External reset....1193
29.8.4 Functional resets....1194
29.8.5 External reset assertion control....1194
29.8.6 Functional reset escalation....1194
29.8.7 Destructive reset escalation....1195
29.8.8 Individual peripheral resets....1195

Chapter 30 Boot....1197
30.1 Introduction....1197
30.2 Features....1197
30.3 Boot modes....1197
30.3.1 Serial Boot mode....1198
30.3.1.1 End-of-line programming....1198
30.3.1.2 Non-responsive modules....1198
30.3.2 Boot from fuses and RCON....1199
30.3.2.1 Boot from fuses....1199
30.3.2.2 Boot from RCON....1199
30.3.2.2.1 Boot from parallel RCON....1199
30.3.2.2.2 Boot from serial RCON....1199
30.3.2.3 RCON mode selection....1200
30.3.2.4 Selection of boot from fuses or boot from RCON....1200
30.4 Program image....1201
30.4.1 Image Vector Table (IVT)....1201
30.4.1.1 IVT boot configuration word....1202
30.4.1.2 Life cycle configuration word....1203
30.4.2 Device Configuration Data (DCD)....1204
30.4.2.1 DCD commands....1204
30.4.2.1.1 Write Data command....1204
30.4.2.1.2 Check Data command....1206
30.4.2.1.3 NOP command....1207
30.4.3 Self-Test....1208
30.4.3.1 Self-Test boot operation....1208
30.4.3.2 Self-Test DCD image structure....1208
30.4.4 Application boot....1208
30.4.5 Serial boot....1209
30.5 Boot register descriptions....1211
30.5.1 Boot memory map....1211
30.5.2 Boot Configuration 1 (BOOT_GPR_BMR1)....1212
30.5.3 Boot Configuration 2 (BOOT_GPR_BMR2)....1212

30.5.4 SRC POR Control (SRC_POR_CTRL_REG)....1213
30.6 Boot flow....1214
30.6.1 Boot flow phases....1215
30.6.1.1 Initialize 1 boot flow phase....1216
30.6.1.2 Self-Test....1217
30.6.1.3 Initialize 2 boot flow phase....1217
30.6.2 Boot flow diagram....1219
30.7 Boot settings and fuse map....1219
30.8 BootROM impact on module registers' reset values....1219
30.9 BootROM clock configuration....1220
30.10 Image authentication and decryption....1220
30.11 Serial boot....1220
30.11.1 XOSC configuration....1221
30.11.2 Serial download protocol....1222
30.11.2.1 Serial download protocol – UART and CAN....1222
30.11.2.2 Serial download protocol – Ethernet....1223
30.11.3 FlexCAN....1224
30.11.3.1 FlexCAN I/O multiplexer configuration....1224
30.11.3.2 Clock configuration....1224
30.11.3.3 Baud rates....1224
30.11.3.4 FlexCAN boot operation....1225
30.11.4 UART....1225
30.11.4.1 UART IOMUX configuration....1225
30.11.4.2 Clock configuration....1225
30.11.4.3 UART baud rates....1226
30.11.5 Ethernet boot....1226
30.11.5.1 IOMUX configuration....1226
30.11.5.2 Functional description....1226
30.11.5.2.1 MII interface detection....1226
30.11.5.2.2 PHY identification and configuration....1228
30.11.5.2.3 GMAC_0 speed and clock configuration....1228
30.11.5.2.4 GMAC_0 controller configuration....1228
30.11.5.2.5 MAC address configuration....1228
30.11.5.2.6 DHCP handshake....1229
30.11.5.2.7 TFTP data transfer flow....1229
30.11.5.3 Flow diagram....1230
30.12 Boot from external flash memory....1231
30.12.1 QuadSPI boot....1232
30.12.1.1 Functional description....1232
30.12.1.2 QuadSPI IOMUX configuration....1235
30.12.1.3 QuadSPI clock configuration....1236
30.12.1.4 QuadSPI configuration parameters....1237
30.12.1.5 QuadSPI boot flow diagram....1239
30.12.2 µSDHC boot....1240
30.12.2.1 µSDHC IOMUX configuration....1240
30.12.2.2 µSDHC clock configuration....1241
30.12.2.3 µSDHC supported data rates....1242
30.12.2.4 µSDHC boot flow diagrams....1242
30.12.2.4.1 µSDHC boot flow – startup....1242
30.12.2.4.2 µSDHC boot flow – voltage validation....1243
30.12.2.4.3 µSDHC boot flow – chip initialization....1244
30.12.2.4.4 µSDHC boot flow – data read....1245
30.12.2.4.5 µSDHC MMC boot device initialization....1246
30.12.2.4.6 µSDHC MMC boot voltage validation....1247
30.13 Fail-Safe....1248

30.14 Boot target.....1248
30.15 Boot target watchdog.....1248
30.16 Standby boot.....1249

Chapter 31 Power Management.....1251
31.1 Introduction.....1251
31.1.1 Power management functional blocks.....1251
31.2 Power management features.....1251
31.3 Power-related I/Os.....1252
31.4 Operating modes.....1252
31.4.1 Reset mode.....1253
31.4.2 Run mode.....1253
31.4.3 Standby mode.....1254
31.4.3.1 Standby mode modules.....1254
31.4.3.2 High-level Standby mode power-down/power-up flow diagram.....1255
31.5 Operating mode transitions.....1255
31.5.1 Reset mode to Run mode.....1255
31.5.2 Configuring clocks.....1255
31.5.3 Turning on cores and partitions.....1255
31.5.4 Run mode to Reset mode.....1255
31.5.5 Run mode to Standby mode.....1255
31.5.5.1 Standby mode entry flow.....1255
31.5.5.1.1 Standby mode entry flow diagram.....1256
31.5.6 Standby mode to Run mode.....1256
31.5.6.1 Standby mode exit flow.....1256
31.5.6.1.1 Standby mode exit flow diagram.....1257
31.5.7 Standby to Reset mode.....1257
31.5.8 Wake-up pin pullup/pulldown control during Run and Standby modes.....1257
31.6 Operating mode control blocks.....1259
31.6.1 MC_ME.....1259
31.6.2 MC_PCU.....1259
31.6.3 MC_RGM.....1260
31.6.4 RTC.....1260
31.6.5 WKPU.....1260
31.6.5.1 NMI pins.....1261
31.6.5.2 Wake-up pins.....1262
31.6.5.3 Wake from RTC.....1262
31.6.6 Wake-up pin glitch filter.....1262
31.7 Power management and power domains.....1262
31.7.1 Precision voltage monitoring.....1262
31.7.2 Supply Presence Detectors (SPD).....1263
31.7.2.1 NCSPD configuration.....1264
31.7.2.2 CSPD configuration.....1266
31.7.3 Voltage overshoot and undershoot protection.....1267
31.7.4 External PMIC interface.....1267
31.7.4.1 Use case: Interfacing to external PMIC.....1267
31.7.4.2 Debug use case: Interfacing to external PMIC.....1267
31.7.4.3 Standby mode use case: Interfacing to external PMIC.....1268
31.7.4.4 Safety context for PMIC WDOG.....1269
31.8 Power mode control blocks.....1269
31.8.1 PMIC control.....1269
31.8.1.1 Static Voltage Scaling (SVS).....1269
31.8.2 PMC.....1272
31.9 Powerup/powerdown ramp rates and sequences.....1273

31.9.1 Pin behavior during powerup..... 1273
31.9.2 Chip startup and shutdown..... 1273
31.9.3 Standby mode entry..... 1274
31.9.4 Standby mode exit..... 1274
31.10 Low power modes..... 1274
31.10.1 Wake-up sources..... 1274
31.11 Power management and power domains..... 1274
31.11.1 I/O power domains..... 1274
31.11.2 Standby pads..... 1275
31.12 Glossary..... 1276

Chapter 32 Mode Entry Module (MC_ME)..... 1277
32.1 Chip-specific MC_ME information..... 1277
32.1.1 MC_ME modes..... 1277
32.1.2 MC_ME partition mapping..... 1277
32.1.2.1 MC_ME partition 0 mapping..... 1277
32.1.2.2 MC_ME partition 1 mapping..... 1278
32.1.2.3 MC_ME partition 2 mapping..... 1280
32.1.2.4 MC_ME partition 3 mapping..... 1280
32.1.3 Core controls..... 1281
32.1.4 Turning off cores and partitions..... 1281
32.1.4.1 Enable output isolation..... 1281
32.1.4.2 Disable core clocks..... 1282
32.1.4.3 Disable partition clocks..... 1282
32.1.4.4 Disable COFB clocks..... 1282
32.1.4.5 Assert partition reset signals..... 1282
32.2 Introduction..... 1282
32.3 Features..... 1283
32.4 MC_ME register descriptions..... 1283
32.4.1 MC_ME memory map..... 1284
32.4.2 Control Key Register (CTL_KEY)..... 1286
32.4.3 Mode Configuration Register (MODE_CONF)..... 1287
32.4.4 Mode Update Register (MODE_UPD)..... 1288
32.4.5 Mode Status Register (MODE_STAT)..... 1289
32.4.6 Main Core ID Register (MAIN_COREID)..... 1290
32.4.7 Partition 0 Process Configuration Register (PRTN0_PCONF)..... 1291
32.4.8 Partition 0 Process Update Register (PRTN0_PUPD)..... 1292
32.4.9 Partition 0 Status Register (PRTN0_STAT)..... 1294
32.4.10 Partition 0 COFB Set 0 Clock Status Register (PRTN0_COFB0_STAT)..... 1295
32.4.11 Partition 0 COFB Set 0 Clock Enable Register (PRTN0_COFB0_CLKEN)..... 1300
32.4.12 Partition 0 Core 0 Process Configuration Register (PRTN0_CORE0_PCONF)..... 1305
32.4.13 Partition 0 Core 0 Process Update Register (PRTN0_CORE0_PUPD)..... 1306
32.4.14 Partition 0 Core 0 Status Register (PRTN0_CORE0_STAT)..... 1307
32.4.15 Partition 0 Core 0 Address Register (PRTN0_CORE0_ADDR)..... 1308
32.4.16 Partition 0 Core 1 Process Configuration Register (PRTN0_CORE1_PCONF)..... 1308
32.4.17 Partition 0 Core 1 Process Update Register (PRTN0_CORE1_PUPD)..... 1309
32.4.18 Partition 0 Core 1 Status Register (PRTN0_CORE1_STAT)..... 1310
32.4.19 Partition 0 Core 1 Address Register (PRTN0_CORE1_ADDR)..... 1311
32.4.20 Partition 0 Core 2 Process Configuration Register (PRTN0_CORE2_PCONF)..... 1312
32.4.21 Partition 0 Core 2 Process Update Register (PRTN0_CORE2_PUPD)..... 1313
32.4.22 Partition 0 Core 2 Status Register (PRTN0_CORE2_STAT)..... 1314
32.4.23 Partition 0 Core 2 Address Register (PRTN0_CORE2_ADDR)..... 1315
32.4.24 Partition 0 Core 3 Process Configuration Register (PRTN0_CORE3_PCONF)..... 1316
32.4.25 Partition 0 Core 3 Process Update Register (PRTN0_CORE3_PUPD)..... 1317

32.4.26 Partition 0 Core 3 Status Register (PRTN0_CORE3_STAT)..... 1318
32.4.27 Partition 0 Core 3 Address Register (PRTN0_CORE3_ADDR)..... 1319
32.4.28 Partition 1 Process Configuration Register (PRTN1_PCONF).....1320
32.4.29 Partition 1 Process Update Register (PRTN1_PUPD)..... 1322
32.4.30 Partition 1 Status Register (PRTN1_STAT)..... 1323
32.4.31 Partition 1 Core 0 Process Configuration Register (PRTN1_CORE0_PCONF).....1324
32.4.32 Partition 1 Core 0 Process Update Register (PRTN1_CORE0_PUPD)..... 1325
32.4.33 Partition 1 Core 0 Status Register (PRTN1_CORE0_STAT)..... 1326
32.4.34 Partition 1 Core 0 Address Register (PRTN1_CORE0_ADDR)..... 1327
32.4.35 Partition 1 Core 1 Process Configuration Register (PRTN1_CORE1_PCONF).....1328
32.4.36 Partition 1 Core 1 Process Update Register (PRTN1_CORE1_PUPD)..... 1329
32.4.37 Partition 1 Core 1 Status Register (PRTN1_CORE1_STAT)..... 1330
32.4.38 Partition 1 Core 1 Address Register (PRTN1_CORE1_ADDR)..... 1331
32.4.39 Partition 1 Core 2 Process Configuration Register (PRTN1_CORE2_PCONF).....1332
32.4.40 Partition 1 Core 2 Process Update Register (PRTN1_CORE2_PUPD)..... 1333
32.4.41 Partition 1 Core 2 Status Register (PRTN1_CORE2_STAT)..... 1334
32.4.42 Partition 1 Core 2 Address Register (PRTN1_CORE2_ADDR)..... 1335
32.4.43 Partition 1 Core 3 Process Configuration Register (PRTN1_CORE3_PCONF).....1336
32.4.44 Partition 1 Core 3 Process Update Register (PRTN1_CORE3_PUPD)..... 1337
32.4.45 Partition 1 Core 3 Status Register (PRTN1_CORE3_STAT)..... 1338
32.4.46 Partition 1 Core 3 Address Register (PRTN1_CORE3_ADDR)..... 1339
32.4.47 Partition 2 Process Configuration Register (PRTN2_PCONF).....1340
32.4.48 Partition 2 Process Update Register (PRTN2_PUPD)..... 1342
32.4.49 Partition 2 Status Register (PRTN2_STAT)..... 1343
32.4.50 Partition 2 COFB Set 0 Clock Status Register (PRTN2_COFB0_STAT)..... 1344
32.4.51 Partition 2 COFB Set 0 Clock Enable Register (PRTN2_COFB0_CLKEN)..... 1349
32.4.52 Partition 3 Process Configuration Register (PRTN3_PCONF).....1354
32.4.53 Partition 3 Process Update Register (PRTN3_PUPD)..... 1356
32.4.54 Partition 3 Status Register (PRTN3_STAT)..... 1357
32.4.55 Partition 3 COFB Set 0 Clock Status Register (PRTN3_COFB0_STAT)..... 1358
32.4.56 Partition 3 COFB Set 0 Clock Enable Register (PRTN3_COFB0_CLKEN)..... 1363
32.5 Partition processes..... 1368
32.6 Mode transition..... 1369
32.7 Standby entry..... 1370
32.7.1 Application core shutdown..... 1370
32.7.2 Application cluster shutdown.....1371
32.7.3 Main core shutdown and standby entry..... 1372
32.8 Glossary..... 1373

Chapter 33 Power Control Unit (MC_PCU).....1374
33.1 Introduction.....1374
33.2 Power sequence FSM..... 1374
33.3 Glossary..... 1376

Chapter 34 Power Management Controller (PMC)..... 1377
34.1 Chip-specific PMC information..... 1377
34.1.1 Device Status Flag (DSF*n*) status signal mapping.....1377
34.1.1.1 Device Status Flag (DSF0) register status signals..... 1377
34.1.1.2 Device Status Flag (DSF1) register status signals..... 1378
34.1.1.3 Device Status Flag (DSF2) register status signals..... 1382
34.1.2 Supply naming reference..... 1382
34.2 Introduction.....1382
34.2.1 Features..... 1383

34.2.2 Block diagram ... 1383
34.2.2.1 SPD monitoring ... 1384
34.2.2.2 POR and NCSPD event generation ... 1384
34.2.3 Modes of operation ... 1384
34.3 Memory map and register definition ... 1385
34.3.1 Transfer error description ... 1385
34.3.2 PMC register descriptions ... 1385
34.3.2.1 PMC memory map ... 1385
34.3.2.2 Source of System Reset (SSR) ... 1385
34.3.2.3 Non-Critical Supply Presence Detector Control (NCSPD_CTL) ... 1387
34.3.2.4 NCSPD Status Register (NCSPD_STAT) ... 1390
34.3.2.5 CSPD Event Capture (CSPDEF) (CSPD_EVENT_CAPTURE) ... 1394
34.3.2.6 NCSPD Event Flag (NCSPDEF) (NCSPD_EVENT_CAPTURE) ... 1397
34.3.2.7 Device Status Flag (DSF0) (POR_WDOG_EVENT_CAPTURE0) ... 1400
34.3.2.8 Device Status Flag (DSF1) (POR_WDOG_EVENT_CAPTURE1) ... 1401
34.3.2.9 Device Status Flag (DSF2) (POR_WDOG_EVENT_CAPTURE2) ... 1401
34.4 Functional description ... 1402
34.4.1 Reset ... 1402
34.4.2 Event flag registers ... 1403
34.4.3 POR watchdog ... 1403

Chapter 35 SRAM Controller (SRAMC) ... 1404
35.1 Standby SRAM ... 1404
35.2 SRAMC registers ... 1404
35.3 Introduction ... 1404
35.3.1 Overview ... 1404
35.3.2 Features ... 1405
35.3.3 Modes of operation ... 1406
35.4 SRAMC register descriptions ... 1406
35.4.1 SRAMC memory map ... 1406
35.4.2 Platform RAM Control Register (PRAMCR) ... 1406
35.4.3 Platform RAM Initialization Address Register Start (PRAMIAS) ... 1407
35.4.4 Platform RAM Initialization Address Register End (PRAMIAE) ... 1408
35.4.5 Platform RAM Status Register (PRAMSR) ... 1409
35.4.6 Platform RAM ECC Address (PRAMECCA) ... 1411
35.5 Functional description ... 1412
35.5.1 Latency of read and write accesses ... 1412
35.5.2 Error report ... 1414
35.6 Safety considerations ... 1414
35.6.1 ECC algorithm ... 1414
35.6.2 Safety logic ... 1416
35.7 Initialization ... 1416
35.7.1 Initialization sequence examples ... 1417
35.7.1.1 Example 1: Configuration of a single pipeline, non-interleaving system, 64-bit data (1 memory bank) ... 1417
35.7.1.2 Example 2: Configuration of a single pipeline, non-interleaving system, 128-bit data (2 memory bank) ... 1417

Chapter 36 DDR Subsystem ... 1418
36.1 Chip-specific DDR information ... 1418
36.1.1 Feature configuration ... 1418
36.1.2 Mapping of masters and AXI ports of DDR controller ... 1418
36.1.3 Exclusive access monitors ... 1418

36.1.4 AXI-urgent signals of DDR memory..... 1418
36.1.5 ECC and parity errors..... 1418
36.1.6 Speculative access of DDR memory..... 1418
36.2 Overview..... 1418
36.2.1 Supported DRAM types..... 1419
36.2.2 Memory controller (MC)..... 1419
36.2.3 Physical interface (PHY)..... 1419
36.3 Architecture..... 1419
36.3.1 Block diagram..... 1419
36.3.2 Parity block..... 1420
36.3.3 Performance monitor..... 1420
36.3.3.1 Performance signals/counters..... 1420
36.3.3.1.1 Generic monitored signals..... 1421
36.3.3.1.2 Perf specific signals monitored..... 1421
36.3.3.1.3 Counter control register..... 1422
36.3.3.1.4 MRR read data register..... 1423
36.3.4 APB common interface..... 1423
36.4 Features..... 1423
36.4.1 DDR MC feature list..... 1423
36.4.2 PHY features..... 1425
36.5 Functional description..... 1427
36.5.1 Exclusive access..... 1427
36.5.2 Software coherency for AXI ports..... 1427
36.5.3 Host interface (HIF)..... 1427
36.5.4 Partial writes..... 1428
36.5.5 XPI queues information..... 1428
36.5.6 Overview of DFI interface..... 1428
36.5.7 Low power software interface..... 1429
36.5.8 Transaction service control..... 1430
36.5.9 Page policy..... 1430
36.5.9.1 Explicit auto-precharge (Per Command)..... 1430
36.5.9.2 Intelligent precharges..... 1430
36.5.9.3 Address collision handling..... 1431
36.5.9.4 Write combine..... 1431
36.5.9.4.1 Registers related to transaction service control..... 1431
36.5.10 Overview of quality of service..... 1432
36.5.10.1 Read classes..... 1432
36.5.10.2 Write classes..... 1432
36.5.11 Significance of urgent signal on QoS..... 1433
36.6 ECC scrubber..... 1433
36.6.1 Scrub period..... 1433
36.6.2 Normal operation..... 1433
36.6.3 Low power operation -software controlled..... 1434
36.6.4 Initialization writes..... 1434
36.7 Overview of inline ECC support..... 1435
36.7.1 Address map..... 1435
36.7.2 Selectable protected regions..... 1436
36.7.3 Locking of ECC region..... 1436
36.7.4 Error injection through software (ECC data poisoning)..... 1437
36.7.5 Performance..... 1437
36.7.6 Interrupts related to inline ECC..... 1437
36.8 Refresh controls..... 1438
36.8.1 Overview of refresh controls..... 1438
36.8.2 Refresh using direct software request of refresh command..... 1438
36.8.3 Refresh using auto refresh feature inside the MC..... 1438

36.8.3.1 Single refresh....1439
36.8.3.2 Burst refresh....1439
36.8.3.3 Speculative refresh....1439
36.8.4 Per-bank Refresh (LPDDR4 only)....1439
36.8.5 Fine granularity refresh (DDR4 only)....1439
36.8.6 Constraints on refresh_timerX_start_value_x32....1440
36.9 ZQ calibration....1440
36.9.1 Overview of ZQ calibration....1441
36.9.2 DDR3/DDR4 devices....1441
36.9.3 LPDDR4 devices....1441
36.9.4 Automatic and software Initiated ZQCS....1442
36.9.5 LPDDR4 ZQ reset command....1442
36.10 MRR/MPR specific sequences....1442
36.10.1 Mode register read and write operations....1442
36.10.2 Multi-purpose register (MPR for DDR4 only)....1443
36.11 ODT control....1443
36.11.1 Overview of ODT control....1443
36.12 Programming considerations....1444
36.12.1 Programming fields based on multiples of DFI clock cycles....1445
36.12.2 Converting times into clock cycles....1445
36.12.3 Programming mode....1445
36.12.4 Standby power state (LP3/IO Retention)....1446
36.13 UMCTL2_MP register descriptions....1448
36.13.1 UMCTL2_MP memory map....1448
36.13.2 Port Status (PSTAT)....1449
36.13.3 Port Common Configuration (PCCFG)....1451
36.13.4 Port n Configuration Read (PCFGR_0)....1452
36.13.5 Port n Configuration Write (PCFGW_0)....1455
36.13.6 Port n Control (PCTRL_0)....1456
36.13.7 Port n Read QoS Configuration Register 0 (PCFGQOS0_0)....1457
36.13.8 Port n Read QoS Configuration Register 1 (PCFGQOS1_0)....1459
36.13.9 Port n Write QoS Configuration Register 0 (PCFGWQOS0_0)....1460
36.13.10 Port n Write QoS Configuration Register 1 (PCFGWQOS1_0)....1463
36.13.11 Port n Configuration Read (PCFGR_1)....1463
36.13.12 Port n Configuration Write (PCFGW_1)....1465
36.13.13 Port n Control (PCTRL_1)....1466
36.13.14 Port n Read QoS Configuration Register 0 (PCFGQOS0_1)....1467
36.13.15 Port n Read QoS Configuration Register 1 (PCFGQOS1_1)....1468
36.13.16 Port n Write QoS Configuration Register 0 (PCFGWQOS0_1)....1469
36.13.17 Port n Write QoS Configuration Register 1 (PCFGWQOS1_1)....1471
36.13.18 Port n Configuration Read (PCFGR_2)....1472
36.13.19 Port n Configuration Write (PCFGW_2)....1473
36.13.20 Port n Control (PCTRL_2)....1474
36.13.21 Port n Read QoS Configuration Register 0 (PCFGQOS0_2)....1475
36.13.22 Port n Read QoS Configuration Register 1 (PCFGQOS1_2)....1477
36.13.23 Port n Write QoS Configuration Register 0 (PCFGWQOS0_2)....1477
36.13.24 Port n Write QoS Configuration Register 1 (PCFGWQOS1_2)....1479
36.13.25 Scrubber Control (SBRCTL)....1480
36.13.26 Scrubber Status (SBRSTAT)....1481
36.13.27 Scrubber Write Data Pattern0 (SBRWDATA0)....1482
36.14 UMCTL2_REGS register descriptions....1482
36.14.1 UMCTL2_REGS memory map....1482
36.14.2 Master (MSTR)....1487
36.14.3 Operating mode status (STAT)....1491
36.14.4 Mode Register read/write control 0 (MRCTRL0)....1493

36.14.5 Mode Register Read/Write Control 1 (MRCTRL1)....1496
36.14.6 Mode Register Read/Write Status (MRSTAT)....1497
36.14.7 Mode Register Read/Write Control 2 (MRCTRL2)....1498
36.14.8 Temperature derate enable (DERATEEN)....1499
36.14.9 Temperature derate interval (DERATEINT)....1501
36.14.10 Temperature derate control (DERATECTL)....1502
36.14.11 Low power control (PWRCTL)....1503
36.14.12 Low Power Timing (PWRTMG)....1506
36.14.13 Hardware low power control (HWLPCTL)....1507
36.14.14 Refresh Control 0 (RFSHCTL0)....1509
36.14.15 Refresh control 1 (RFSHCTL1)....1512
36.14.16 Refresh Control 3 (RFSHCTL3)....1513
36.14.17 Refresh Timing (RFSHTMG)....1514
36.14.18 Refresh timing 1 (RFSHTMG1)....1516
36.14.19 ECC Configuration 0 (ECCCFG0)....1517
36.14.20 ECC Configuration 1 (ECCCFG1)....1520
36.14.21 SECDED ECC Status (ECCSTAT)....1522
36.14.22 ECC Control (ECCCTL)....1524
36.14.23 ECC Error Counter (ECCERRCNT)....1527
36.14.24 ECC Corrected Error Address 0 (ECCCADDR0)....1528
36.14.25 ECC Corrected Error Address 1 (ECCCADDR1)....1529
36.14.26 ECC Corrected Syndrome 0 (ECCCSYN0)....1530
36.14.27 ECC Corrected Syndrome 1 (ECCCSYN1)....1531
36.14.28 ECC Corrected Syndrome 2 (ECCCSYN2)....1531
36.14.29 ECC Corrected Data Bit Mask 0 (ECCBITMASK0)....1532
36.14.30 ECC Corrected Data Bit Mask 1 (ECCBITMASK1)....1533
36.14.31 ECC Corrected Data Bit Mask 2 (ECCBITMASK2)....1534
36.14.32 ECC Uncorrected Error Address 0 (ECCUADDR0)....1535
36.14.33 ECC Uncorrected Error Address 1 (ECCUADDR1)....1536
36.14.34 ECC Uncorrected Syndrome 0 (ECCUSYN0)....1537
36.14.35 ECC Uncorrected Syndrome 1 (ECCUSYN1)....1538
36.14.36 ECC Uncorrected Syndrome 2 (ECCUSYN2)....1538
36.14.37 ECC Data Poisoning Address 0 (ECCPOISONADDR0)....1539
36.14.38 ECC Data Poisoning Address 1 (ECCPOISONADDR1)....1540
36.14.39 CRC Parity Control 0 (CRCPARCTL0)....1541
36.14.40 CRC Parity Control 1 (CRCPARCTL1)....1542
36.14.41 CRC Parity Status (CRCPARSTAT)....1544
36.14.42 DRAM Initialization 0 (INIT0)....1544
36.14.43 DRAM Initialization 1 (INIT1)....1546
36.14.44 DRAM Initialization 2 (INIT2)....1547
36.14.45 DRAM Initialization 3 (INIT3)....1548
36.14.46 DRAM Initialization 4 (INIT4)....1549
36.14.47 DRAM Initialization 5 (INIT5)....1550
36.14.48 DRAM Initialization 6 (INIT6)....1551
36.14.49 DRAM Initialization 7 (INIT7)....1552
36.14.50 DIMM Control (DIMMCTL)....1553
36.14.51 Rank Control (RANKCTL)....1555
36.14.52 DRAM Timing 0 (DRAMTMG0)....1557
36.14.53 DRAM Timing 1 (DRAMTMG1)....1559
36.14.54 DRAM Timing 2 (DRAMTMG2)....1561
36.14.55 DRAM Timing 3 (DRAMTMG3)....1564
36.14.56 DRAM Timing 4 (DRAMTMG4)....1566
36.14.57 DRAM Timing 5 (DRAMTMG5)....1568
36.14.58 DRAM Timing 6 (DRAMTMG6)....1570
36.14.59 DRAM Timing 7 (DRAMTMG7)....1572

36.14.60 DRAM Timing 8 (DRAMTMG8)....1573
36.14.61 DRAM Timing 9 (DRAMTMG9)....1574
36.14.62 DRAM Timing 10 (DRAMTMG10)....1575
36.14.63 DRAM Timing 11 (DRAMTMG11)....1577
36.14.64 DRAM Timing 12 (DRAMTMG12)....1578
36.14.65 DRAM Timing 13 (DRAMTMG13)....1579
36.14.66 DRAM Timing 14 (DRAMTMG14)....1580
36.14.67 DRAM Timing 15 (DRAMTMG15)....1581
36.14.68 ZQ Control 0 (ZQCTL0)....1582
36.14.69 ZQ Control 1 (ZQCTL1)....1584
36.14.70 ZQ Control 2 (ZQCTL2)....1585
36.14.71 ZQ Status (ZQSTAT)....1586
36.14.72 DFI Timing 0 (DFITMG0)....1586
36.14.73 DFI Timing 1 (DFITMG1)....1589
36.14.74 DFI Low Power Configuration 0 (DFILPCFG0)....1591
36.14.75 DFI Low Power Configuration 1 (DFILPCFG1)....1592
36.14.76 DFI Update 0 (DFIUPD0)....1593
36.14.77 DFI Update 1 (DFIUPD1)....1595
36.14.78 DFI Update 2 (DFIUPD2)....1596
36.14.79 DFI Miscellaneous Control (DFIMISC)....1597
36.14.80 DFI Timing 2 (DFITMG2)....1599
36.14.81 DFI Timing 3 (DFITMG3)....1600
36.14.82 DFI Status (DFISTAT)....1601
36.14.83 DM/DBI Control (DBICTL)....1602
36.14.84 DFI PHY Master (DFIPHYMSTR)....1603
36.14.85 Address Map 0 (ADDRMAP0)....1604
36.14.86 Address Map 1 (ADDRMAP1)....1605
36.14.87 Address Map 2 (ADDRMAP2)....1606
36.14.88 Address Map 3 (ADDRMAP3)....1608
36.14.89 Address Map 4 (ADDRMAP4)....1611
36.14.90 Address Map 5 (ADDRMAP5)....1613
36.14.91 Address Map 6 (ADDRMAP6)....1615
36.14.92 Address Map 7 (ADDRMAP7)....1617
36.14.93 Address Map 8 (ADDRMAP8)....1618
36.14.94 Address Map 9 (ADDRMAP9)....1619
36.14.95 Address Map 10 (ADDRMAP10)....1620
36.14.96 Address Map 11 (ADDRMAP11)....1621
36.14.97 ODT Configuration (ODTCFG)....1622
36.14.98 ODT/Rank Map (ODTMAP)....1624
36.14.99 Scheduler Control (SCHED)....1625
36.14.100 Scheduler Control 1 (SCHED1)....1628
36.14.101 High Priority Read CAM 1 (PERFHPR1)....1628
36.14.102 Low Priority Read CAM 1 (PERFLPR1)....1630
36.14.103 Write CAM 1 (PERFWR1)....1631
36.14.104 Debug 0 (DBG0)....1632
36.14.105 Debug 1 (DBG1)....1633
36.14.106 CAM Debug (DBGCAM)....1634
36.14.107 Command Debug (DBGCMD)....1636
36.14.108 Status Debug (DBGSTAT)....1637
36.14.109 CAM Debug 1 (DBGCAM1)....1638
36.14.110 Software Register Programming Control Enable (SWCTL)....1639
36.14.111 Software Register Programming Control Status (SWSTAT)....1640
36.14.112 On-Chip Parity Configuration 0 (OCPARCFG0)....1640
36.14.113 On-Chip Parity Configuration 1 (OCPARCFG1)....1643
36.14.114 On-Chip Parity Status 0 (OCPARSTAT0)....1644

36.14.115 On-Chip Parity Status 1 (OCPARSTAT1)..... 1645
36.14.116 On-Chip Parity Status 2 (OCPARSTAT2)..... 1647
36.14.117 AXI Poison Configuration (POISONCFG)..... 1647
36.14.118 AXI Poison Status (POISONSTAT)..... 1649
36.14.119 Advanced ECC Index (ADVECCINDEX)..... 1651
36.14.120 ECC Poison Pattern 0 (ECCPOISONPAT0)..... 1652
36.14.121 ECC Poison Pattern 2 (ECCPOISONPAT2)..... 1652
36.14.122 Address protection within ECC Status (ECCAPSTAT)..... 1653
36.14.123 Register parity configuration (REGPARCFG)..... 1654
36.14.124 Register parity status (REGPARSTAT)..... 1655
36.14.125 On-Chip command/Address Protection Configuration (OCCAPCFG)..... 1656
36.14.126 On-Chip command/Address Protection Status (OCCAPSTAT)..... 1657
36.14.127 On-Chip command/Address Protection Configuration 1 (OCCAPCFG1)..... 1659
36.14.128 On-Chip command/Address Protection Status 1 (OCCAPSTAT1)..... 1661
36.14.129 Temperature Derate Status (DERATESTAT)..... 1662
36.15 Performance monitor register descriptions..... 1663
36.15.1 PERF memory map..... 1663
36.15.2 Counter control 0 (COUNTER_0_CTRL_REG_ADDR)..... 1664
36.15.3 Counter control 1 (COUNTER_1_CTRL_REG_ADDR)..... 1665
36.15.4 Counter control 2 (COUNTER_2_CTRL_REG_ADDR)..... 1668
36.15.5 Counter control 3 (COUNTER_3_CTRL_REG_ADDR)..... 1670
36.15.6 Counter Data (COUNTER_0_DATA_REG_ADDR - COUNTER_3_DATA_REG_ADDR). 1673
36.15.7 Mode Register Read 0 address (MRR_0_DATA_REG_ADDR)..... 1674
36.15.8 Mode Register Read 1 address (MRR_1_DATA_REG_ADDR)..... 1675

Chapter 37 DDR General Purpose Registers (DDR_GPR)..... 1678
37.1 Introduction..... 1678
37.2 DDR_GPR register descriptions..... 1678
37.2.1 DDR_GPR memory map..... 1678
37.2.2 Configuration 0 (DDR_Config_0)..... 1678
37.2.3 Configuration 1 (DDR_Config_1)..... 1680
37.2.4 Configuration 2 (DDR_Config_2)..... 1681
37.2.5 Configuration 3 (DDR_Config_3)..... 1682
37.2.6 Configuration 4 (DDR_Config_4)..... 1683
37.2.7 Configuration 5 (DDR_Config_5)..... 1685
37.2.8 LP3 IO Retention Control (DDR_RET_CONTROL)..... 1687

Chapter 38 Quad Serial Peripheral Interface (QuadSPI)..... 1689
38.1 Chip-specific QuadSPI configuration..... 1689
38.1.1 Overview..... 1689
38.1.2 Supported modes..... 1689
38.1.3 QuadSPI clocking..... 1690
38.1.4 QuadSPI limitations..... 1690
38.1.5 Chip-specific feature configuration..... 1690
38.2 Introduction..... 1690
38.2.1 Features..... 1691
38.2.2 RX buffer push event..... 1691
38.2.3 RX buffer POP event..... 1691
38.2.4 Block diagram..... 1691
38.2.5 QuadSPI modes of operation..... 1692
38.3 External signal description..... 1693
38.3.1 Driving external signals..... 1694
38.4 Memory map and register definition..... 1696

38.4.1 Register write access..... 1696
38.4.2 QuadSPI register descriptions..... 1696
38.4.2.1 QuadSPI memory map..... 1696
38.4.2.2 Module Configuration Register (MCR)..... 1698
38.4.2.3 IP Configuration Register (IPCR)..... 1703
38.4.2.4 Flash Memory Configuration Register (FLSHCR)..... 1704
38.4.2.5 Buffer 0 Configuration Register (BUF0CR)..... 1705
38.4.2.6 Buffer 1 Configuration Register (BUF1CR)..... 1706
38.4.2.7 Buffer 2 Configuration Register (BUF2CR)..... 1708
38.4.2.8 Buffer 3 Configuration Register (BUF3CR)..... 1709
38.4.2.9 Buffer Generic Configuration Register (BFGENCR)..... 1710
38.4.2.10 Buffer 0 Top Index Register (BUF0IND)..... 1711
38.4.2.11 Buffer 1 Top Index Register (BUF1IND)..... 1712
38.4.2.12 Buffer 2 Top Index Register (BUF2IND)..... 1713
38.4.2.13 AHB Write Configuration Register (AWRCR)..... 1714
38.4.2.14 DLL Flash Memory A Configuration Register (DLLCRA)..... 1716
38.4.2.15 DLL Flash Memory B Configuration Register (DLLCRB)..... 1718
38.4.2.16 Parity Configuration Register (PARITYCR)..... 1720
38.4.2.17 Serial Flash Memory Address Register (SFAR)..... 1722
38.4.2.18 Serial Flash Memory Address Configuration Register (SFACR)..... 1723
38.4.2.19 Sampling Register (SMPR)..... 1725
38.4.2.20 RX Buffer Status Register (RBSR)..... 1726
38.4.2.21 RX Buffer Control Register (RBCT)..... 1727
38.4.2.22 AHB Write Status Register (AWRSR)..... 1728
38.4.2.23 DLL Status Register (DLLSR)..... 1729
38.4.2.24 Data Learning Configuration Register (DLCR)..... 1731
38.4.2.25 Data Learning Status Flash Memory A Register (DLSR_FA)..... 1733
38.4.2.26 Data Learning Status Flash Memory B Register (DLSR_FB)..... 1734
38.4.2.27 TX Buffer Status Register (TBSR)..... 1735
38.4.2.28 TX Buffer Data Register (TBDR)..... 1736
38.4.2.29 TX Buffer Control Register (TBCT)..... 1737
38.4.2.30 Status Register (SR)..... 1737
38.4.2.31 Flag Register (FR)..... 1740
38.4.2.32 Interrupt and DMA Request Select and Enable Register (RSER)..... 1744
38.4.2.33 Sequence Pointer Clear Register (SPTRCLR)..... 1748
38.4.2.34 Serial Flash Memory A1 Top Address Register (SFA1AD)..... 1749
38.4.2.35 Serial Flash Memory A2 Top Address Register (SFA2AD)..... 1750
38.4.2.36 Serial Flash Memory B1 Top Address Register (SFB1AD)..... 1751
38.4.2.37 Serial Flash Memory B2 Top Address Register (SFB2AD)..... 1752
38.4.2.38 Data Learn Pattern Register (DLPR)..... 1753
38.4.2.39 Flash Memory A Failing Address Status Register (FAILA_ADDR)..... 1754
38.4.2.40 flash Memory B Failing Address Status Register (FAILB_ADDR)..... 1755
38.4.2.41 RX Buffer Data Register (RBDR0 - RBDR63)..... 1755
38.4.2.42 LUT Key Register (LUTKEY)..... 1756
38.4.2.43 LUT Lock Configuration Register (LCKCR)..... 1757
38.4.2.44 LUT Register (LUT0)..... 1758
38.4.2.45 LUT Register (LUT1)..... 1760
38.4.2.46 LUT Register (LUT2 - LUT79)..... 1761
38.4.3 Serial flash memory address assignment..... 1762
38.5 Flash memory mapped AMBA bus..... 1763
38.5.1 AHB bus access read considerations..... 1764
38.5.2 Memory-mapped serial flash memory data—individual flash memory mode on flash memory A..... 1765
38.5.3 Memory-mapped serial flash memory data—individual flash memory mode on flash memory B..... 1766

38.5.4 Parallel flash memory mode....1766
38.5.5 ARDB register descriptions....1767
38.5.5.1 ARDB memory map....1767
38.5.5.2 AHB RX Data Buffer Register (ARDB0 - ARDB127)....1768
38.6 Functional description....1769
38.6.1 Serial flash memory access schemes....1769
38.6.2 Normal mode....1769
38.6.2.1 Programmable sequence engine....1770
38.6.2.2 Flexible read xAHB buffers....1773
38.6.2.3 Abort mechanism during AHB read....1774
38.6.2.4 HBURST support with AHB read....1774
38.6.2.5 LUT....1775
38.6.2.6 Issuing SFM commands....1776
38.6.2.7 Flash memory programming....1777
38.6.2.8 Flash memory read....1778
38.6.2.9 AHB write....1780
38.6.2.10 Byte ordering of serial flash memory read data....1781
38.6.2.11 Normal mode interrupt and DMA requests....1783
38.6.2.12 TX buffer operation....1785
38.6.2.13 Address scheme....1785
38.6.2.14 OTFAD support....1786
38.6.3 Module Disable mode....1786
38.6.4 Leaving Module Disable mode....1787
38.7 Initialization/application information....1787
38.7.1 Power up and reset....1787
38.7.2 Available status/flag information....1787
38.7.2.1 IP commands....1787
38.7.2.2 AHB commands....1787
38.7.2.3 SFM commands....1788
38.7.2.4 Overview of error flags....1788
38.7.2.5 IP bus and AHB access command collisions....1789
38.7.3 Flash memory device selection....1789
38.7.4 DMA usage....1789
38.7.4.1 DMA usage in normal mode....1789
38.7.4.1.1 Bandwidth considerations....1789
38.7.5 Flash memory devices address mapping....1792
38.7.5.1 Single mode....1793
38.7.5.2 Serial mode....1793
38.7.5.3 Parallel mode....1794
38.8 Byte ordering – endianness....1795
38.8.1 Programming flash memory data....1795
38.8.2 Reading flash memory data into the RX buffer....1796
38.8.2.1 Readout of the RX buffer through RBDRn....1796
38.8.2.2 Readout of the RX buffer through ARDBn....1796
38.8.3 Reading flash memory data into the AHB buffer....1796
38.8.3.1 Readout of the AHB buffer through memory-mapped read....1796
38.9 Driving flash memory control signals in single and dual modes....1796
38.10 Serial flash memory devices....1797
38.10.1 Example sequences....1797
38.10.1.1 Read command (Spansion hyperflash memory)....1797
38.10.1.2 Read status register (Spansion hyperflash memory)....1798
38.10.1.3 Word program (Spansion hyperflash memory)....1798
38.10.1.4 Fast read sequence (Macronix/Numonyx/Spansion/Winbond)....1800
38.10.1.5 Fast dual I/O DT read sequence (Macronix)....1800
38.10.1.6 Fast read quad output (Winbond)....1801

38.10.1.7 4 x I/O read enhance performance mode (XIP) (Macronix)....1801
38.10.1.8 Dual command page program (Numonyx)....1801
38.10.1.9 Sector erase (Macronix/Numonyx/Spansion)....1802
38.10.1.10 Read status register (Macronix/Numonyx/Spansion/Winbond)....1802
38.10.1.11 Data learn instruction sequence....1802
38.10.2 Dual-die flash memories....1804
38.10.3 Boot initialization sequence....1804
38.11 Sampling of serial flash memory input data....1805
38.11.1 Basic description....1805
38.11.2 DQS sampling method....1805
38.11.2.1 Basic description....1805
38.11.2.2 External DQS....1806
38.11.2.3 Pad loopback....1807
38.11.2.4 External center-aligned read strobe....1807
38.12 Data learning....1808
38.12.1 Basic description....1808
38.12.2 DQS sampling method....1808
38.12.3 Programming the data learning pattern in flash memory....1810
38.13 DLL and delay chain usage....1810
38.14 Data input hold requirement of flash memory....1811
38.15 Glossary....1812

Chapter 39 Ultra Secured Digital Host Controller (uSDHC)....1813
39.1 Chip-specific uSDHC information....1813
39.1.1 Chip-specific uSDHC information....1813
39.2 Introduction....1813
39.2.1 Features....1814
39.2.2 Modes and operations....1815
39.2.2.1 Data transfer modes....1815
39.3 External Signals....1815
39.3.1 Signals overview....1816
39.4 uSDHC memory map/register definition....1817
39.4.1 uSDHC register descriptions....1817
39.4.1.1 uSDHC memory map....1817
39.4.1.2 DMA System Address (DS_ADDR)....1819
39.4.1.3 Block Attributes (BLK_ATT)....1820
39.4.1.4 Command Argument (CMD_ARG)....1822
39.4.1.5 Command Transfer Type (CMD_XFR_TYP)....1823
39.4.1.6 Command Response0 (CMD_RSP0)....1826
39.4.1.7 Command Response1 (CMD_RSP1)....1827
39.4.1.8 Command Response2 (CMD_RSP2)....1828
39.4.1.9 Command Response3 (CMD_RSP3)....1828
39.4.1.10 Data Buffer Access Port (DATA_BUFF_ACC_PORT)....1830
39.4.1.11 Present State (PRES_STATE)....1831
39.4.1.12 Protocol Control (PROT_CTRL)....1837
39.4.1.13 System Control (SYS_CTRL)....1841
39.4.1.14 Interrupt Status (INT_STATUS)....1846
39.4.1.15 Interrupt Status Enable (INT_STATUS_EN)....1852
39.4.1.16 Interrupt Signal Enable (INT_SIGNAL_EN)....1856
39.4.1.17 Auto CMD12 Error Status (AUTOCMD12_ERR_STATUS)....1859
39.4.1.18 Host Controller Capabilities (HOST_CTRL_CAP)....1862
39.4.1.19 Watermark Level (WTMK_LVL)....1865
39.4.1.20 Mixer Control (MIX_CTRL)....1866
39.4.1.21 Force Event (FORCE_EVENT)....1869

39.4.1.22 ADMA Error Status (ADMA_ERR_STATUS)..... 1872
39.4.1.23 ADMA System Address (ADMA_SYS_ADDR)..... 1873
39.4.1.24 DLL (Delay Line) Control (DLL_CTRL)..... 1874
39.4.1.25 DLL Status (DLL_STATUS)..... 1876
39.4.1.26 CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS)..... 1877
39.4.1.27 Strobe DLL control (STROBE_DLL_CTRL)..... 1879
39.4.1.28 Strobe DLL status (STROBE_DLL_STATUS)..... 1881
39.4.1.29 Vendor Specific Register (VEND_SPEC)..... 1882
39.4.1.30 MMC Boot (MMC_BOOT)..... 1885
39.4.1.31 Vendor Specific 2 Register (VEND_SPEC2)..... 1887
39.4.1.32 Tuning Control (TUNING_CTRL)..... 1889
39.4.1.33 Command Queuing Version (CQVER)..... 1890
39.4.1.34 Command Queuing Capabilities (CQCAP)..... 1891
39.4.1.35 Command Queuing Configuration (CQCFG)..... 1892
39.4.1.36 Command Queuing Control (CQCTL)..... 1894
39.4.1.37 Command Queuing Interrupt Status (CQIS)..... 1895
39.4.1.38 Command Queuing Interrupt Status Enable (CQISTE)..... 1896
39.4.1.39 Command Queuing Interrupt Signal Enable (CQISGE)..... 1898
39.4.1.40 Command Queuing Interrupt Coalescing (CQIC)..... 1899
39.4.1.41 Command Queuing Task Descriptor List Base Address (CQTDLBA)..... 1901
39.4.1.42 Command Queuing Task Descriptor List Base Address Upper 32 Bits (CQTDLBAU)..... 1902
39.4.1.43 Command Queuing Task Doorbell (CQTDBR)..... 1902
39.4.1.44 Command Queuing Task Completion Notification (CQTCN)..... 1904
39.4.1.45 Command Queuing Device Queue Status (CQDQS)..... 1904
39.4.1.46 Command Queuing Device Pending Tasks (CQDPT)..... 1905
39.4.1.47 Command Queuing Task Clear (CQTCLR)..... 1906
39.4.1.48 Command Queuing Send Status Configuration 1 (CQSSC1)..... 1907
39.4.1.49 Command Queuing Send Status Configuration 2 (CQSSC2)..... 1908
39.4.1.50 Command Queuing Command Response for Direct-Command Task (CQCRDCT)..... 1909
39.4.1.51 Command Queuing Response Mode Error Mask (CQRMEM)..... 1910
39.4.1.52 Command Queuing Task Error Information (CQTERRI)..... 1911
39.4.1.53 Command Queuing Command Response Index (CQCRI)..... 1913
39.4.1.54 Command Queuing Command Response Argument (CQCRA)..... 1914
39.5 Functional Description..... 1914
39.5.1 Data buffer..... 1914
39.5.1.1 Write operation sequence..... 1915
39.5.1.2 Read operation sequence..... 1916
39.5.1.3 Data buffer and block size..... 1916
39.5.1.4 Dividing large data transfer..... 1916
39.5.2 DMA AXI interface..... 1917
39.5.2.1 Internal DMA request..... 1917
39.5.2.2 AXI master interface..... 1918
39.5.2.3 ADMA engine..... 1918
39.5.2.3.1 ADMA concept and descriptor format..... 1918
39.5.2.3.2 ADMA interrupt..... 1921
39.5.2.3.3 ADMA error..... 1921
39.5.3 Register bank with IP bus interface..... 1922
39.5.3.1 SD protocol unit..... 1922
39.5.3.2 SD control misc..... 1923
39.5.3.3 SD clock control..... 1923
39.5.3.4 Command control..... 1923
39.5.3.5 Data control..... 1923
39.5.4 Clock and reset manager..... 1924
39.5.5 Clock generator..... 1924
39.5.6 SDIO card interrupt..... 1924

39.5.6.1 Interrupts in 1-bit mode....1924
39.5.6.2 Interrupt in 4-bit mode....1924
39.5.6.3 Card interrupt handling....1925
39.5.7 Card insertion and removal detection....1926
39.5.8 Power management and wakeup events....1926
39.5.8.1 Setting wakeup events....1927
39.5.9 MMC fast boot....1927
39.5.9.1 Boot operation....1927
39.5.9.2 Alternative boot operation....1928
39.6 Initialization/Application of uSDHC....1928
39.6.1 Command send & response receive basic operation....1929
39.6.2 Card identification mode....1929
39.6.2.1 Card detect....1929
39.6.2.2 Reset....1930
39.6.2.3 Voltage validation....1931
39.6.2.4 Card registry....1932
39.6.3 Card access....1933
39.6.3.1 Block write....1933
39.6.3.1.1 Normal Write....1933
39.6.3.1.2 DDR write....1934
39.6.3.1.3 Write with Pause....1934
39.6.3.2 Block read....1935
39.6.3.2.1 Normal read....1935
39.6.3.2.2 DDR read....1936
39.6.3.2.3 Read with Pause....1936
39.6.3.2.4 Delay Line (DLL) in read path....1937
39.6.3.3 Suspend Resume....1938
39.6.3.3.1 Suspend....1938
39.6.3.3.2 Resume....1939
39.6.3.4 ADMA usage....1939
39.6.3.5 Transfer error....1939
39.6.3.5.1 CRC error....1939
39.6.3.5.2 Internal DMA error....1939
39.6.3.5.3 Transfer ADMA error....1940
39.6.3.5.4 Auto CMD12 error....1940
39.6.3.6 Card interrupt....1940
39.6.4 Switch function....1940
39.6.4.1 Query, enable, and disable SDIO high-speed mode....1941
39.6.4.2 Query, enable, and disable SD high-speed mode/DDR50/SDR50/SDR104....1941
39.6.4.3 Query, enable, and disable MMC high-speed mode....1942
39.6.4.4 Set MMC bus width....1942
39.6.5 ADMA operation....1942
39.6.5.1 ADMA1 operation....1943
39.6.5.2 ADMA2 operation....1943
39.6.6 Fast boot operation....1944
39.6.6.1 Normal fast boot flow....1944
39.6.6.2 Alternative fast boot flow....1944
39.6.6.3 Fast boot application case (in DMA mode)....1945
39.7 Commands for MMC/SD/SDIO....1946
39.8 Software Restrictions....1952
39.8.1 Initialization active....1952
39.8.2 Software polling procedure....1952
39.8.3 Suspend operation....1953
39.8.4 Data length setting....1953
39.8.5 (A)DMA address setting....1953

39.8.6 Data port access ... 1953
39.8.7 Change clock frequency ... 1953
39.8.8 Multi-block read ... 1953

Chapter 40 System Timer Module (STM) ... 1954
40.1 Chip-specific STM information ... 1954
40.1.1 STM instances ... 1954
40.2 Introduction ... 1954
40.3 Features ... 1954
40.4 STM block diagram ... 1955
40.5 STM_TS register descriptions ... 1955
40.5.1 STM_TS memory map ... 1955
40.5.2 Control (CR) ... 1956
40.5.3 Count (CNT) ... 1957
40.5.4 Channel Control (CCR0 - CCR3) ... 1958
40.5.5 Channel Interrupt (CIR0 - CIR3) ... 1959
40.5.6 Channel Compare (CMP0 - CMP3) ... 1960
40.6 STM register descriptions ... 1961
40.6.1 STM memory map ... 1961
40.6.2 Control (CR) ... 1961
40.6.3 Count (CNT) ... 1963
40.6.4 Channel Control (CCR0 - CCR3) ... 1963
40.6.5 Channel Interrupt (CIR0 - CIR3) ... 1964
40.6.6 Channel Compare (CMP0 - CMP3) ... 1965
40.7 Configure the timer ... 1966
40.8 Configure the compare channels ... 1966
40.9 Respond to compare channel events ... 1966
40.10 Functional description ... 1966
40.10.1 Count-up timer ... 1966
40.10.2 Compare channels ... 1967
40.10.3 Behavior in different chip modes ... 1967
40.11 Glossary ... 1967

Chapter 41 Software Watchdog Timer (SWT) ... 1968
41.1 Chip-specific SWT information ... 1968
41.1.1 SWT instances ... 1968
41.1.2 SWT time-out events: configuring reset and interrupt requests ... 1968
41.1.3 Reset SWT only when a timeout triggers a reset request ... 1968
41.2 Introduction ... 1968
41.2.1 Features ... 1969
41.3 Initialize the SWT ... 1969
41.3.1 Select the clock source ... 1969
41.3.2 Set the timeout period ... 1969
41.3.3 Control timeout behavior ... 1969
41.3.3.1 Select reset request on timeout ... 1969
41.3.3.2 Select interrupt on initial timeout ... 1970
41.3.4 Configure locking and unlocking ... 1970
41.3.4.1 Enable the hard lock ... 1970
41.3.4.2 Enable the soft lock ... 1970
41.3.4.3 Unlock the soft lock ... 1970
41.3.4.4 Avoid accidental soft unlock ... 1970
41.3.5 Select behavior on an invalid access ... 1971
41.4 Enable the SWT ... 1971

41.5 Initiate service operations....1971
41.5.1 Initiate a fixed service sequence....1971
41.5.2 Initiate a keyed service sequence....1971
41.5.3 Select window service mode....1972
41.6 Reset only the SWT....1972
41.7 Test SWT operation....1972
41.8 Functional description....1972
41.8.1 Behavior in different chip modes....1972
41.8.2 Register access latency....1973
41.8.3 Service key generation....1973
41.9 SWT register descriptions....1973
41.9.1 SWT memory map....1973
41.9.2 Control (CR)....1974
41.9.3 Interrupt (IR)....1977
41.9.4 Timeout (TO)....1978
41.9.5 Window (WN)....1979
41.9.6 Service (SR)....1980
41.9.7 Counter Output (CO)....1981
41.9.8 Service Key (SK)....1982
41.9.9 Reset Request (RRR)....1982

Chapter 42 Periodic Interrupt Timer (PIT)....1984
42.1 Chip-specific PIT information....1984
42.1.1 PIT module instance details....1984
42.1.2 PIT triggers....1984
42.2 Introduction....1984
42.2.1 Block diagram....1984
42.2.2 Features....1985
42.3 Modes of operation....1985
42.4 PIT register descriptions....1985
42.4.1 PIT memory map....1986
42.4.2 PIT Module Control Register (MCR)....1987
42.4.3 PIT Upper Lifetime Timer Register (LTMR64H)....1988
42.4.4 PIT Lower Lifetime Timer Register (LTMR64L)....1989
42.4.5 Timer Load Value Register (LDVAL0 - LDVAL6)....1989
42.4.6 Current Timer Value Register (CVAL0 - CVAL6)....1991
42.4.7 Timer Control Register (TCTRL0 - TCTRL6)....1992
42.4.8 Timer Flag Register (TFLG0 - TFLG6)....1994
42.5 Functional description....1995
42.5.1 General operation....1995
42.5.1.1 Timers....1995
42.5.1.2 Debug mode....1996
42.5.2 Interrupts....1996
42.5.3 Chained timers....1996
42.6 Initialization and application information....1996
42.7 Example configuration for chained timers....1997
42.8 Example configuration for the Lifetime Timer....1998

Chapter 43 FlexTimer (FTM)....1999
43.1 Chip-specific FTM information....1999
43.1.1 FTM instances....1999
43.1.2 FTM global time base....1999
43.1.3 Initialization triggers and external triggers....2000

43.2 Introduction....2000
43.2.1 Features....2000
43.2.2 Modes of operation....2001
43.2.3 Block Diagram....2001
43.3 Signal Description....2002
43.4 Memory Map and Register Definition....2003
43.4.1 Memory map....2003
43.4.2 Register descriptions....2003
43.4.3 FTM register descriptions....2003
43.4.3.1 FTM memory map....2003
43.4.3.2 Status And Control (SC)....2005
43.4.3.3 Counter (CNT)....2008
43.4.3.4 Modulo (MOD)....2009
43.4.3.5 Channel (n) Status And Control (C0SC - C5SC)....2010
43.4.3.6 Channel (n) Value (C0V - C5V)....2012
43.4.3.7 Counter Initial Value (CNTIN)....2013
43.4.3.8 Capture And Compare Status (STATUS)....2014
43.4.3.9 Features Mode Selection (MODE)....2015
43.4.3.10 Synchronization (SYNC)....2017
43.4.3.11 Initial State For Channels Output (OUTINIT)....2019
43.4.3.12 Output Mask (OUTMASK)....2020
43.4.3.13 Function For Linked Channels (COMBINE)....2021
43.4.3.14 Deadtime Configuration (DEADTIME)....2025
43.4.3.15 FTM External Trigger (EXTTRIG)....2027
43.4.3.16 Channels Polarity (POL)....2029
43.4.3.17 Fault Mode Status (FMS)....2030
43.4.3.18 Input Capture Filter Control (FILTER)....2031
43.4.3.19 Quadrature Decoder Control And Status (QDCTRL)....2032
43.4.3.20 Configuration (CONF)....2034
43.4.3.21 Synchronization Configuration (SYNCONF)....2036
43.4.3.22 FTM Inverting Control (INVCTRL)....2038
43.4.3.23 FTM Software Output Control (SWOCTRL)....2039
43.4.3.24 FTM PWM Load (PWMLOAD)....2040
43.4.3.25 Half Cycle Register (HCR)....2042
43.4.3.26 Mirror of Modulo Value (MOD_MIRROR)....2043
43.4.3.27 Mirror of Channel (n) Match Value (C0V_MIRROR - C5V_MIRROR)....2044
43.5 Functional Description....2045
43.5.1 Clock source....2045
43.5.1.1 Counter clock source....2045
43.5.2 Prescaler....2045
43.5.3 Counter....2046
43.5.3.1 Up counting....2046
43.5.3.2 Up-down counting....2047
43.5.3.3 Free running counter....2049
43.5.3.4 Counter reset....2049
43.5.3.5 Counter events....2049
43.5.4 Channel Modes....2050
43.5.5 Input Capture Mode....2052
43.5.5.1 Filter for Input Capture Mode....2053
43.5.5.2 FTM Counter Reset in Input Capture Mode....2055
43.5.6 Output Compare mode....2055
43.5.7 Edge-Aligned PWM (EPWM) mode....2056
43.5.8 Center-Aligned PWM (CPWM) mode....2057
43.5.9 Combine mode....2059
43.5.9.1 Asymmetrical PWM....2065

43.5.10 Modified Combine PWM Mode ... 2065
43.5.10.1 Synchronization ... 2067
43.5.11 Complementary Mode ... 2067
43.5.12 Registers updated from write buffers ... 2068
43.5.12.1 CNTIN register update ... 2068
43.5.12.2 MOD and HCR registers update ... 2069
43.5.12.3 CnV register update ... 2069
43.5.13 PWM synchronization ... 2070
43.5.13.1 Hardware trigger ... 2070
43.5.13.2 Software trigger ... 2071
43.5.13.3 Synchronization Points ... 2071
43.5.13.4 MOD register synchronization ... 2072
43.5.13.5 CNTIN register synchronization ... 2075
43.5.13.6 C(n)V and C(n+1)V register synchronization ... 2075
43.5.13.7 OUTMASK register synchronization ... 2075
43.5.13.8 INVCTRL register synchronization ... 2077
43.5.13.9 SWOCTRL register synchronization ... 2078
43.5.13.10 FTM counter synchronization ... 2079
43.5.14 Inverting ... 2081
43.5.15 Software Output Control Mode ... 2083
43.5.16 Deadtime insertion ... 2084
43.5.16.1 Deadtime insertion corner cases ... 2085
43.5.17 Output mask ... 2086
43.5.18 Polarity Control ... 2087
43.5.19 Initialization ... 2087
43.5.20 Features Priority ... 2088
43.5.21 External Trigger ... 2088
43.5.22 Initialization Trigger ... 2089
43.5.23 Capture Test Mode ... 2091
43.5.24 DMA ... 2092
43.5.25 Dual Edge Capture Mode ... 2092
43.5.25.1 One-Shot Capture mode ... 2093
43.5.25.2 Continuous Capture mode ... 2093
43.5.25.3 Pulse width measurement ... 2094
43.5.25.4 Period measurement ... 2095
43.5.25.5 Read coherency mechanism ... 2097
43.5.26 Quadrature Decoder Mode ... 2098
43.5.26.1 Quadrature Decoder boundary conditions ... 2101
43.5.27 Debug mode ... 2102
43.5.28 Reload Points ... 2102
43.5.28.1 Reload Opportunities ... 2103
43.5.28.2 Frequency of Reload Opportunities ... 2104
43.5.28.3 Update of the Registers ... 2104
43.5.29 Global Load ... 2105
43.5.30 Global time base (GTB) ... 2105
43.5.30.1 Enabling the global time base (GTB) ... 2106
43.5.31 Channel trigger output ... 2106
43.5.32 External Control of Channels Output ... 2107
43.5.33 Dithering ... 2107
43.5.33.1 PWM Period Dithering ... 2107
43.5.33.1.1 Up Counting ... 2108
43.5.33.1.2 Up-Down Counting ... 2109
43.5.33.2 PWM Edge Dithering ... 2110
43.5.33.2.1 EPWM Mode ... 2110
43.5.33.2.2 CPWM Mode ... 2111

43.5.33.2.3 Combine Mode....2112
43.5.33.2.4 Modified Combine PWM Mode....2114
43.6 Reset Overview....2115
43.7 FTM Interrupts....2116
43.7.1 Timer Overflow Interrupt....2116
43.7.2 Reload Point Interrupt....2116
43.7.3 Channel (n) Interrupt....2117
43.8 Initialization Procedure....2117

Chapter 44 Real Time Clock (RTC)....2119
44.1 Chip-specific RTC information....2119
44.1.1 RTC clocking....2119
44.2 Introduction....2119
44.2.1 Features....2119
44.2.2 Block diagram....2119
44.2.3 Modes of operation....2120
44.2.3.1 Functional mode....2120
44.2.3.2 Debug mode....2121
44.3 RTC register descriptions....2121
44.3.1 RTC memory map....2121
44.3.2 RTC Supervisor control register (RTCSUPV)....2121
44.3.3 RTC Control register (RTCC)....2122
44.3.4 RTC Status register (RTCS)....2124
44.3.5 RTC Counter register (RTCCNT)....2126
44.3.6 API Compare value register (APIVAL)....2127
44.3.7 RTC Compare value register (RTCVAL)....2128
44.4 Functional description....2128
44.4.1 RTC....2128
44.4.2 API functional description....2129
44.5 Recommended programming flow....2129
44.6 Glossary....2130

Chapter 45 Inter-Integrated Circuit (I2C)....2131
45.1 Overview....2131
45.2 Introduction to I^2C....2131
45.2.1 Definition: I^2C module....2131
45.2.2 Module block diagram....2131
45.2.3 Features....2132
45.2.4 Modes of operation....2132
45.2.5 Definition: I^2C conditions....2133
45.3 External signal descriptions....2133
45.3.1 Signal overview....2133
45.3.2 Detailed external signal descriptions....2134
45.4 I2C register descriptions....2134
45.4.1 I2C memory map....2134
45.4.2 I2C Bus Address Register (IBAD)....2135
45.4.3 I2C Bus Frequency Divider Register (IBFD)....2135
45.4.4 I2C Bus Control Register (IBCR)....2136
45.4.5 I2C Bus Status Register (IBSR)....2138
45.4.6 I2C Bus Data I/O Register (IBDR)....2141
45.4.7 I2C Bus Interrupt Config Register (IBIC)....2142
45.4.8 I2C Bus Debug Register (IBDBG)....2142
45.5 Functional description....2144

45.5.1 Notes about module operation....2144
45.5.2 Transactions....2144
45.5.2.1 Protocol overview....2144
45.5.2.2 Transaction protocol definitions....2145
45.5.2.3 I2C calling address requirements....2145
45.5.2.4 High-level protocol steps....2145
45.5.2.5 START condition....2146
45.5.2.6 Slave address transmission....2146
45.5.2.7 Data transmission....2146
45.5.2.8 STOP condition....2147
45.5.2.9 Repeated START condition....2147
45.5.3 Arbitration procedure....2147
45.5.4 Clock behavior....2147
45.5.4.1 Clock synchronization....2147
45.5.4.2 Clock stretching....2148
45.5.4.3 Handshaking....2148
45.5.4.4 Clock rate and IBFD settings....2148
45.5.4.4.1 Timing definitions....2148
45.5.4.4.2 Divider and hold values....2149
45.5.5 Interrupts....2163
45.5.5.1 Interrupt vector....2163
45.5.5.2 Interrupt description....2164
45.5.6 DEBUG mode....2164
45.5.7 DMA interface....2166
45.6 Initialization/application information....2166
45.6.1 Recommended interrupt service flow....2166
45.6.2 General programming guidelines (for both master and slave mode)....2167
45.6.2.1 Initializing the I2C module....2167
45.6.2.2 Software response after a transfer....2168
45.6.3 Programming guidelines specific to Master mode....2168
45.6.3.1 Generating START....2168
45.6.3.2 Transmit/receive sequence....2169
45.6.3.3 Generating STOP....2171
45.6.3.4 Generating repeated START....2172
45.6.3.5 Loss of arbitration....2172
45.6.4 Programming guidelines specific to Slave mode....2172
45.6.5 DMA application information....2172
45.6.5.1 DMA mode, master transmit....2173
45.6.5.2 DMA mode, master reception....2173
45.6.5.3 Exiting DMA mode and system requirement considerations....2174

Chapter 46 Gigabit Ethernet Media Access Controller (GMAC)....2177
46.1 Introduction....2177
46.2 Features....2178
46.3 Architecture....2179
46.3.1 Interfaces....2181
46.3.2 DMA controller....2181
46.3.3 MAC Transaction Layer (MTL)....2181
46.3.4 Media Access Controller (MAC)....2181
46.3.5 Interrupts....2181
46.4 Transmit and receive FIFOs....2182
46.5 Flow control....2182
46.6 Double VLAN tagging....2182
46.7 Descriptors....2182

46.7.1 Descriptor structure....2182
46.7.2 Transmit descriptor.... 2183
46.7.3 Receive descriptor.... 2183
46.8 Functional description....2183

Chapter 47 Packet Forwarding Engine (PFE)....2185
47.1 Introduction....2185
47.2 Features.... 2185
47.3 Initialization....2186
47.4 Functional description....2186
47.5 Packet parsing....2186
47.6 Layer 2 parsing....2186
47.7 Bridge classifier.... 2186
47.8 Layer 3 parsing....2187
47.9 Route classifier....2187
47.10 Queuing block....2187
47.11 Traffic manager unit (TMU).... 2187
47.12 Buffer management....2188
47.13 Host Interface (HIF)....2188
47.14 PHY interface options....2188
47.15 PFE configuration....2188
47.16 Software components....2188

Chapter 48 LINFlexD.... 2190
48.1 Chip-specific LINFlexD information.... 2190
48.1.1 Number of LINFlexD filters and channels.... 2190
48.1.2 LIN_CLK and bus clock relationship.... 2190
48.2 Introduction....2190
48.2.1 Block diagram.... 2190
48.3 Main features....2191
48.3.1 LIN mode features....2192
48.3.2 UART mode features.... 2192
48.4 Functional description....2193
48.4.1 LIN protocol....2193
48.4.1.1 Frames....2193
48.4.1.2 Data field....2193
48.4.1.3 Break....2193
48.4.1.4 Sync byte.... 2194
48.4.1.5 Identifier.... 2194
48.4.1.6 Checksum....2194
48.4.2 LINFlexD features.... 2194
48.4.2.1 Operating modes.... 2194
48.4.2.2 Test mode s.... 2195
48.4.2.3 Master mode....2196
48.4.2.3.1 Header transmission....2196
48.4.2.3.2 Data transmission.... 2196
48.4.2.3.3 Data reception....2197
48.4.2.3.4 Data discard....2197
48.4.2.4 Slave mode....2197
48.4.2.4.1 Data transmission.... 2197
48.4.2.4.2 Data reception....2197
48.4.2.5 Errors.... 2198
48.4.2.5.1 Header error....2198

48.4.2.5.2 SDEF ... 2198
48.4.2.5.3 IDPEF ... 2198
48.4.2.6 Bit error ... 2198
48.4.2.7 Framing error ... 2198
48.4.2.8 Checksum error ... 2199
48.4.2.9 Overrun error ... 2199
48.4.2.10 Timeout error ... 2199
48.4.2.10.1 Response timeout mechanism ... 2199
48.4.2.10.2 Header timeout mechanism ... 2200
48.4.2.11 Noise ... 2200
48.4.2.12 Start Detection and Delimiter Detection in receiver ... 2200
48.4.2.12.1 Start detection mechanism ... 2202
48.4.2.12.2 Break delimiter detection mechanism ... 2202
48.4.2.13 Baud rate generation ... 2204
48.4.2.14 Wakeup management ... 2205
48.4.3 Timer ... 2206
48.4.4 UART mode ... 2206
48.4.4.1 8-bit data frames ... 2206
48.4.4.2 9-bit frames ... 2206
48.4.4.3 Buffer in UART mode ... 2207
48.4.4.4 UART transmitter ... 2207
48.4.4.5 UART receiver ... 2208
48.4.5 DMA interface ... 2210
48.4.5.1 Main features ... 2210
48.4.5.2 Definitions ... 2210
48.4.5.3 Master node – TX mode ... 2211
48.4.5.4 Master node - RX mode ... 2214
48.4.5.5 UART – TX mode ... 2216
48.4.5.6 UART – RX mode ... 2218
48.4.5.7 Use cases and limitations ... 2220
48.5 LINFlexD register descriptions ... 2220
48.5.1 LINFlexD memory map ... 2221
48.5.2 LIN Control Register 1 (LINCR1) ... 2222
48.5.3 LIN Interrupt Enable Register (LINIER) ... 2225
48.5.4 LIN Status Register (LINSR) ... 2228
48.5.5 LIN Error Status Register (LINESR) ... 2231
48.5.6 UART Mode Control Register (UARTCR) ... 2233
48.5.7 UART Mode Status Register (UARTSR) ... 2239
48.5.8 LIN Time-Out Control Status Register (LINTCSR) ... 2241
48.5.9 LIN Output Compare Register (LINOCR) ... 2243
48.5.10 LIN Time-Out Control Register (LINTOCR) ... 2243
48.5.11 LIN Fractional Baud Rate Register (LINFBRR) ... 2244
48.5.12 LIN Integer Baud Rate Register (LINIBRR) ... 2246
48.5.13 LIN Checksum Field Register (LINCFR) ... 2247
48.5.14 LIN Control Register 2 (LINCR2) ... 2248
48.5.15 Buffer Identifier Register (BIDR) ... 2250
48.5.16 Buffer Data Register Least Significant (BDRL) ... 2252
48.5.17 Buffer Data Register Most Significant (BDRM) ... 2252
48.5.18 Global Control Register (GCR) ... 2253
48.5.19 UART Preset Timeout Register (UARTPTO) ... 2255
48.5.20 UART Current Timeout Register (UARTCTO) ... 2256
48.5.21 DMA Tx Enable Register (DMATXE) ... 2257
48.5.22 DMA Rx Enable Register (DMARXE) ... 2258
48.6 Programming considerations ... 2259
48.6.1 Master node ... 2259

48.6.1.1 Transmitter....2260
48.6.1.2 Receiver....2260
48.6.1.3 Transmitter, bit error....2260
48.6.1.4 Receiver, checksum error....2261
48.6.2 Slave node....2261
48.6.2.1 Transmitter (no identifier filters)....2261
48.6.2.2 Receiver (no identifier filters)....2261
48.6.2.3 No filters, transmitter, bit error....2261
48.6.2.4 No filters, receiver, checksum error....2262
48.6.3 Timeout....2262
48.6.4 UART mode....2263
48.6.5 Interrupts....2263
48.6.6 LINFlexD clock tolerance....2264
48.6.6.1 Faster receiver tolerance....2264
48.6.6.2 Slower receiver tolerance....2265

Chapter 49 Serial Peripheral Interface (SPI)....2266
49.1 Chip-specific SPI information....2266
49.1.1 Module instance information....2266
49.2 Introduction....2266
49.2.1 Block diagram....2266
49.2.2 Features....2267
49.2.3 Interface configurations....2268
49.2.3.1 SPI configuration....2268
49.2.4 Modes of operation....2269
49.2.4.1 Master mode....2269
49.2.4.2 Slave mode....2269
49.2.4.3 Module Disable mode....2269
49.2.4.4 Debug mode....2269
49.3 Module signal descriptions....2269
49.3.1 PCS0/SS_b....2270
49.3.2 PCS1....2270
49.3.3 PCS2....2270
49.3.4 PCS3....2270
49.3.5 PCS4....2270
49.3.6 PCS5/PCSS_b....2270
49.3.7 PCS6–PCS7....2271
49.3.8 SCK....2271
49.3.9 SIN....2271
49.3.10 SOUT....2271
49.4 SPI register descriptions....2271
49.4.1 SPI memory map....2271
49.4.2 Module Configuration Register (MCR)....2272
49.4.3 Transfer Count Register (TCR)....2277
49.4.4 Clock and Transfer Attributes Register (in Master mode) (CTAR0 - CTAR5)....2278
49.4.5 Clock and Transfer Attributes Register (in Slave mode) (CTAR0_SLAVE)....2283
49.4.6 Status Register (SR)....2285
49.4.7 DMA/Interrupt Request Select and Enable Register (RSER)....2288
49.4.8 PUSH TX FIFO Register In Master Mode (PUSHR)....2291
49.4.9 PUSH TX FIFO Register In Slave Mode (PUSHR_SLAVE)....2294
49.4.10 POP RX FIFO Register (POPR)....2295
49.4.11 Transmit FIFO Registers (TXFR0 - TXFR4)....2296
49.4.12 Receive FIFO Registers (RXFR0 - RXFR4)....2297
49.4.13 Clock and Transfer Attributes Register Extended (CTARE0 - CTARE5)....2298

49.4.14 Status Register Extended (SREX)..... 2299
49.5 Functional description.....2300
49.5.1 Starting and stopping module transfers..... 2301
49.5.2 SPI configuration.....2301
49.5.2.1 Master mode.....2302
49.5.2.2 Slave mode.....2302
49.5.2.3 FIFO disable operation..... 2302
49.5.2.4 TX FIFO buffering mechanism.....2302
49.5.2.4.1 Filling the TX FIFO.....2302
49.5.2.4.2 Draining the TX FIFO.....2303
49.5.2.5 CMD FIFO buffering mechanism..... 2303
49.5.2.6 RX FIFO buffering mechanism..... 2303
49.5.2.6.1 Filling the RX FIFO..... 2304
49.5.2.6.2 Draining the RX FIFO..... 2304
49.5.3 Module baud rate and clock delay generation..... 2304
49.5.3.1 Baud rate generator.....2304
49.5.3.2 PCS to SCK delay (t_{CSC}).....2304
49.5.3.3 After SCK delay (t_{ASC})..... 2305
49.5.3.4 Delay after transfer (t_{DT})..... 2305
49.5.3.5 PCSS enable (PCSS_b)..... 2305
49.5.4 Transfer formats.....2306
49.5.4.1 Classic SPI transfer format (CPHA = 0).....2307
49.5.4.2 Classic SPI transfer format (CPHA = 1).....2307
49.5.4.3 Modified SPI transfer format (MTFE = 1, CPHA = 0).....2308
49.5.4.4 Modified SPI transfer format (MTFE = 1, CPHA = 1).....2310
49.5.4.5 Continuous selection format..... 2311
49.5.4.6 Fast continuous selection format.....2313
49.5.5 Continuous SCK.....2313
49.5.6 Slave mode operation constraints.....2315
49.5.7 Parity generation and check.....2315
49.5.7.1 Parity for SPI frames.....2315
49.5.8 Interrupts/DMA requests..... 2316
49.5.8.1 EOQ interrupt request.....2316
49.5.8.2 Transmit FIFO fill interrupt or DMA request.....2316
49.5.8.3 CMD FIFO fill interrupt or DMA request.....2317
49.5.8.4 Transmit FIFO invalid write interrupt request.....2317
49.5.8.5 Transfer complete interrupt request.....2317
49.5.8.6 Command transfer complete interrupt request..... 2317
49.5.8.7 Transmit FIFO underflow interrupt request.....2317
49.5.8.8 Receive FIFO drain interrupt or DMA request..... 2317
49.5.8.9 Receive FIFO overflow interrupt request..... 2318
49.5.8.10 SPI frame parity error interrupt request..... 2318
49.5.9 Power saving features.....2318
49.5.9.1 Module Disable mode.....2318
49.6 Initialization/application information.....2318
49.6.1 How to manage queues..... 2318
49.6.2 Switching Master and Slave modes..... 2319
49.6.3 Initializing module in Master/Slave modes..... 2319
49.6.4 Baud rate settings..... 2319
49.6.5 Delay settings.....2320
49.6.6 Calculation of FIFO pointer addresses.....2321
49.6.6.1 Address calculation for the first-in entry and last-in entry in the TX FIFO.....2322
49.6.6.2 Address calculation for the first-in entry and last-in entry in the CMD FIFO.....2322
49.6.6.3 Address calculation for the first-in entry and last-in entry in the RX FIFO..... 2322

Chapter 50 CAN (FlexCAN).....2323
50.1 Chip-specific FlexCAN information.....2323
50.1.1 Overview.....2323
50.1.2 Message buffers and data rate.....2323
50.1.3 Rx FIFO.....2323
50.1.4 Mode configuration.....2323
50.1.5 Booting with FlexCAN.....2323
50.1.6 High-resolution timestamp.....2323
50.2 Introduction.....2324
50.2.1 Overview.....2324
50.2.2 FlexCAN module features.....2325
50.2.3 Modes of operation.....2326
50.3 FlexCAN signal descriptions.....2327
50.3.1 CAN Rx.....2327
50.3.2 CAN Tx.....2327
50.4 Memory map/register definition.....2327
50.4.1 FlexCAN memory mapping.....2327
50.4.2 CAN register descriptions.....2329
50.4.2.1 CAN memory map.....2330
50.4.2.2 Module Configuration register (MCR).....2331
50.4.2.3 Control 1 register (CTRL1).....2337
50.4.2.4 Free Running Timer (TIMER).....2341
50.4.2.5 Rx Mailboxes Global Mask register (RXMGMASK).....2342
50.4.2.6 Rx 14 Mask register (RX14MASK).....2344
50.4.2.7 Rx 15 Mask register (RX15MASK).....2345
50.4.2.8 Error Counter (ECR).....2345
50.4.2.9 Error and Status 1 register (ESR1).....2347
50.4.2.10 Interrupt Masks 2 register (IMASK2).....2354
50.4.2.11 Interrupt Masks 1 register (IMASK1).....2355
50.4.2.12 Interrupt Flags 2 register (IFLAG2).....2356
50.4.2.13 Interrupt Flags 1 register (IFLAG1).....2357
50.4.2.14 Control 2 register (CTRL2).....2360
50.4.2.15 Error and Status 2 register (ESR2).....2366
50.4.2.16 CRC register (CRCR).....2367
50.4.2.17 Legacy Rx FIFO Global Mask register (RXFGMASK).....2368
50.4.2.18 Legacy Rx FIFO Information register (RXFIR).....2370
50.4.2.19 CAN Bit Timing register (CBT).....2371
50.4.2.20 Interrupt Masks 4 register (IMASK4).....2373
50.4.2.21 Interrupt Masks 3 register (IMASK3).....2373
50.4.2.22 Interrupt Flags 4 register (IFLAG4).....2374
50.4.2.23 Interrupt Flags 3 register (IFLAG3).....2375
50.4.2.24 Rx Individual Mask registers (RXIMR0 - RXIMR127).....2376
50.4.2.25 Memory Error Control register (MECR).....2377
50.4.2.26 Error Injection Address register (ERRIAR).....2380
50.4.2.27 Error Injection Data Pattern register (ERRIDPR).....2381
50.4.2.28 Error Injection Parity Pattern register (ERRIPPR).....2382
50.4.2.29 Error Report Address register (RERRAR).....2383
50.4.2.30 Error Report Data register (RERRDR).....2385
50.4.2.31 Error Report Syndrome register (RERRSYNR).....2386
50.4.2.32 Error Status register (ERRSR).....2388
50.4.2.33 Enhanced CAN Bit Timing Prescalers (EPRS).....2390
50.4.2.34 Enhanced Nominal CAN Bit Timing (ENCBT).....2391
50.4.2.35 Enhanced Data Phase CAN bit Timing (EDCBT).....2393
50.4.2.36 Enhanced Transceiver Delay Compensation (ETDC).....2394

50.4.2.37 CAN FD Control register (FDCTRL)..... 2396
50.4.2.38 CAN FD Bit Timing register (FDCBT)..... 2400
50.4.2.39 CAN FD CRC register (FDCRC)..... 2402
50.4.2.40 Enhanced Rx FIFO Control register (ERFCR)..... 2404
50.4.2.41 Enhanced Rx FIFO Interrupt Enable register (ERFIER)..... 2411
50.4.2.42 Enhanced Rx FIFO Status register (ERFSR)..... 2413
50.4.2.43 High Resolution Time Stamp (HR_TIME_STAMP0 - HR_TIME_STAMP127)..... 2415
50.4.2.44 Enhanced Rx FIFO Filter Element (ERFFEL0 - ERFFEL127)..... 2416
50.4.3 Message buffer structure..... 2417
50.4.4 FlexCAN memory partition for CAN FD..... 2424
50.4.5 FlexCAN message buffer memory map..... 2426
50.4.6 Legacy Rx FIFO structure..... 2436
50.4.7 Enhanced Rx FIFO structure..... 2438
50.5 Functional description..... 2441
50.5.1 Transmit process..... 2441
50.5.2 Arbitration process..... 2442
50.5.2.1 Lowest-number mailbox first..... 2443
50.5.2.2 Highest-priority mailbox first..... 2443
50.5.2.2.1 Local priority disabled..... 2443
50.5.2.2.2 Local priority enabled..... 2443
50.5.2.3 Arbitration process (continued)..... 2444
50.5.3 Receive process..... 2445
50.5.4 Matching process..... 2446
50.5.5 Move process..... 2448
50.5.5.1 Move-in..... 2448
50.5.5.2 Move-out..... 2449
50.5.6 Data coherence..... 2449
50.5.6.1 Transmission abort mechanism..... 2449
50.5.6.2 Mailbox inactivation..... 2450
50.5.6.3 Mailbox lock mechanism..... 2450
50.5.7 Enhanced Rx FIFO..... 2451
50.5.7.1 Enhanced Rx FIFO matching process..... 2452
50.5.7.2 Enhanced Rx FIFO under DMA operation..... 2454
50.5.7.3 Enhanced Rx FIFO clear operation..... 2455
50.5.8 Legacy Rx FIFO..... 2455
50.5.8.1 Legacy Rx FIFO under DMA operation..... 2456
50.5.8.2 Clear Legacy FIFO operation..... 2456
50.5.9 CAN protocol related features..... 2457
50.5.9.1 CAN FD ISO compliance..... 2457
50.5.9.2 CAN FD frames..... 2457
50.5.9.3 Transceiver delay compensation..... 2460
50.5.9.4 Remote frames..... 2462
50.5.9.5 Overload frames..... 2463
50.5.9.6 Message buffer time stamp..... 2463
50.5.9.7 High resolution time stamp..... 2463
50.5.9.8 Protocol timing..... 2463
50.5.9.9 Arbitration and matching timing..... 2468
50.5.9.10 Tx arbitration start delay..... 2468
50.5.10 Clock domains and restrictions..... 2472
50.5.11 Modes of operation details..... 2476
50.5.11.1 Freeze mode..... 2476
50.5.11.2 Module Disable mode..... 2477
50.5.12 Interrupts..... 2477
50.5.13 Bus interface..... 2478
50.5.14 Detection and correction of memory errors..... 2478

50.5.14.1 Sources of the memory access....2479
50.5.14.2 Error indication....2479
50.5.14.3 Error reporting....2480
50.5.14.4 Response to errors....2480
50.5.14.5 Error injection....2481
50.6 Initialization/application information....2481
50.6.1 FlexCAN initialization sequence....2481
50.7 Glossary....2482

Chapter 51 Timestamp....2484
51.1 Timestamping and distribution....2484
51.2 Block diagram....2484
51.3 Timestamp synchronization controls....2484
51.4 Application information....2485
51.4.1 Enable timestamping....2485
51.4.2 Disable timestamping or changing the timestamp source....2485

Chapter 52 SerDes Subsystem....2486
52.1 Chip-specific SerDes information....2486
52.1.1 Mapping of SerDes signals to general purpose registers....2486
52.1.2 SerDes subsystem working modes....2486
52.1.3 Resets required for operation....2487
52.1.4 Register configuration for SerDes base spec....2487
52.1.5 Linking up mode0 and SerDes....2487
52.1.6 Unexpected behavior when enabling error injection in DMA Channel context memories of SerDes....2487
52.1.7 Converting nonsecure accesses to secure accesses....2487

Chapter 53 SerDes General Purpose Registers (SERDES_GPR)....2488
53.1 PCIe GPRs....2488
53.1.1 Introduction....2488
53.1.2 SERDES_GPR register descriptions....2488
53.1.2.1 SERDES_GPR memory map....2488
53.1.2.2 PCIE Configuration 0 (PCIE_Config_0)....2488
53.2 PCIe GPRs....2489
53.2.1 Introduction....2489
53.2.2 SERDES_GPR register descriptions....2489
53.2.2.1 SERDES_GPR memory map....2489
53.2.2.2 PCIE Configuration 0 (PCIE_Config_0)....2490

Chapter 54 Low Latency Communication Engine (LLCE)....2492
54.1 Introduction....2492
54.2 High-level features....2493
54.3 Reset and boot....2494
54.4 Modes of operation....2494
54.5 Subsystem control....2494
54.5.1 Reset control....2494
54.6 Host interface....2494
54.7 Clocking options....2495
54.8 Global timestamping....2496
54.9 Host interface events or interrupts....2496

54.10 FIFO interrupts..... 2496
54.11 LLCE generic FIFO.....2496
54.11.1 Features.....2497
54.11.2 Modes of operation..... 2497
54.11.3 LLCE FIFO register descriptions.....2498
54.11.3.1 LLCE FIFO registers memory map.....2498
54.11.3.2 FIFO Configuration Register (FCR)..... 2498
54.11.3.3 FIFO Status Register 0 (SR0).....2500
54.11.3.4 FIFO Status Register 1 (SR1).....2503
54.11.3.5 FIFO interrupt enable register (IER)..... 2506
54.11.3.6 FIFO Interrupt Line Register (ILR)..... 2508
54.11.3.7 FIFO push register 0 (PUSHR0).....2510
54.11.3.8 FIFO POP Register 0 (POPR0)..... 2510
54.11.3.9 FIFO Fixed MSB Register (FMR0).....2511
54.11.4 Functional description..... 2512
54.11.4.1 Basic push operation..... 2512
54.11.4.2 Mulit-word push operation.....2512
54.11.4.3 Basic pop operation..... 2512
54.11.4.4 Multi-word pop operation..... 2512
54.11.4.5 Pop operation when FIFO_FCR[POPEN] is 0..... 2512
54.11.4.6 FIFO flush..... 2512
54.11.4.7 FIFO watermark operation.....2513
54.11.4.8 Last FIFO entry overwrite operation..... 2513
54.11.4.9 Error scenarios.....2513
54.11.4.10 FIFO performance.....2513
54.11.5 Interrupts..... 2514
54.11.6 Programming.....2514
54.12 Core-to-Core module.....2515
54.12.1 Block diagram..... 2516
54.12.2 Features..... 2516
54.12.3 Core-to-core register descriptions.....2517
54.12.3.1 Core_to_core memory map.....2517
54.12.3.2 HOST interrupt from Core 0 Register (HINTC0R)..... 2518
54.12.3.3 HOST Interrupt from Core 1 Register (HINTC1R)..... 2520
54.12.3.4 HOST Interrupt from Core 2 Register (HINTC2R)..... 2523
54.12.3.5 HOST Interrupt from Core 3 Register (HINTC3R)..... 2526
54.12.3.6 Core 0 Interrupt from HOST (C0INTHR).....2528
54.12.3.7 Core 1 Interrupt From HOST Register (C1INTHR).....2531
54.12.3.8 Core 2 Interrupt from HOST Register (C2INTHR)..... 2534
54.12.3.9 Core 3 Interrupt from HOST Register (C3INTHR)..... 2536
54.12.3.10 HOST Interrupt from Core 0 Enable Register (HINTC0ER).....2539
54.12.3.11 HOST Interrupt from Core 1 Enable Register (HINTC1ER).....2541
54.12.3.12 HOST Interrupt from Core 2 Enable Register (HINTC2ER).....2544
54.12.3.13 HOST Interrupt from Core 3 Enable Register (HINTC3ER).....2546
54.12.3.14 Core 0 Interrupt from Host Enable Register (C0INTHER)..... 2548
54.12.3.15 Core 1 Interrupt from HOST Enable Register (C1INTHER).....2551
54.12.3.16 Core 2 Interrupt from HOST Enable Register (C2INTHER).....2553
54.12.3.17 Core 3 Interrupt from HOST Enable Register (C3INTHER).....2555
54.12.4 Functional description..... 2557
54.13 Interrupts and status.....2560
54.13.1 Interrupt concentrator (IC).....2560
54.13.2 Interrupt Routing Configuration Module (IRCM).....2560
54.14 External signal description.....2564
54.15 LLCE register descriptions..... 2565
54.15.1 System control memory map..... 2565

54.15.2 System Reset Control (LLCE_SYSRSTR)..... 2565

Chapter 55 Universal Serial Bus On-The-Go Controller (USBOTG)..... 2567
55.1 Chip-specific USB information..... 2567
55.1.1 Chip-specific USBOTG information..... 2567
55.1.2 SBUSCFG[AHBBRST] field reset value..... 2567
55.2 Introduction..... 2567
55.2.1 Overview..... 2568
55.2.2 Features..... 2568
55.2.3 Modes of operation..... 2569
55.3 External signal description..... 2569
55.4 USBOTG register descriptions..... 2570
55.4.1 Core register descriptions..... 2570
55.4.1.1 UOTG memory map..... 2570
55.4.1.2 Identification (ID)..... 2572
55.4.1.3 General Hardware Parameters (HWGENERAL)..... 2573
55.4.1.4 Host Hardware Parameters (HWHOST)..... 2574
55.4.1.5 Device Hardware Parameters (HWDEVICE)..... 2575
55.4.1.6 TX Buffer Hardware Parameters (HWTXBUF)..... 2576
55.4.1.7 RX Buffer Hardware Parameters (HWRXBUF)..... 2577
55.4.1.8 General Purpose Timer Load (GPTIMER0LD - GPTIMER1LD)..... 2578
55.4.1.9 General Purpose Timer Controller (GPTIMER0CTRL - GPTIMER1CTRL)..... 2579
55.4.1.10 System Bus Configuration (SBUSCFG)..... 2580
55.4.1.11 Capability Register Length (CAPLENGTH)..... 2582
55.4.1.12 Host Controller Interface Version (HCIVERSION)..... 2582
55.4.1.13 Host Controller Structural Parameters (HCSPARAMS)..... 2583
55.4.1.14 Host Controller Capability Parameters (HCCPARAMS)..... 2585
55.4.1.15 Device Controller Interface Version (DCIVERSION)..... 2586
55.4.1.16 Device Controller Capability Parameters (DCCPARAMS)..... 2587
55.4.1.17 USB Command (USBCMD)..... 2588
55.4.1.18 USB Status (USBSTS)..... 2592
55.4.1.19 Interrupt Enable (USBINTR)..... 2596
55.4.1.20 USB Frame Index (FRINDEX)..... 2598
55.4.1.21 Frame List Base Address or Device Address (PERIODICLISTBASE_or_DEVICEADDR)..... 2599
55.4.1.22 Next Asynchronous Address or Endpoint List Address (ASYNCLISTADDR_or_ENDPOINTLISTADDR)..... 2601
55.4.1.23 Programmable Burst Size (BURSTSIZE)..... 2602
55.4.1.24 TX FIFO Fill Tuning (TXFILLTUNING)..... 2603
55.4.1.25 IC_USB Enable and Voltage Negotiation (IC_USB)..... 2605
55.4.1.26 ULPI Register Access (ULPI_VIEWPORT)..... 2606
55.4.1.27 Endpoint NAK (ENDPTNAK)..... 2608
55.4.1.28 Endpoint NAK Enable (ENDPTNAKEN)..... 2609
55.4.1.29 Configure Flag (CONFIGFLAG)..... 2610
55.4.1.30 Port Status and Control (PORTSC1)..... 2611
55.4.1.31 OTG Status Control (OTGSC)..... 2619
55.4.1.32 USB Device Mode (USBMODE)..... 2622
55.4.1.33 Endpoint Setup Status (ENDPTSETUPSTAT)..... 2624
55.4.1.34 Endpoint Initialization (ENDPTPRIME)..... 2625
55.4.1.35 Endpoint De-Initialize (ENDPTFLUSH)..... 2626
55.4.1.36 Endpoint Status (ENDPTSTAT)..... 2627
55.4.1.37 Endpoint Complete (ENDPTCOMPLETE)..... 2629
55.4.1.38 Endpoint Control 0 (ENDPTCTRL0)..... 2630
55.4.1.39 Endpoint Control (ENDPTCTRL1 - ENDPTCTRL5)..... 2632
55.4.2 Non-core register descriptions..... 2635

55.4.2.1 UOTGNC memory map ... 2635
55.4.2.2 Control register (CR) ... 2635
55.5 Functional description ... 2637
55.5.1 USB dual role device/host controller ... 2637
55.5.1.1 Host mode ... 2638
55.5.1.2 Peripheral (Device) mode ... 2638
55.5.2 System interface ... 2638
55.5.3 DMA engine ... 2638
55.5.4 FIFO RAM controller ... 2638
55.5.5 USB power control block ... 2638
55.5.5.1 Entering low-power Suspend mode ... 2638
55.6 Wake-up events ... 2638
55.6.1 Host mode events ... 2639
55.6.2 Device mode events ... 2639
55.7 Interrupts ... 2639
55.7.1 USB core interrupts ... 2639
55.7.2 USB wake-up interrupts ... 2639
55.8 Data structures ... 2639
55.8.1 Device data structures ... 2639
55.8.1.1 Endpoint dQH ... 2640
55.8.1.1.1 Endpoint capabilities and characteristics (offset = 0h) ... 2641
55.8.1.1.2 Transfer overlay (offset = 8h-20h) ... 2642
55.8.1.1.3 Current dTD Pointer (offset = 0h) ... 2642
55.8.1.1.4 Setup buffer (offset = 28h-2Ch) ... 2643
55.8.1.2 Endpoint dTD ... 2643
55.8.1.2.1 Next dTD pointer (offset = 0h) ... 2644
55.8.1.2.2 dTD token (offset = 4h) ... 2644
55.8.1.2.3 dTD buffer page pointer list (offset = 8h-18h) ... 2646
55.8.2 Host data structures ... 2646
55.8.2.1 Periodic frame list ... 2647
55.8.2.2 Asynchronous list QH pointer ... 2648
55.8.2.3 iTD ... 2649
55.8.2.3.1 Next link pointer ... 2650
55.8.2.3.2 iTD transaction status and control list ... 2651
55.8.2.3.3 iTD buffer page pointer list (plus) ... 2652
55.8.2.4 siTD ... 2654
55.8.2.4.1 Next link pointer ... 2654
55.8.2.4.2 siTD endpoint capabilities/characteristics ... 2655
55.8.2.4.3 siTD transfer state ... 2656
55.8.2.4.4 siTD buffer pointer list (plus) ... 2657
55.8.2.4.5 siTD back link pointer ... 2658
55.8.2.5 Queue element transfer descriptor (qTD) ... 2659
55.8.2.5.1 Next qTD pointer ... 2660
55.8.2.5.2 Alternate next qTD pointer ... 2660
55.8.2.5.3 qTD token ... 2661
55.8.2.5.4 qTD buffer page pointer list ... 2664
55.8.2.6 QH ... 2665
55.8.2.6.1 QHLP ... 2666
55.8.2.6.2 QH endpoint capabilities/characteristics ... 2666
55.8.2.6.3 Transfer overlay-QH ... 2668
55.8.2.7 Periodic FSTN ... 2669
55.8.2.7.1 FSTN normal path pointer ... 2670
55.8.2.7.2 FSTN back path link pointer ... 2671
55.9 Device mode operation ... 2671
55.9.1 Port state and control ... 2671

55.9.2 Bus reset....2672
55.9.3 Suspend/resume....2673
55.9.3.1 Suspend operational model....2673
55.9.3.2 Resume....2673
55.9.4 Managing endpoints....2674
55.9.4.1 Endpoint initialization....2674
55.9.4.2 Stalling....2675
55.9.4.3 Data toggle....2675
55.9.4.3.1 Data toggle reset....2675
55.9.4.3.2 Data toggle inhibit....2675
55.9.5 Packet transfers....2676
55.9.5.1 Priming transmit endpoints....2676
55.9.5.2 Priming receive endpoints....2676
55.9.5.3 Interrupt/bulk endpoint operation....2676
55.9.6 Control endpoint operation....2678
55.9.6.1 Setup phase....2678
55.9.6.2 Data phase....2678
55.9.6.3 Status phase....2679
55.9.6.4 Control endpoint bus response matrix....2679
55.9.7 Isochronous endpoint operation....2679
55.9.7.1 Isochronous pipe synchronization....2680
55.9.7.2 Isochronous endpoint bus response matrix....2681
55.9.8 Managing QHs....2681
55.9.8.1 QH initialization....2682
55.9.8.2 Setup transfers operation....2682
55.9.9 Managing transfers with transfer descriptors....2682
55.9.9.1 Software link pointers....2682
55.9.9.2 Building a TD....2683
55.9.9.3 Executing a TD....2683
55.9.9.4 Transfer completion....2684
55.9.9.5 Flushing/de-priming an endpoint....2684
55.9.9.6 Device error matrix....2684
55.9.10 Servicing interrupts....2685
55.9.10.1 High frequency interrupts....2685
55.9.10.2 Error interrupts....2686
55.10 Host mode operation....2686
55.10.1 Servicing interrupts....2686
55.10.1.1 High frequency interrupts....2686
55.10.1.2 Low frequency interrupts....2686
55.10.1.3 Error interrupts....2687
55.10.2 Managing isochronous transfers using iTDs....2687
55.10.2.1 Host controller operational model for iTDs....2687
55.10.2.2 Software operational model for iTDs....2689
55.10.2.2.1 Periodic scheduling threshold....2690
55.10.3 Managing control/bulk/interrupt transfers through QHs....2690
55.10.3.1 Fetch QH....2692
55.10.3.2 Advance queue....2692
55.10.3.3 Execute transaction....2693
55.10.3.3.1 Interrupt transfer pre-condition criteria....2693
55.10.3.3.2 Asynchronous transfer pre-operations and pre-condition criteria....2693
55.10.3.3.3 Transfer type independent pre-operations....2693
55.10.3.3.4 Halting a QH....2695
55.10.3.3.5 Asynchronous Schedule Park mode....2695
55.10.3.4 Write back qTD....2696
55.10.3.5 Follow QH horizontal pointer....2697

55.10.3.6 Buffer pointer list use for data streaming with qTDs..... 2697
55.10.3.7 Adding interrupt QHs to the periodic schedule..... 2698
55.10.3.8 Managing transfer complete interrupts from queue heads..... 2699
55.10.4 Split transactions..... 2699
55.10.4.1 Split transactions for asynchronous transfers..... 2699
55.10.4.1.1 Asynchronous—do start split..... 2700
55.10.4.1.2 Asynchronous—do complete split..... 2700
55.10.4.2 Split transaction interrupt..... 2701
55.10.4.2.1 Split transaction scheduling mechanisms for interrupt..... 2701
55.10.4.2.2 Host controller operational model for FSTNs..... 2703
55.10.4.2.3 Software operational model for FSTNs..... 2705
55.10.4.2.4 Tracking split transaction progress for interrupt transfers..... 2705
55.10.4.2.5 Split transaction execution state machine for interrupt..... 2706
55.10.4.2.6 Rebalancing the periodic schedule..... 2710
55.10.4.3 Split transaction isochronous..... 2711
55.10.4.3.1 Split transaction scheduling mechanisms for isochronous..... 2711
55.10.4.3.2 Tracking split transaction progress for isochronous transfers..... 2714
55.10.4.3.3 Split transaction execution state machine for isochronous..... 2715
55.10.4.3.4 Periodic isochronous - do start split..... 2716
55.10.4.3.5 Periodic isochronous - do complete split..... 2718
55.10.4.3.6 Complete-split for scheduling boundary cases 2a, 2b..... 2719
55.11 Deviations from the EHCI specifications..... 2720
55.11.1 Embedded transaction translator function..... 2720
55.11.1.1 Capability registers..... 2720
55.11.1.2 Operational registers..... 2721
55.11.1.3 Discovery..... 2721
55.11.1.4 Data structures..... 2721
55.11.1.5 Operational model..... 2722
55.11.1.5.1 Microframe pipeline..... 2722
55.11.1.5.2 Split-state machines..... 2722
55.11.1.5.3 Asynchronous transaction scheduling and buffer management..... 2723
55.11.1.5.4 Periodic transaction scheduling and buffer management..... 2723
55.11.1.5.5 Multiple transaction translators..... 2724
55.11.2 Non-zero fields in the register file..... 2724
55.11.3 SOF interrupt..... 2724
55.11.4 Embedded design..... 2724
55.11.4.1 Frame adjust register..... 2724
55.11.5 Miscellaneous variations from EHCI..... 2724
55.11.5.1 Programmable physical interface behavior..... 2724
55.11.5.2 Discovery..... 2724
55.11.5.2.1 Port reset..... 2724
55.11.5.2.2 Port speed detection..... 2725
55.11.5.3 Port Test mode..... 2725

Chapter 56 FlexRay Communication Controller (FlexRay)..... 2726
56.1 Introduction..... 2726
56.1.1 References..... 2726
56.1.2 Overview..... 2726
56.1.3 Features..... 2727
56.1.4 Modes of operation..... 2728
56.1.4.1 Disabled mode..... 2728
56.1.4.1.1 Leave Disabled mode..... 2728
56.1.4.2 Normal mode..... 2728
56.1.4.2.1 Enter Normal mode..... 2728

56.2 External signal description....2728
56.3 PE clocking....2729
56.4 Register descriptions....2729
56.4.1 Register reset....2730
56.4.2 Register write access....2730
56.4.2.1 Register write access restriction....2730
56.4.2.2 Register write access requirements....2731
56.4.2.3 Internal register access....2731
56.4.3 PCR*n* fields....2731
56.5 FlexRay register descriptions....2734
56.5.1 FlexRay memory map....2735
56.5.2 Module Version (MVR)....2772
56.5.3 Module Configuration (MCR)....2772
56.5.4 System Memory Base Address High (SYMBADHR)....2775
56.5.5 System Memory Base Address Low (SYMBADLR)....2776
56.5.6 Strobe Signal Control (STBSCR)....2777
56.5.7 MB Data Size (MBDSR)....2779
56.5.8 MB Segment Size and Utilization (MBSSUTR)....2780
56.5.9 PE DRAM Access (PEDRAR)....2782
56.5.10 PE DRAM Data (PEDRDR)....2783
56.5.11 Protocol Operation Control Register (POCR)....2783
56.5.12 Global Interrupt Flag And Enable (GIFER)....2786
56.5.13 Protocol Interrupt Flag 0 (PIFR0)....2789
56.5.14 Protocol Interrupt Flag 1 (PIFR1)....2792
56.5.15 Protocol Interrupt Enable 0 (PIER0)....2794
56.5.16 Protocol Interrupt Enable 1 (PIER1)....2797
56.5.17 CHI Error Flag (CHIERFR)....2798
56.5.18 MB Interrupt Vector (MBIVEC)....2801
56.5.19 Channel A Status Error Counter Register (CASERCR)....2802
56.5.20 Channel B Status Error Counter (CBSERCR)....2803
56.5.21 Protocol Status 0 (PSR0)....2804
56.5.22 Protocol Status 1 (PSR1)....2806
56.5.23 Protocol Status Register 2 (PSR2)....2808
56.5.24 Protocol Status 3 (PSR3)....2811
56.5.25 MT Counter (MTCTR)....2814
56.5.26 Cycle Counter (CYCTR)....2814
56.5.27 Slot Counter Channel A (SLTCTAR)....2815
56.5.28 Slot Counter Channel B (SLTCTBR)....2816
56.5.29 Rate Correction Value (RTCORVR)....2817
56.5.30 Offset Correction Value (OFCORVR)....2818
56.5.31 Combined Interrupt Flag (CIFR)....2818
56.5.32 System Memory Access Timeout (SYMATOR)....2820
56.5.33 Sync Frame Counter (SFCNTR)....2821
56.5.34 Sync Frame Table Offset (SFTOR)....2822
56.5.35 Sync Frame Table Configuration Control Status (SFTCCSR)....2823
56.5.36 Sync Frame ID Rejection Filter (SFIDRFR)....2825
56.5.37 Sync Frame ID Acceptance Filter Value (SFIDAFVR)....2825
56.5.38 Sync Frame ID Acceptance Filter Mask (SFIDAFMR)....2826
56.5.39 NMV n (NMVR0 - NMVR5)....2827
56.5.40 Network Management Vector Length Register (NMVLR)....2828
56.5.41 Timer Configuration And Control (TICCR)....2829
56.5.42 Timer 1 Cycle Set Register (TI1CYSR)....2831
56.5.43 Timer 1 MT Offset (TI1MTOR)....2832
56.5.44 Timer 2 Configuration 0 (Absolute Timer Configuration) (TI2CR0_ABS)....2832
56.5.45 Timer 2 Configuration 0 (Relative Timer Configuration) (TI2CR0_REL)....2833

56.5.46 Timer 2 Configuration 1 (Absolute Timer Configuration) (TI2CR1_ABS)....2834
56.5.47 Timer 2 Configuration 1 (Relative Timer Configuration) (TI2CR1_REL)....2835
56.5.48 Slot Status Selection (SSSR)....2835
56.5.49 Slot Status Counter Condition (SSCCR)....2837
56.5.50 Slot Status (SSR0 - SSR7)....2839
56.5.51 Slot Status Counter (SSCR0 - SSCR3)....2842
56.5.52 MTS A Configuration (MTSACFR)....2843
56.5.53 MTS B Configuration (MTSBCFR)....2844
56.5.54 Receive Shadow Buffer Index (RSBIR)....2845
56.5.55 Receive FIFO Watermark And Selection (RFWMSR)....2846
56.5.56 Receive FIFO Start Index (RFSIR)....2847
56.5.57 Receive FIFO Depth And Size (RFDSR)....2848
56.5.58 Receive FIFO A Read Index (RFARIR)....2849
56.5.59 Receive FIFO B Read Index (RFBRIR)....2850
56.5.60 Receive FIFO Message ID Acceptance Filter Value (RFMIDAFVR)....2850
56.5.61 Receive FIFO Message ID Acceptance Filter Mask (RFMIDAFMR)....2851
56.5.62 Receive FIFO Frame ID Rejection Filter Value (RFFIDRFVR)....2852
56.5.63 Receive FIFO Frame ID Rejection Filter Mask (RFFIDRFMR)....2852
56.5.64 Receive FIFO Range Filter Configuration (RFRFCFR)....2853
56.5.65 Receive FIFO Range Filter Control (RFRFCTR)....2854
56.5.66 Last Dynamic Transmit Slot Channel A (LDTXSLAR)....2856
56.5.67 Last Dynamic Transmit Slot Channel B (LDTXSLBR)....2857
56.5.68 Protocol Configuration 0 (PCR0)....2858
56.5.69 Protocol Configuration 1 (PCR1)....2858
56.5.70 Protocol Configuration 2 (PCR2)....2859
56.5.71 Protocol Configuration 3 (PCR3)....2860
56.5.72 Protocol Configuration 4 (PCR4)....2861
56.5.73 Protocol Configuration 5 (PCR5)....2861
56.5.74 Protocol Configuration 6 (PCR6)....2862
56.5.75 Protocol Configuration 7 (PCR7)....2863
56.5.76 Protocol Configuration 8 (PCR8)....2864
56.5.77 Protocol Configuration 9 (PCR9)....2864
56.5.78 Protocol Configuration 10 (PCR10)....2865
56.5.79 Protocol Configuration 11 (PCR11)....2866
56.5.80 Protocol Configuration 12 (PCR12)....2867
56.5.81 Protocol Configuration 13 (PCR13)....2867
56.5.82 Protocol Configuration 14 (PCR14)....2868
56.5.83 Protocol Configuration 15 (PCR15)....2869
56.5.84 Protocol Configuration 16 (PCR16)....2869
56.5.85 Protocol Configuration 17 (PCR17)....2870
56.5.86 Protocol Configuration 18 (PCR18)....2871
56.5.87 Protocol Configuration 19 (PCR19)....2871
56.5.88 Protocol Configuration 20 (PCR20)....2872
56.5.89 Protocol Configuration 21 (PCR21)....2873
56.5.90 Protocol Configuration 22 (PCR22)....2873
56.5.91 Protocol Configuration 23 (PCR23)....2874
56.5.92 Protocol Configuration 24 (PCR24)....2875
56.5.93 Protocol Configuration 25 (PCR25)....2875
56.5.94 Protocol Configuration 26 (PCR26)....2876
56.5.95 Protocol Configuration 27 (PCR27)....2877
56.5.96 Protocol Configuration 28 (PCR28)....2877
56.5.97 Protocol Configuration 29 (PCR29)....2878
56.5.98 Protocol Configuration 30 (PCR30)....2879
56.5.99 Stopwatch Count High (STPWHR)....2879
56.5.100 Stopwatch Count Low (STPWLR)....2880

56.5.101 Protocol Event Output Enable And Stopwatch Control (PEOER)....2881
56.5.102 Receive FIFO Start Data Offset (RFSDOR)....2882
56.5.103 Receive FIFO System Memory Base Address High (RFSYMBADHR)....2882
56.5.104 Receive FIFO System Memory Base Address Low (RFSYMBADLR)....2883
56.5.105 Receive FIFO Periodic Timer (RFPTR)....2884
56.5.106 Receive FIFO Fill Level and Pop Count (RFFLPCR)....2884
56.5.107 ECC Error Interrupt Flag And Enable (EEIFER)....2885
56.5.108 ECC Error Report And Injection Control (EERICR)....2888
56.5.109 ECC Error Report Address (EERAR)....2889
56.5.110 ECC Error Report Data (EERDR)....2891
56.5.111 ECC Error Report Code (EERCR)....2892
56.5.112 ECC Error Injection Address Register (EEIAR)....2893
56.5.113 ECC Error Injection Data (EEIDR)....2894
56.5.114 ECC Error Injection Code (EEICR)....2895
56.5.115 MB Configuration Control Status (MBCCSR0 - MBCCSR255)....2896
56.5.116 MB Cycle Counter Filter (MBCCFR0 - MBCCFR255)....2898
56.5.117 MB Frame ID (MBFIDR0 - MBFIDR255)....2900
56.5.118 MB Index (MBIDXR0 - MBIDXR255)....2901
56.5.119 MB Data Field Offset (MBDOR0 - MBDOR259)....2901
56.5.120 LRAM ECC Error Test Register (LEETR0 - LEETR5)....2902
56.6 Functional description....2903
56.6.1 MB concept....2903
56.6.2 Physical MB....2903
56.6.2.1 MB header field....2903
56.6.2.1.1 Frame header....2904
56.6.2.1.2 Slot status....2904
56.6.2.2 MB data field....2904
56.6.3 MB types....2904
56.6.3.1 Individual MBs....2904
56.6.3.1.1 Individual MB segments....2905
56.6.3.2 Receive shadow buffers....2906
56.6.3.3 Receive FIFO....2906
56.6.3.4 MB configuration and control data....2908
56.6.3.4.1 Individual MB configuration data....2908
56.6.3.5 Individual MB control data....2909
56.6.3.6 Receive shadow buffer configuration data....2909
56.6.3.7 Receive FIFO control and configuration data....2909
56.6.3.7.1 Receive FIFO configuration data....2909
56.6.3.7.2 Receive FIFO control data....2909
56.6.3.7.3 Receive FIFO status data....2909
56.6.4 FlexRay memory area layout....2909
56.6.4.1 FlexRay memory area layout when MCR[FAM] = 0....2910
56.6.4.2 FlexRay memory area layout when MCR[FAM] = 1....2910
56.6.4.3 MB header area when MCR[FAM] = 0....2911
56.6.4.4 MB header area with MCR[FAM] = 1....2911
56.6.4.5 FIFO MB header area with MCR[FAM] = 1....2912
56.6.4.6 MB data area....2912
56.6.4.7 Sync frame table area....2912
56.6.5 Physical MB description....2912
56.6.5.1 MB protection and data consistency....2912
56.6.5.2 MB header field description....2912
56.6.5.2.1 Frame header description....2912
56.6.5.2.2 Slot status description....2915
56.6.5.3 MB data field description....2921
56.6.5.3.1 MB data field read access....2922

56.6.5.3.2 MB data field write access....2922
56.6.6 Individual MB functional description....2922
56.6.6.1 Individual MB configuration....2923
56.6.6.1.1 Common configuration data....2923
56.6.6.1.2 Specific configuration data....2923
56.6.6.2 Transmit MBs....2924
56.6.6.2.1 Access regions....2924
56.6.6.2.2 MB states....2925
56.6.6.2.3 MB transitions....2926
56.6.6.2.4 Transmit message setup....2928
56.6.6.2.5 Message transmission....2929
56.6.6.2.6 Null frame transmission....2930
56.6.6.2.7 MB status update....2931
56.6.6.3 Receive MBs....2932
56.6.6.3.1 MB transitions....2934
56.6.6.3.2 Message reception....2935
56.6.6.3.3 MB update....2936
56.6.6.3.4 Received message access....2937
56.6.6.3.5 Receive shadow buffer concept....2938
56.6.7 Individual MB search....2938
56.6.7.1 MB cycle counter filtering....2939
56.6.7.2 MB channel assignment consistency....2939
56.6.7.3 Node-related slot multiplexing....2940
56.6.7.4 MB search error....2940
56.6.7.4.1 MB search start while running....2940
56.6.7.4.2 Illegal MB index found....2940
56.6.8 Individual Message Buffer Reconfiguration....2940
56.6.8.1 Reconfiguration Schemes....2940
56.6.8.1.1 Basic Type Not Changed (RC1)....2941
56.6.8.1.2 Buffer Type Not Changed (RC2)....2941
56.6.9 Receive FIFOs....2941
56.6.9.1 Overview....2941
56.6.9.2 FIFO Configuration....2941
56.6.9.2.1 Single System Memory Base Address Mode....2941
56.6.9.2.2 Dual System Memory Base Address Mode....2941
56.6.9.3 FIFO Periodic Timer....2942
56.6.9.4 FIFO Reception....2942
56.6.9.5 FIFO Almost-Full Interrupt Generation....2942
56.6.9.6 FIFO Overflow Error Generation....2942
56.6.9.7 FIFO Message Access....2942
56.6.9.8 FIFO Update....2943
56.6.9.8.1 FIFO Interrupt Flag Update....2943
56.6.9.9 FIFO Filtering....2943
56.6.9.9.1 RX FIFO Frame ID Value-Mask Rejection Filter....2944
56.6.9.9.2 RX FIFO Frame ID Range Rejection Filter....2945
56.6.9.9.3 RX FIFO Frame ID Range Acceptance filter....2945
56.6.9.9.4 RX FIFO Message ID Acceptance Filter....2945
56.6.10 Channel Device Modes....2946
56.6.10.1 Dual Channel Device Mode....2946
56.6.10.2 Single Channel Device Mode....2946
56.6.11 External Clock Synchronization....2947
56.6.12 Sync Frame ID and Sync Frame Deviation Tables....2948
56.6.12.1 Sync Frame ID Table Content....2948
56.6.12.2 Sync Frame Deviation Table Content....2948
56.6.12.3 Sync Frame ID and Sync Frame Deviation Table Setup....2949

56.6.12.4 Sync Frame ID and Sync Frame Deviation Table Generation....2949
56.6.12.5 Sync Frame Table Access....2950
56.6.12.5.1 Sync Frame Table Locking and Unlocking....2950
56.6.13 MTS Generation....2951
56.6.14 Key Slot Transmission....2951
56.6.14.1 Key Slot Assignment....2951
56.6.14.2 Key Slot Transmission in POC:startup....2951
56.6.14.3 Key Slot Transmission in POC:normal active....2951
56.6.15 Sync Frame Filtering....2952
56.6.15.1 Sync Frame Acceptance Filtering....2952
56.6.15.2 Sync Frame Rejection Filtering....2952
56.6.16 Strobe Signal Support....2952
56.6.16.1 Strobe Signal Assignment....2953
56.6.16.2 Strobe Signal Timing....2953
56.6.17 Timer Support....2953
56.6.17.1 Absolute Timer T1....2953
56.6.17.2 Absolute / Relative Timer T2....2954
56.6.17.2.1 Absolute Timer T2....2954
56.6.17.2.2 Relative Timer T2....2954
56.6.18 Slot Status Monitoring....2954
56.6.18.1 Channel Status Error Counter Registers....2956
56.6.18.2 Protocol Status Registers....2956
56.6.18.3 Slot Status Registers....2956
56.6.18.4 Slot Status Counter Registers....2956
56.6.18.5 Message Buffer Slot Status Field....2957
56.6.19 System Bus Access....2958
56.6.19.1 System Bus Access Failure Types....2958
56.6.19.1.1 System Bus Illegal Address Access....2958
56.6.19.1.2 System Bus Access Timeout....2958
56.6.19.2 System Bus Access Failure Response....2958
56.6.19.2.1 Continue after System Bus Access Failure....2958
56.6.19.2.2 Freeze after System Bus Access Failure....2958
56.6.20 Interrupt Support....2959
56.6.20.1 Individual Interrupt Sources....2959
56.6.20.1.1 Message Buffer Interrupts....2959
56.6.20.1.2 FIFO Interrupts....2959
56.6.20.1.3 Wakeup Interrupt....2959
56.6.20.1.4 Protocol Interrupts....2959
56.6.20.1.5 CHI Interrupts....2959
56.6.20.2 Combined Interrupt Sources....2959
56.6.20.2.1 Receive Message Buffer Interrupt....2959
56.6.20.2.2 Transmit Message Buffer Interrupt....2959
56.6.20.2.3 Protocol Interrupt....2959
56.6.20.2.4 CHI Interrupt....2960
56.6.20.2.5 Module Interrupt....2960
56.6.21 Lower Bit Rate Support....2961
56.6.22 PE Data Memory PE DRAM....2962
56.6.22.1 PE DRAM Read Access....2962
56.6.22.2 PE DRAM Write Access....2963
56.6.22.3 PE DRAM Write Access Limitations....2963
56.6.23 CHI Lookup-Table Memory CHI LRAM....2963
56.6.23.1 CHI LRAM Read and Write Access....2964
56.6.24 Memory Content Error Detection....2964
56.6.24.1 Memory Error Types....2964
56.6.24.2 Memory Error Reporting....2965

56.6.24.2.1 PE DRAM Checkbits....2965
56.6.24.2.2 PE DRAM Syndrome....2966
56.6.24.2.3 CHI LRAM checkbits....2967
56.6.24.2.4 CHI LRAM Syndrome....2967
56.6.24.3 Memory Error Response....2967
56.6.24.3.1 CHI LRAM Error Response after CC Read....2968
56.6.24.3.2 CHI LRAM Error Response after Application Read....2968
56.6.24.3.3 PE DRAM Error Response after CC Read....2968
56.6.24.3.4 PE DRAM Error Response after Application Read in *POC:default config* state....2968
56.6.24.3.5 PE DRAM Error Response after Application Read out of *POC:default config*....2968
56.6.25 Memory Error Injection....2968
56.6.25.1 CHI LRAM Error Injection....2968
56.6.25.2 PE DRAM Error Injection....2969
56.6.26 Stopwatch Function....2970
56.7 Application Information....2971
56.7.1 Module Configuration....2971
56.7.1.1 Configure System Memory Access Time-Out Register (SYMATOR)....2971
56.7.1.1.1 System Bus Wait State Constraints....2972
56.7.1.2 Configure Data Field Offsets....2972
56.7.2 Initialization Sequence....2972
56.7.2.1 Module Initialization....2972
56.7.2.2 Protocol Initialization....2972
56.7.2.3 CHI LRAM Initialization....2973
56.7.2.4 PE DRAM Initialization....2973
56.7.3 Memory Fault Injection out of POC:default config....2973
56.7.3.1 CHI LRAM Error Injection out of POC:default config....2973
56.7.3.2 PE DRAM Error Injection out of POC:default config....2973
56.7.4 Shut Down Sequence....2973
56.7.5 Number of Usable Message Buffers....2974
56.7.6 Protocol Control Command Execution....2975
56.7.7 Message Buffer Search On Simple Message Buffer Configuration....2975
56.7.7.1 Simple Message Buffer Configuration....2976
56.7.7.2 Behavior in static segment....2977
56.7.7.3 Behavior in dynamic segment....2977
56.7.7.3.1 Transmit Data Not Available....2977
56.7.7.3.2 Transmit Data Available....2977
56.8 Glossary....2978

Chapter 57 Cross-Triggering Unit (CTU)....2980
57.1 Chip-specific CTU information....2980
57.1.1 CTU connections....2980
57.2 Introduction....2980
57.3 Block diagram....2981
57.4 CTU overview....2981
57.4.1 Modes of operation....2982
57.5 Signal description....2982
57.6 CTU register descriptions....2982
57.6.1 CTU memory map....2982
57.6.2 Trigger Generator Subunit Input Selection Register (TGSISR)....2984
57.6.3 Trigger Generator Subunit Control Register (TGSCR)....2987
57.6.4 Trigger Compare Register (T0CR - T7CR)....2988
57.6.5 TGS Counter Compare Register (TGSCCR)....2988
57.6.6 TGS Counter Reload Register (TGSCRR)....2989
57.6.7 Commands List Control Register 1 (CLCR1)....2989

57.6.8 Commands List Control Register 2 (CLCR2)....2991
57.6.9 Trigger Handler Control Register 1 (THCR1)....2992
57.6.10 Trigger Handler Control Register 2 (THCR2)....2995
57.6.11 Commands List Register A for ADC single-conversion mode commands (CLR_A_1 - CLR_A_24)....2998
57.6.12 Command List Register B for ADC dual-conversion mode commands (CLR_B_1 - CLR_B_24)....3000
57.6.13 Command List Register C for self-test commands (CLR_C_1 - CLR_C_24)....3001
57.6.14 FIFO DMA Control Register (FDCR)....3003
57.6.15 FIFO Control Register (FCR)....3005
57.6.16 FIFO Threshold Register (FTH)....3007
57.6.17 FIFO Status Register (FST)....3008
57.6.18 FIFO Right Aligned Data Register (FR0 - FR3)....3011
57.6.19 FIFO Signed Left Aligned Data Register (FL0 - FL3)....3012
57.6.20 Error Flag Register (EFR)....3013
57.6.21 Interrupt Flag Register (IFR)....3015
57.6.22 Interrupt/DMA Register (IR)....3017
57.6.23 Control ON Time Register (COTR)....3019
57.6.24 Control Register (CR)....3019
57.6.25 Digital Filter Register (DFR)....3021
57.6.26 Expected Value A Register (EXPAR)....3022
57.6.27 Expected Value B Register (EXPBR)....3023
57.6.28 Counter Range Register (CNTRNGR)....3024
57.6.29 List Control/Status Register (LISTCSR)....3025
57.7 Functional description....3026
57.7.1 Trigger Generator Subunit (TGS)....3026
57.7.2 TGS in triggered mode....3026
57.7.3 TGS in sequential mode....3028
57.7.4 TGS counter....3029
57.7.5 Scheduler subunit (SU)....3031
57.7.6 ADC commands list....3032
57.7.7 ADC command list format....3032
57.7.8 ADC command list operation modes....3034
57.7.9 ADC results FIFOs....3034
57.7.10 Reload mechanism....3035
57.8 Interrupts and DMA requests....3036
57.8.1 DMA support....3037
57.8.2 CTU faults and errors....3037
57.8.2.1 CTU parallel list errors....3037
57.8.3 CTU interrupt/DMA requests....3038
57.9 Self-test mode....3038
57.9.1 Self-test execution errors....3040

Chapter 58 Analog-to-Digital Converter (SAR_ADC)....3041
58.1 Chip-specific SAR_ADC information....3041
58.1.1 SAR_ADC instances....3041
58.1.2 SAR_ADC resolution....3041
58.1.3 Analog watchdogs....3041
58.1.4 Normal and injection trigger....3041
58.1.5 SAR_ADC channel assignments....3041
58.2 Introduction....3042
58.3 Overview....3042
58.3.1 Interfaces....3043
58.3.2 Submodules....3043

58.3.3 Clock and reset.... 3043
58.3.4 Special channels.... 3044
58.4 Features.... 3044
58.5 Functional description.... 3045
58.5.1 Conversion.... 3045
58.5.2 Normal Conversion mode.... 3045
58.5.2.1 Starting conversions in Normal Conversion mode.... 3046
58.5.2.2 Operation modes in Normal Conversion mode.... 3046
58.5.3 Injected Conversion mode.... 3047
58.5.3.1 Starting conversions in Injected Conversion mode.... 3048
58.5.4 Abort of conversion.... 3048
58.5.5 CTU.... 3049
58.5.6 ADC clock prescaler and sample time settings.... 3049
58.5.7 Presampling.... 3049
58.5.7.1 Presampling channel enable.... 3050
58.5.8 Programmable analog watchdog.... 3050
58.5.9 DMA functionality.... 3052
58.5.10 Interrupts.... 3052
58.5.11 Power-Down mode.... 3055
58.5.12 Low-power mode support.... 3056
58.5.13 External start feature.... 3056
58.5.14 Auto-Clock-Off mode.... 3056
58.5.15 Calibration.... 3057
58.5.16 Self-test.... 3058
58.5.16.1 CPU mode.... 3059
58.5.16.2 CTU mode.... 3061
58.5.16.3 Abort and abort chain for self-testing channel.... 3061
58.5.16.4 Self-test analog watchdog.... 3061
58.5.16.5 S1 algorithm: calculated voltage comparison method.... 3063
58.5.16.6 Watchdog timer.... 3063
58.5.16.6.1 Watchdog sequence checking.... 3064
58.5.16.7 Baud rate control for test channel.... 3065
58.5.16.7.1 Abort chain when baud rate is non-zero.... 3065
58.5.17 Conversion time.... 3065
58.5.18 Conversion data processing.... 3068
58.5.19 User defined offset and gain values.... 3068
58.5.19.1 Writing for 12-bit ADC.... 3068
58.5.19.1.1 User gain compensation (GAIN_USER).... 3068
58.5.19.1.2 User offset compensation (OFFSET_USER).... 3069
58.5.19.1.3 Using OFFSET_USER and GAIN_USER together.... 3069
58.6 Programming sequences.... 3069
58.6.1 Running calibration.... 3069
58.6.2 Running normal conversion.... 3070
58.6.3 Running injected conversion.... 3070
58.6.4 Running CTU triggered conversion.... 3071
58.6.5 Running self-test.... 3071
58.7 SAR_ADC register descriptions.... 3071
58.7.1 SAR_ADC memory map.... 3071
58.7.2 Main Configuration Register (MCR).... 3073
58.7.3 Main Status Register (MSR).... 3077
58.7.4 Interrupt Status Register (ISR).... 3080
58.7.5 Channel Pending Register (CEOCFR0).... 3081
58.7.6 Channel Pending Register 1 (CEOCFR1).... 3083
58.7.7 Interrupt Mask Register (IMR).... 3084
58.7.8 Channel Interrupt Mask Register 0 (CIMR0).... 3086

58.7.9 Channel Interrupt Mask Register 1 (CIMR1)....3087
58.7.10 Watchdog Threshold Interrupt Status Register (WTISR)....3089
58.7.11 Watchdog Threshold Interrupt Mask Register (WTIMR)....3092
58.7.12 DMAE Register (DMAE)....3095
58.7.13 DMA Register 0 (DMAR0)....3096
58.7.14 DMA Register 1 (DMAR1)....3098
58.7.15 Analog Watchdog Threshold Register 0 (THRHLR0)....3100
58.7.16 Analog Watchdog Threshold Register 1 (THRHLR1)....3101
58.7.17 Analog Watchdog Threshold Register 2 (THRHLR2)....3101
58.7.18 Analog Watchdog Threshold Register 2 (THRHLR3)....3102
58.7.19 Presampling Control Register (PSCR)....3103
58.7.20 Presampling Register 0 (PSR0)....3104
58.7.21 Presampling Register 1 (PSR1)....3106
58.7.22 Conversion Timing Register 0 (CTR0)....3107
58.7.23 Conversion Timing Register 1 (CTR1)....3108
58.7.24 Normal Conversion Mask Register (NCMR0)....3109
58.7.25 Normal Conversion Mask Register (NCMR1)....3110
58.7.26 Injected Conversion Mask Register (JCMR0)....3111
58.7.27 Injected Conversion Mask Register (JCMR1)....3113
58.7.28 User OFFSET and Gain Register (USROFSGN)....3114
58.7.29 Power Down Exit Delay Register (PDEDR)....3115
58.7.30 Precision Channel n Data Register (PCDR0 - PCDR7)....3116
58.7.31 Internal Channel n Data Register (ICDR32 - ICDR39)....3117
58.7.32 Analog Watchdog Threshold Register 4 (THRHLR4)....3118
58.7.33 Analog Watchdog Threshold Register 5 (THRHLR5)....3119
58.7.34 Analog Watchdog Threshold Register 6 (THRHLR6)....3120
58.7.35 Analog Watchdog Threshold Register 7 (THRHLR7)....3121
58.7.36 Channel Watchdog Select Register 0 (CWSELR0)....3121
58.7.37 Channel Watchdog Select Register 4 (CWSELR4)....3124
58.7.38 Channel Watchdog Enable Register (CWENR0)....3126
58.7.39 Channel Watchdog Enable Register (CWENR1)....3127
58.7.40 Analog Watchdog Out of Range Register (AWORR0)....3129
58.7.41 Analog Watchdog Out of Range Register (AWORR1)....3130
58.7.42 Self-Test Configuration Register 1 (STCR1)....3131
58.7.43 Self-Test Configuration Register 2 (STCR2)....3132
58.7.44 Self-Test Configuration Register 3 (STCR3)....3135
58.7.45 Self-Test Baud Rate Register (STBRR)....3137
58.7.46 Self-Test Status Register 1 (STSR1)....3138
58.7.47 Self-Test Status Register 2 (STSR2)....3141
58.7.48 Self-Test Status Register 3 (STSR3)....3142
58.7.49 Self-Test Status Register 4 (STSR4)....3143
58.7.50 Self-Test Data Register 1 (STDR1)....3144
58.7.51 Self-Test Data Register 2 (STDR2)....3145
58.7.52 Self-Test Analog Watchdog Register (STAW0R)....3146
58.7.53 Self-Test Analog Watchdog Register (STAW1AR)....3148
58.7.54 Self-Test Analog Watchdog Register 1B (STAW1BR)....3148
58.7.55 Self-Test Analog Watchdog Register 2 (STAW2R)....3149
58.7.56 Self-Test Analog Watchdog Register 3 (STAW3R)....3150
58.7.57 Self-Test Analog Watchdog Register 4 (STAW4R)....3151
58.7.58 Self-Test Analog Watchdog Register 5 (STAW5R)....3152
58.7.59 Calibration Status register (CALSTAT)....3154

Chapter 59 Thermal Monitoring Unit (TMU)....3158
59.1 Chip-specific TMU information....3158

59.1.1 Local temperature sensor placements....3158
59.1.2 Recommended TCMCFG register settings....3158
59.1.3 Calibration table....3158
59.2 Introduction....3159
59.2.1 Features....3160
59.2.2 Modes of operation....3160
59.3 TMU register descriptions....3160
59.3.1 TMU memory map....3160
59.3.2 Mode (TMR)....3161
59.3.3 Status (TSR)....3163
59.3.4 Monitor Site (TMSR)....3164
59.3.5 Monitor Temperature Measurement Interval (TMTMIR)....3165
59.3.6 Interrupt Enable (TIER)....3166
59.3.7 Interrupt Detect (TIDR)....3168
59.3.8 Interrupt Immediate Site Capture (TIISCR)....3170
59.3.9 Interrupt Average Site Capture (TIASCR)....3171
59.3.10 Interrupt Critical Site Capture (TICSCR)....3172
59.3.11 Monitor High Temperature Capture (TMHTCR)....3173
59.3.12 Monitor Low Temperature Capture (TMLTCR)....3174
59.3.13 Monitor Rising Temperature Rate Capture (TMRTRCR)....3175
59.3.14 Monitor Falling Temperature Rate Capture (TMFTRCR)....3176
59.3.15 Monitor High Temperature Immediate Threshold (TMHTITR)....3177
59.3.16 Monitor High Temperature Average Threshold (TMHTATR)....3178
59.3.17 Monitor High Temperature Average Critical Threshold (TMHTACTR)....3179
59.3.18 Monitor Low Temperature Immediate Threshold (TMLTITR)....3180
59.3.19 Monitor Low Temperature Average Threshold (TMLTATR)....3181
59.3.20 Monitor Low Temperature Average Critical Threshold (TMLTACTR)....3182
59.3.21 Monitor Rising Temperature Rate Critical Threshold (TMRTRCTR)....3183
59.3.22 Monitor Falling Temperature Rate Critical Threshold (TMFTRCTR)....3184
59.3.23 Temperature Configuration (TTCFGR)....3185
59.3.24 Sensor Configuration (TSCFGR)....3187
59.3.25 Report Immediate Temperature at Site (TRITSR0 - TRITSR2)....3187
59.3.26 Report Average Temperature at Site (TRATSR0 - TRATSR2)....3188
59.3.27 Central Module Configuration (TCMCFG)....3189
59.3.28 Temperature Range Control a (TTRCR0 - TTRCR15)....3192
59.4 Functional description....3193
59.4.1 Monitoring....3194
59.4.2 Reporting....3194
59.4.3 Using temperature thresholds....3195
59.5 Initialization information....3195

Chapter 60 Security....3197
60.1 Introduction....3197
60.1.1 Detection and response to parametric intrusion attempts....3198
60.1.2 Secure software updates....3198
60.1.3 Secure vehicle network and messaging....3198
60.1.4 Secure boot....3198
60.2 Security subsystem registers....3198
60.2.1 Security subsystem register descriptions....3198
60.2.1.1 SEC memory map....3198
60.2.1.2 External Debugger Status (EXT_DBGSTAT)....3199
60.2.1.3 HSE GPR 0 (HSE_GPR0)....3200
60.2.1.4 HSE GPR n (HSE_GPR1)....3201
60.2.1.5 HSE GPR n (HSE_GPR2)....3201

60.2.1.6 HSE GPR n (HSE_GPR3 - HSE_GPR11)....3202

Chapter 61 Messaging Unit (MU)....3204
61.1 Chip-specific MU information....3204
61.2 Overview....3204
61.2.1 Features....3205
61.3 Register Definition....3205
61.3.1 MUA register descriptions....3206
61.3.1.1 MUA memory map....3206
61.3.1.2 Version ID Register (VER)....3207
61.3.1.3 Parameter Register (PAR)....3208
61.3.1.4 Control Register (CR)....3209
61.3.1.5 Status Register (SR)....3210
61.3.1.6 Core Control Register 0 (CCR0)....3212
61.3.1.7 Core Sticky Status Register 0 (CSSR0)....3214
61.3.1.8 Flag Control Register (FCR)....3215
61.3.1.9 Flag Status Register (FSR)....3216
61.3.1.10 General Interrupt Enable Register (GIER)....3217
61.3.1.11 General Control Register (GCR)....3218
61.3.1.12 General Status Register (GSR)....3219
61.3.1.13 Transmit Control Register (TCR)....3220
61.3.1.14 Transmit Status Register (TSR)....3221
61.3.1.15 Receive Control Register (RCR)....3222
61.3.1.16 Receive Status Register (RSR)....3223
61.3.1.17 Transmit Register (TR0 - TR15)....3224
61.3.1.18 Receive Register (RR0 - RR15)....3224
61.3.2 MUB register descriptions....3225
61.3.2.1 MUB memory map....3225
61.3.2.2 Version ID Register (VER)....3226
61.3.2.3 Parameter Register (PAR)....3227
61.3.2.4 Control Register (CR)....3228
61.3.2.5 Status Register (SR)....3230
61.3.2.6 Core Control Register 0 (CCR0)....3232
61.3.2.7 Core Sticky Status Register 0 (CSSR0)....3233
61.3.2.8 Flag Control Register (FCR)....3235
61.3.2.9 Flag Status Register (FSR)....3236
61.3.2.10 General Interrupt Enable Register (GIER)....3237
61.3.2.11 General Control Register (GCR)....3238
61.3.2.12 General Status Register (GSR)....3239
61.3.2.13 Transmit Control Register (TCR)....3239
61.3.2.14 Transmit Status Register (TSR)....3240
61.3.2.15 Receive Control Register (RCR)....3241
61.3.2.16 Receive Status Register (RSR)....3242
61.3.2.17 Transmit Register (TR0 - TR15)....3243
61.3.2.18 Receive Register (RR0 - RR15)....3244
61.4 Functional Description....3245
61.4.1 Messaging Examples....3246
61.4.1.1 Messaging Protocols using Interrupts....3246
61.4.1.2 Messaging Protocols using Event Interrupts....3248
61.4.1.3 Exclusive Access to Shared Memory....3249
61.4.1.4 Packet Data Transfers....3250
61.4.1.5 Free the Processor from Deadlock....3251
61.4.2 Event Update Timing....3251
61.4.3 Interrupts....3252

61.4.4 Resets....3252

Chapter 62 On-Chip One Time Programmable (OCOTP) Controller....3254
62.1 Introduction....3254
62.2 Features....3254
62.3 Block diagram....3254
62.4 Operation....3255
62.4.1 Shadow register reload....3255
62.4.2 Fuse values....3255
62.4.3 Fuse protection and overrides....3256
62.4.4 eFuse and shadow register read....3256
62.4.5 eFuse and shadow register writes....3257
62.4.6 eFuse powerdown....3258
62.4.7 Write postamble....3258
62.5 Behavior during reset....3258
62.6 Power control of OTP memory....3258
62.7 CRC (Polynomial CRC-32/MPEG-2) test....3259
62.8 Fuse ECC....3259
62.9 Memory maps/register definitions....3260
62.9.1 OCOTP register descriptions....3260
62.9.1.1 OCOTP memory map....3260
62.9.1.2 System master's control (CTRL_SYS)....3261
62.9.1.3 System master's eFuse address for read or write operation (ADDR_SYS)....3263
62.9.1.4 System master's eFuse write data (WRDATA_SYS)....3264
62.9.1.5 System master's eFuse read data (RDATA_SYS)....3264
62.9.1.6 FBXC control (CTRL_FBX)....3265
62.9.1.7 eFuse region for CRC validation (CRC_RGN_SYS)....3266
62.9.1.8 Pre-calculated CRC eFuse address for comparision (CRC_ADDR_SYS)....3267
62.9.1.9 CRC calculated from the eFuses (CRC_VALUE_SYS)....3268
62.9.1.10 General purpose (GPR0 - GPR1)....3269
62.9.1.11 System master's access status (STATUS_SYS)....3269
62.9.1.12 OCOTP design version (VERSION)....3271
62.9.1.13 ECC status for single-bit ECC error (SEC0)....3272
62.9.1.14 ECC status for single-bit ECC error (SEC1)....3273
62.9.1.15 ECC status for single-bit ECC error (SEC2)....3273
62.9.1.16 ECC status for single-bit ECC error (SEC3)....3274
62.9.1.17 ECC status for double-bit ECC error (DED0)....3275
62.9.1.18 ECC status for double-bit ECC error (DED1)....3276
62.9.1.19 ECC status for double-bit ECC error (DED2)....3276
62.9.1.20 ECC status for double-bit ECC error (DED3)....3277
62.9.1.21 ECC error injector (ERR_INJCTR)....3278
62.9.1.22 Shadow (SHADOWS0 - SHADOWS127)....3279

Chapter 63 Fusebox Controller (FBXC)....3280
63.1 Introduction....3280
63.1.1 Overview....3280
63.1.2 Features....3280

Chapter 64 Safety....3281
64.1 Introduction....3281

Chapter 65 Error Reporting Module (ERM)..... 3282
65.1 Chip-specific ERM information..... 3282
65.1.1 Instances and numbers of channels..... 3282
65.1.2 Memory error event sources and captured information..... 3282
65.1.3 Over-reported single-bit ECC errors for QuadSPI accesses..... 3300
65.2 ERM_PFE, ERM_LLCE, and ERM_Standby_RAM instances..... 3300
65.2.1 Instances and numbers of channels..... 3300
65.2.2 Memory error event sources and captured information..... 3301
65.3 Introduction..... 3314
65.3.1 Overview..... 3314
65.3.2 Features..... 3314
65.4 ERM_CPU0 register descriptions..... 3314
65.4.1 ERM_CPU0 memory map..... 3315
65.4.2 ERM Configuration Register 0 (CR0)..... 3315
65.4.3 ERM Status Register 0 (SR0)..... 3319
65.4.4 ERM Memory a Error Address Register (EAR0 - EAR1)..... 3322
65.4.5 ERM Memory a Syndrome Register (SYN0 - SYN1)..... 3323
65.4.6 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT 5)..... 3324
65.5 ERM_CPU1 register descriptions..... 3325
65.5.1 ERM_CPU1 memory map..... 3325
65.5.2 ERM Configuration Register 0 (CR0)..... 3326
65.5.3 ERM Status Register 0 (SR0)..... 3330
65.5.4 ERM Memory a Error Address Register (EAR0 - EAR1)..... 3333
65.5.5 ERM Memory a Syndrome Register (SYN0 - SYN1)..... 3334
65.5.6 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT 5)..... 3335
65.6 ERM_CPU2 register descriptions..... 3336
65.6.1 ERM_CPU2 memory map..... 3336
65.6.2 ERM Configuration Register 0 (CR0)..... 3337
65.6.3 ERM Status Register 0 (SR0)..... 3341
65.6.4 ERM Memory a Error Address Register (EAR0 - EAR1)..... 3344
65.6.5 ERM Memory a Syndrome Register (SYN0 - SYN1)..... 3345
65.6.6 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT 5)..... 3346
65.7 ERM_PER register descriptions..... 3347
65.7.1 ERM_PER memory map..... 3347
65.7.2 ERM Configuration Register 0 (CR0)..... 3349
65.7.3 ERM Configuration Register 1 (CR1)..... 3353
65.7.4 ERM Status Register 0 (SR0)..... 3355
65.7.5 ERM Status Register 1 (SR1)..... 3360
65.7.6 ERM Memory a Error Address Register (EAR0 - EAR9)..... 3362
65.7.7 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT 9)..... 3363
65.7.8 ERM Memory a Syndrome Register (SYN8 - SYN9)..... 3364
65.8 ERM_PFE0 register descriptions..... 3365
65.8.1 ERM_PFE0 memory map..... 3365
65.8.2 ERM Configuration Register 0 (CR0)..... 3367
65.8.3 ERM Configuration Register 1 (CR1)..... 3371
65.8.4 ERM Status Register 0 (SR0)..... 3373
65.8.5 ERM Status Register 1 (SR1)..... 3378
65.8.6 ERM Memory a Error Address Register (EAR0 - EAR10)..... 3380
65.8.7 ERM Memory a Syndrome Register (SYN0 - SYN10)..... 3381

65.8.8 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT 10)....3382
65.9 ERM_PFE1 register descriptions....3383
65.9.1 ERM_PFE1 memory map....3383
65.9.2 ERM Configuration Register 0 (CR0)....3385
65.9.3 ERM Configuration Register 1 (CR1)....3389
65.9.4 ERM Status Register 0 (SR0)....3392
65.9.5 ERM Status Register 1 (SR1)....3397
65.9.6 ERM Memory a Error Address Register (EAR0 - EAR12)....3401
65.9.7 ERM Memory a Syndrome Register (SYN0 - SYN12)....3401
65.9.8 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT 12)....3402
65.10 ERM_PFE10 register descriptions....3403
65.10.1 ERM_PFE10 memory map....3404
65.10.2 ERM Configuration Register 0 (CR0)....3406
65.10.3 ERM Configuration Register 1 (CR1)....3410
65.10.4 ERM Status Register 0 (SR0)....3413
65.10.5 ERM Status Register 1 (SR1)....3418
65.10.6 ERM Memory a Error Address Register (EAR0 - EAR12)....3421
65.10.7 ERM Memory a Syndrome Register (SYN0 - SYN12)....3422
65.10.8 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT12)....3423
65.11 ERM_PFE11 register descriptions....3424
65.11.1 ERM_PFE11 memory map....3424
65.11.2 ERM Configuration Register 0 (CR0)....3427
65.11.3 ERM Configuration Register 1 (CR1)....3431
65.11.4 ERM Configuration Register 2 (CR2)....3435
65.11.5 ERM Status Register 0 (SR0)....3439
65.11.6 ERM Status Register 1 (SR1)....3444
65.11.7 ERM Status Register 2 (SR2)....3449
65.11.8 ERM Memory a Error Address Register (EAR0 - EAR23)....3453
65.11.9 ERM Memory a Syndrome Register (SYN0 - SYN23)....3454
65.11.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT23)....3455
65.12 ERM_PFE12 register descriptions....3457
65.12.1 ERM_PFE12 memory map....3457
65.12.2 ERM Configuration Register 0 (CR0)....3460
65.12.3 ERM Configuration Register 1 (CR1)....3464
65.12.4 ERM Configuration Register 2 (CR2)....3468
65.12.5 ERM Status Register 0 (SR0)....3470
65.12.6 ERM Status Register 1 (SR1)....3474
65.12.7 ERM Status Register 2 (SR2)....3479
65.12.8 ERM Memory a Error Address Register (EAR0 - EAR16)....3481
65.12.9 ERM Memory a Syndrome Register (SYN0 - SYN16)....3481
65.12.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)....3482
65.13 ERM_PFE13 register descriptions....3484
65.13.1 ERM_PFE13 memory map....3484
65.13.2 ERM Configuration Register 0 (CR0)....3486
65.13.3 ERM Configuration Register 1 (CR1)....3490
65.13.4 ERM Configuration Register 2 (CR2)....3494
65.13.5 ERM Status Register 0 (SR0)....3497
65.13.6 ERM Status Register 1 (SR1)....3501
65.13.7 ERM Status Register 2 (SR2)....3506
65.13.8 ERM Memory a Error Address Register (EAR0 - EAR17)....3508

65.13.9 ERM Memory a Syndrome Register (SYN0 - SYN17)....3509
65.13.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT17)....3510
65.14 ERM_PFE14 register descriptions....3511
65.14.1 ERM_PFE14 memory map....3511
65.14.2 ERM Configuration Register 0 (CR0)....3513
65.14.3 ERM Configuration Register 1 (CR1)....3517
65.14.4 ERM Status Register 0 (SR0)....3520
65.14.5 ERM Status Register 1 (SR1)....3525
65.14.6 ERM Memory a Error Address Register (EAR0 - EAR12)....3529
65.14.7 ERM Memory a Syndrome Register (SYN0 - SYN12)....3529
65.14.8 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT12)....3530
65.15 ERM_PFE15 register descriptions....3531
65.15.1 ERM_PFE15 memory map....3532
65.15.2 ERM Configuration Register 0 (CR0)....3534
65.15.3 ERM Configuration Register 1 (CR1)....3538
65.15.4 ERM Status Register 0 (SR0)....3541
65.15.5 ERM Status Register 1 (SR1)....3546
65.15.6 ERM Memory a Error Address Register (EAR0 - EAR12)....3549
65.15.7 ERM Memory a Syndrome Register (SYN0 - SYN12)....3550
65.15.8 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT12)....3551
65.16 ERM_PFE2 register descriptions....3552
65.16.1 ERM_PFE2 memory map....3552
65.16.2 ERM Configuration Register 0 (CR0)....3555
65.16.3 ERM Configuration Register 1 (CR1)....3559
65.16.4 ERM Configuration Register 2 (CR2)....3563
65.16.5 ERM Status Register 0 (SR0)....3565
65.16.6 ERM Status Register 1 (SR1)....3569
65.16.7 ERM Status Register 2 (SR2)....3574
65.16.8 ERM Memory a Error Address Register (EAR0 - EAR16)....3576
65.16.9 ERM Memory a Syndrome Register (SYN0 - SYN16)....3576
65.16.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)....3577
65.17 ERM_PFE3 register descriptions....3579
65.17.1 ERM_PFE3 memory map....3579
65.17.2 ERM Configuration Register 0 (CR0)....3581
65.17.3 ERM Configuration Register 1 (CR1)....3585
65.17.4 ERM Configuration Register 2 (CR2)....3589
65.17.5 ERM Status Register 0 (SR0)....3591
65.17.6 ERM Status Register 1 (SR1)....3596
65.17.7 ERM Status Register 2 (SR2)....3601
65.17.8 ERM Memory a Error Address Register (EAR0 - EAR16)....3602
65.17.9 ERM Memory a Syndrome Register (SYN0 - SYN16)....3603
65.17.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)....3604
65.18 ERM_PFE4 register descriptions....3605
65.18.1 ERM_PFE4 memory map....3606
65.18.2 ERM Configuration Register 0 (CR0)....3608
65.18.3 ERM Configuration Register 1 (CR1)....3612
65.18.4 ERM Configuration Register 2 (CR2)....3616
65.18.5 ERM Status Register 0 (SR0)....3618
65.18.6 ERM Status Register 1 (SR1)....3623
65.18.7 ERM Status Register 2 (SR2)....3628

65.18.8 ERM Memory a Error Address Register (EAR0 - EAR16)..... 3629
65.18.9 ERM Memory a Syndrome Register (SYN0 - SYN16)..... 3630
65.18.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)..... 3631
65.19 ERM_PFE5 register descriptions..... 3632
65.19.1 ERM_PFE5 memory map..... 3633
65.19.2 ERM Configuration Register 0 (CR0)..... 3635
65.19.3 ERM Configuration Register 1 (CR1)..... 3639
65.19.4 ERM Configuration Register 2 (CR2)..... 3643
65.19.5 ERM Status Register 0 (SR0)..... 3645
65.19.6 ERM Status Register 1 (SR1)..... 3650
65.19.7 ERM Status Register 2 (SR2)..... 3655
65.19.8 ERM Memory a Error Address Register (EAR0 - EAR16)..... 3656
65.19.9 ERM Memory a Syndrome Register (SYN0 - SYN16)..... 3657
65.19.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)..... 3658
65.20 ERM_PFE6 register descriptions..... 3659
65.20.1 ERM_PFE6 memory map..... 3660
65.20.2 ERM Configuration Register 0 (CR0)..... 3662
65.20.3 ERM Configuration Register 1 (CR1)..... 3666
65.20.4 ERM Configuration Register 2 (CR2)..... 3670
65.20.5 ERM Status Register 0 (SR0)..... 3672
65.20.6 ERM Status Register 1 (SR1)..... 3677
65.20.7 ERM Status Register 2 (SR2)..... 3682
65.20.8 ERM Memory a Error Address Register (EAR0 - EAR16)..... 3683
65.20.9 ERM Memory a Syndrome Register (SYN0 - SYN16)..... 3684
65.20.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)..... 3685
65.21 ERM_PFE7 register descriptions..... 3686
65.21.1 ERM_PFE7 memory map..... 3687
65.21.2 ERM Configuration Register 0 (CR0)..... 3689
65.21.3 ERM Configuration Register 1 (CR1)..... 3693
65.21.4 ERM Configuration Register 2 (CR2)..... 3697
65.21.5 ERM Status Register 0 (SR0)..... 3699
65.21.6 ERM Status Register 1 (SR1)..... 3704
65.21.7 ERM Status Register 2 (SR2)..... 3709
65.21.8 ERM Memory a Error Address Register (EAR0 - EAR16)..... 3710
65.21.9 ERM Memory a Syndrome Register (SYN0 - SYN16)..... 3711
65.21.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)..... 3712
65.22 ERM_PFE8 register descriptions..... 3713
65.22.1 ERM_PFE8 memory map..... 3714
65.22.2 ERM Configuration Register 0 (CR0)..... 3716
65.22.3 ERM Configuration Register 1 (CR1)..... 3720
65.22.4 ERM Configuration Register 2 (CR2)..... 3724
65.22.5 ERM Status Register 0 (SR0)..... 3726
65.22.6 ERM Status Register 1 (SR1)..... 3731
65.22.7 ERM Status Register 2 (SR2)..... 3736
65.22.8 ERM Memory a Error Address Register (EAR0 - EAR16)..... 3737
65.22.9 ERM Memory a Syndrome Register (SYN0 - SYN16)..... 3738
65.22.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)..... 3739
65.23 ERM_PFE9 register descriptions..... 3740
65.23.1 ERM_PFE9 memory map..... 3741
65.23.2 ERM Configuration Register 0 (CR0)..... 3743

65.23.3 ERM Configuration Register 1 (CR1)....3747
65.23.4 ERM Configuration Register 2 (CR2)....3751
65.23.5 ERM Status Register 0 (SR0)....3753
65.23.6 ERM Status Register 1 (SR1)....3758
65.23.7 ERM Status Register 2 (SR2)....3763
65.23.8 ERM Memory a Error Address Register (EAR0 - EAR16)....3764
65.23.9 ERM Memory a Syndrome Register (SYN0 - SYN16)....3765
65.23.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)....3766
65.24 ERM_STDBY_SRAM register descriptions....3767
65.24.1 ERM_STDBY_SRAM memory map....3768
65.24.2 ERM Configuration Register 0 (CR0)....3768
65.24.3 ERM Status Register 0 (SR0)....3770
65.24.4 ERM Memory 0 Error Address Register (EAR0)....3772
65.24.5 ERM Memory 0 Syndrome Register (SYN0)....3773
65.24.6 ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)....3774
65.25 ERM_eDMA0 register descriptions....3774
65.25.1 ERM_eDMA0 memory map....3775
65.25.2 ERM Configuration Register 0 (CR0)....3775
65.25.3 ERM Status Register 0 (SR0)....3777
65.25.4 ERM Memory 0 Error Address Register (EAR0)....3779
65.25.5 ERM Memory 0 Syndrome Register (SYN0)....3779
65.25.6 ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)....3780
65.26 ERM_eDMA1 register descriptions....3781
65.26.1 ERM_eDMA1 memory map....3781
65.26.2 ERM Configuration Register 0 (CR0)....3782
65.26.3 ERM Status Register 0 (SR0)....3784
65.26.4 ERM Memory 0 Error Address Register (EAR0)....3786
65.26.5 ERM Memory 0 Syndrome Register (SYN0)....3786
65.26.6 ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)....3787
65.27 ERM register descriptions....3788
65.27.1 ERM memory map....3788
65.27.2 ERM Configuration Register 0 (CR0)....3790
65.27.3 ERM Configuration Register 1 (CR1)....3794
65.27.4 ERM Status Register 0 (SR0)....3796
65.27.5 ERM Status Register 1 (SR1)....3800
65.27.6 ERM Memory a Error Address Register (EAR0 - EAR6)....3802
65.27.7 ERM Memory a Syndrome Register (SYN0 - SYN6)....3803
65.27.8 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT8)....3804
65.28 Functional description....3805
65.28.1 Single-bit correction events....3805
65.28.2 Non-correctable error events....3805
65.29 Initialization....3806
65.30 Glossary....3806

Chapter 66 Fault Collection and Control Unit (FCCU)....3807
66.1 Chip-specific FCCU information....3807
66.1.1 Chip-boundary FCCU signals....3807
66.1.2 Fault mapping....3807
66.1.3 Clock frequency....3807
66.1.4 FOSU timer interval....3807
66.1.5 CFG register event bit values by source (C)....3807
66.1.6 Supported internal chip reactions....3809

66.1.7 Supported FCCU EOUT fault output modes (protocols)....3809
66.1.8 Recommended reaction for the fault NCF[21] NoC Mission_Fault_0....3809
66.1.9 NCFs reaction restriction....3809
66.1.10 Clearing FCCU faults....3809
66.2 Introduction....3809
66.3 Features....3809
66.4 Block diagram....3810
66.5 Signal description....3811
66.5.1 Signals....3811
66.6 Functional description....3813
66.6.1 Definitions....3813
66.6.2 FSM description....3813
66.6.3 Fault priority scheme and nesting....3815
66.6.4 Fault recovery....3815
66.6.5 EOUT interface....3817
66.6.5.1 Bi-Stable protocol....3819
66.7 Prepare FCCU for configuration....3819
66.7.1 Introduction....3819
66.7.2 About preparing FCCU for configuration....3820
66.7.3 Configure the CONFIG state....3820
66.7.4 Put FCCU in the CONFIG state....3820
66.8 Configure FCCU....3820
66.8.1 Introduction....3820
66.8.2 About configuring FCCU....3820
66.8.3 Configure the non-critical fault channels....3821
66.9 Put FCCU in the NORMAL state....3821
66.9.1 Introduction....3821
66.9.2 About putting FCCU in NORMAL state....3821
66.9.3 Put FCCU in the NORMAL state....3821
66.10 Manage faults....3822
66.10.1 Introduction....3822
66.10.2 Determine if there are any unrecovered non-critical faults....3822
66.10.3 Recover a software-recoverable non-critical fault....3822
66.10.4 Clear the freeze-status indicators....3822
66.11 Run operations....3823
66.11.1 Introduction....3823
66.11.2 About running operations....3823
66.11.3 Run an operation....3823
66.12 Register descriptions....3823
66.12.1 FCCU register descriptions....3823
66.12.1.1 FCCU memory map....3823
66.12.1.2 Control (CTRL)....3825
66.12.1.3 Control Key (CTRLK)....3827
66.12.1.4 Configuration (CFG)....3828
66.12.1.5 Non-critical Fault Configuration (NCF_CFG0 - NCF_CFG3)....3830
66.12.1.6 Non-critical Fault-State Configuration (NCFS_CFG0 - NCFS_CFG7)....3832
66.12.1.7 Non-critical Fault Status (NCF_S0 - NCF_S3)....3834
66.12.1.8 Non-critical Fault Key (NCFK)....3835
66.12.1.9 Non-critical Fault Enable (NCF_E0 - NCF_E3)....3836
66.12.1.10 Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0 - NCF_TOE3)....3837
66.12.1.11 Non-critical-Fault Alarm-State Timeout Interval (NCF_TO)....3839
66.12.1.12 Configuration-State Timeout Interval (CFG_TO)....3840
66.12.1.13 IO Control (EINOUT)....3841
66.12.1.14 Status (STAT)....3843
66.12.1.15 Normal-to-Alarm Freeze Status (N2AF_STATUS)....3844

66.12.1.16 Alarm-to-Fault Freeze Status (A2FF_STATUS)....3845
66.12.1.17 Normal-to-Fault Freeze Status (N2FF_STATUS)....3847
66.12.1.18 Fault-to-Alarm Freeze Status (F2AF_STATUS)....3848
66.12.1.19 Non-critical Fault Fake (NCFF)....3849
66.12.1.20 IRQ Status (IRQ_STAT)....3850
66.12.1.21 IRQ Enable (IRQ_EN)....3852
66.12.1.22 Transient Configuration Lock (TRANS_LOCK)....3853
66.12.1.23 Permanent Configuration Lock (PERMNT_LOCK)....3854
66.12.1.24 Delta T (DELTA_T)....3855
66.12.1.25 Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0 - IRQ_ALARM_EN3)....3856
66.12.1.26 Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0 - NMI_EN3)3858
66.12.1.27 Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0 - EOUT_SIG_EN3)....3859
66.12.1.28 Alarm-State Timer (TMR_ALARM)....3860
66.12.1.29 Configuration-State Timer (TMR_CFG)....3861
66.12.1.30 Fault-Output Timer (TMR_ETMR)....3862
66.12.2 Configuration registers....3863
66.12.2.1 Definition....3863
66.12.2.2 Configuration registers....3863
66.12.3 CFG register bit value at different events....3864
66.13 FOSU....3864
66.14 Use cases and limitations....3865
66.15 Glossary....3866

Chapter 67 Clock Monitoring Unit-Frequency Check (CMU_FC)....3867
67.1 Chip-specific CMU_FC information....3867
67.1.1 Overview....3867
67.2 Introduction....3867
67.2.1 Basic operation....3867
67.2.2 Features....3868
67.3 CMU_FC register descriptions....3869
67.3.1 CMU_FC memory map....3869
67.3.2 Global Configuration Register (GCR)....3870
67.3.3 Reference Count Configuration Register (RCCR)....3871
67.3.4 High Threshold Configuration Register (HTCR)....3872
67.3.5 Low Threshold Configuration Register (LTCR)....3872
67.3.6 Status Register (SR)....3873
67.3.7 Interrupt Enable Register (IER)....3875
67.4 Functional description....3882
67.4.1 Frequency checking....3882
67.4.1.1 Monitored clock lost....3883
67.5 Programming guidelines....3883
67.5.1 Programming RCCR[REF_CNT]....3883
67.5.2 Programming HFREF and LFREF....3884
67.5.3 Module programming sequence....3886
67.6 Glossary....3887

Chapter 68 Clock Monitoring Unit - Frequency Meter (CMU_FM)....3888
68.1 Chip-specific CMU_FM information....3888
68.1.1 Overview....3888
68.2 Introduction....3888
68.2.1 Basic operation....3888
68.2.2 Features....3888

68.3 CMU_FM register descriptions....3889
68.3.1 CMU_FM memory map....3889
68.3.2 Global Configuration Register (GCR)....3889
68.3.3 Reference Count Configuration Register (RCCR)....3890
68.3.4 Status Register (SR)....3891
68.4 Functional description....3892
68.5 Programming guidelines....3892
68.5.1 Programming RCCR[REF_CNT]....3892
68.5.2 Module programming sequence....3893

Chapter 69 Selftest Top General Purpose Registers (SELFTEST_GPR_TOP)....3895
69.1 Introduction....3895
69.2 SELFTEST_GPR_TOP register descriptions....3895
69.2.1 SELFTEST_GPR_TOP memory map....3895
69.2.2 Reset Domain Self-test Enable (Reset_Domain_Selftest_Enable_Register)....3895
69.2.3 Reset Domain Self-test Enable Status (Reset_Domain_Selftest_Enable_Status_Register)....3896

Chapter 70 Selftest General Purpose Registers (SELFTEST_GPR)....3898
70.1 Introduction....3898
70.2 Memory map for self-test GPRs....3898
70.3 SELFTEST_GPR register descriptions....3898
70.3.1 SELFTEST_GPR memory map....3898
70.3.2 Generic 0 (Generic_Reg0)....3899
70.3.3 Generic 1 (Generic_Reg1)....3900
70.3.4 LBIST Program (LBIST_Prog_Reg)....3900

Chapter 71 Self-Test Control Unit (STCU2)....3902
71.1 Chip-specific STCU2 information....3902
71.1.1 Supported BIST sequences....3902
71.1.2 Self-test overview....3902
71.1.3 STCU2 LBIST mapping....3903
71.1.4 STCU2 MBIST mapping....3905
71.1.5 Running MBIST sequences on CM7 memories....3907
71.1.6 Indicating fault state during the main reset domain self-test....3907
71.1.7 Error events management in main reset domain self-test....3907
71.1.8 Online self-test reset or interrupt generation....3908
71.1.9 AUTOLOCK_VALUE for register write access through STCU2_SKC....3908
71.1.10 PLL loss of lock during self test....3908
71.1.11 STCU2 BIST start (BSTART) register description....3908
71.1.12 STCU2 algorithm select (ALGOSEL) register description....3909
71.1.13 Self-test programming sequence....3910
71.1.14 Handling unrecoverable faults....3911
71.2 Introduction....3911
71.3 Main features....3911
71.4 Block diagram....3912
71.5 Peripheral bus interface....3913
71.6 BISTs and BIST partitions....3914
71.6.1 Definition: BIST....3914
71.6.2 Definition: BIST partition....3914
71.6.3 Example: BIST partitions on a chip....3914
71.6.4 Types of BIST partitions....3914
71.7 BIST sequences....3915

71.7.1 Definition: BIST sequence....3915
71.7.2 STCU2 executes MBISTs before LBISTs....3915
71.7.3 Example: Single-phase BIST sequence....3915
71.7.4 Example: Multi-phase BIST sequence....3915
71.7.5 Types of BIST sequences....3915
71.7.6 Supported BIST sequences....3916
71.8 STCU2 register descriptions....3916
71.8.1 STCU2 memory map....3916
71.8.2 STCU2 Run Software (RUNSW)....3929
71.8.3 STCU2 SK Code (SKC)....3930
71.8.4 STCU2 Configuration (CFG)....3931
71.8.5 STCU2 Watchdog Granularity (WDG)....3934
71.8.6 STCU2 Interrupt Flag (INT_FLG)....3935
71.8.7 STCU2 Error (ERR_STAT)....3936
71.8.8 STCU2 Error FM (ERR_FM)....3938
71.8.9 STCU2 Online LBIST Status (LBSSW0)....3939
71.8.10 STCU2 Online LBIST Status (LBSSW1)....3940
71.8.11 STCU2 Online LBIST End Flag (LBESW0)....3941
71.8.12 STCU2 Online LBIST End Flag (LBESW1)....3942
71.8.13 STCU2 Online LBIST Unrecoverable FM (LBUFM0)....3943
71.8.14 STCU2 Online LBIST Unrecoverable FM (LBUFM1)....3944
71.8.15 STCU2 Online MBIST Status (MBSSW0)....3945
71.8.16 STCU2 Online MBIST Status (MBSSW1)....3950
71.8.17 STCU2 Online MBIST Status (MBSSW2)....3955
71.8.18 STCU2 Online MBIST Status (MBSSW3)....3960
71.8.19 STCU2 Online MBIST End Flag (MBESW0)....3960
71.8.20 STCU2 Online MBIST End Flag (MBESW1)....3964
71.8.21 STCU2 Online MBIST End Flag (MBESW2)....3968
71.8.22 STCU2 Online MBIST End Flag (MBESW3)....3972
71.8.23 STCU2 MBIST Unrecoverable FM (MBUFM0)....3973
71.8.24 STCU2 MBIST Unrecoverable FM (MBUFM1)....3978
71.8.25 STCU2 MBIST Unrecoverable FM (MBUFM2)....3983
71.8.26 STCU2 MBIST Unrecoverable FM (MBUFM3)....3988
71.8.27 STCU2 LBIST Control (LB_CTRL0 - LB_CTRL59)....3989
71.8.28 STCU2 LBIST PC Stop (LB_PCS0 - LB_PCS59)....3993
71.8.29 STCU2 Online LBIST MISR Expected Low (LB_MISRELSW0 - LB_MISRELSW59)....3994
71.8.30 STCU2 Online LBIST MISR Expected High (LB_MISREHSW0 - LB_MISREHSW59)....3994
71.8.31 STCU2 Online LBIST MISR Read Low (LB_MISRRLSW0 - LB_MISRRLSW59)....3995
71.8.32 STCU2 Online LBIST MISR Read High (LB_MISRRHSW0 - LB_MISRRHSW59)....3996
71.8.33 STCU2 Algorithm Select (ALGOSEL)....3997
71.8.34 STCU2 MBIST Stagger (STGGR)....3997
71.8.35 STCU2 BIST Start (BSTART)....3998
71.8.36 STCU2 MBIST Control (MB_CTRL0 - MB_CTRL109)....3999
71.9 Functional description....4000
71.9.1 FSM description....4000
71.9.2 BIST scheduling....4000
71.9.3 ABORT management....4001
71.9.3.1 Hardware ABORT management....4001
71.9.4 FCCU interface....4001
71.9.5 Watchdogs....4001
71.9.5.1 BIST watchdog timer....4001
71.9.5.2 Register write-access watchdog timer....4001
71.10 Use cases and limitations....4001
71.11 Glossary....4002

Chapter 72 Safety by Software (SBSW)..... 4003
72.1 Introduction..... 4003
72.1.1 Acronyms and abbreviations..... 4003
72.2 Block diagram..... 4003
72.3 Signal descriptions..... 4006
72.4 Functional description..... 4006
72.4.1 SBSW controller..... 4007
72.4.1.1 Unlocking/locking the TMC configuration..... 4007
72.4.1.2 Unlocking/locking the TMWDP configuration..... 4008
72.4.2 TMC..... 4008
72.4.2.1 Digital comparator..... 4009
72.4.2.2 Timer..... 4009
72.4.2.3 Block diagram..... 4011
72.4.2.4 TMC states..... 4011
72.4.2.5 TMC events..... 4012
72.4.3 TMWDPI..... 4012
72.4.3.1 TMWDP description..... 4013
72.4.3.2 Block diagram..... 4014
72.4.3.3 TMWDPI registers..... 4015
72.4.3.4 TMWDP/TMWDPI events..... 4016
72.4.4 SBSW interface summary..... 4016
72.5 Memory map and register definition..... 4017
72.5.1 SBSW register descriptions..... 4017
72.5.1.1 SBSW memory map..... 4017
72.5.1.2 TMC Configuration Unlock (TMC_CONFIG_UNLOCK)..... 4028
72.5.1.3 TMWDP Configuration Unlock (TMWDP_CONFIG_UNLOCK)..... 4029
72.5.1.4 TMC Configuration Status (TMC_CONFIG_STATUS)..... 4030
72.5.1.5 TMWDP Configuration Status (TMWDP_CONFIG_STATUS)..... 4031
72.5.1.6 Debug Mode (DEBUG_MODE)..... 4032
72.5.1.7 TMC Fault Status (TMC_FAULT_STATUS)..... 4033
72.5.1.8 TMWDP Fault Status (TMWDP_FAULT_STATUS)..... 4033
72.5.1.9 TMC Configuration (TMC_0_CONFIG - TMC_31_CONFIG)..... 4034
72.5.1.10 TMC Distance (TMC_0_DISTANCE - TMC_31_DISTANCE)..... 4035
72.5.1.11 TMC Timeout (TMC_0_TIMEOUT - TMC_31_TIMEOUT)..... 4036
72.5.1.12 TMC Control (TMC_0_CONTROL - TMC_31_CONTROL)..... 4037
72.5.1.13 TMC R0 Data (TMC_0_R0 - TMC_31_R0)..... 4038
72.5.1.14 TMC R1 Data (TMC_0_R1 - TMC_31_R1)..... 4039
72.5.1.15 TMC Status (TMC_0_STATUS - TMC_31_STATUS)..... 4040
72.5.1.16 TMC Timer (TMC_0_TIMER - TMC_31_TIMER)..... 4041
72.5.1.17 TMWDP Configuration Address (TMWDP_CONFIG_ADDR)..... 4042
72.5.1.18 TMWDP Control (TMWDP_CONTROL)..... 4043
72.5.1.19 TMWDP Status (TMWDP_STATUS)..... 4044
72.5.1.20 TMWDP Automata Status (TMWDP_AUTOMATA_STATUS)..... 4045
72.5.1.21 TMWDP Automata Illegal Transition (TMWDP_AUTOMATA_ILLGL_TRANS)..... 4046
72.5.1.22 TMWDP Automata Time Violation (TMWDP_AUTOMATA_TIME_VIOLATION)..... 4047
72.5.1.23 TMWDP Automaton Status (AUT_0_STATUS - AUT_31_STATUS)..... 4048
72.5.1.24 TMWDP Automaton Progress Request (AUT_0_PRGS_REQ - AUT_31_PRGS_REQ)..... 4049
72.5.1.25 TMWDP Core Domain ID (TC_ID)..... 4051

Chapter 73 Error Injection Module (EIM)..... 4053
73.1 Chip-specific EIM information..... 4053
73.1.1 EIM instances..... 4053
73.1.2 ACP_EIM channel assignments..... 4053

73.1.3 EIM_0 channel assignments.... 4062
73.1.4 EIM_1 channel assignments.... 4064
73.1.5 EIM_2 channel assignments.... 4068
73.1.6 EIM_3 channel assignments.... 4070
73.1.7 Constraints to use EIM on A53 cache memories.... 4071
73.2 Chip-specific EIM information.... 4072
73.2.1 EIM instances.... 4072
73.2.2 EIM_MISC channel assignments.... 4076
73.2.3 EIM_PFE channel assignments.... 4078
73.2.4 EIM_LLCE channel assignments.... 4087
73.3 Introduction.... 4088
73.3.1 Overview.... 4088
73.3.2 Features.... 4089
73.4 EIM register descriptions.... 4089
73.4.1 EIM memory map.... 4092
73.4.2 Error Injection Module Configuration Register (EIMCR).... 4098
73.4.3 Error Injection Channel Enable register (EICHEN).... 4099
73.4.4 Error Injection Channel Descriptor n, Word0 (EICHD0_WORD0 - EICHD7_WORD0).... 4108
73.4.5 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD10_WORD1).... 4109
73.4.6 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD28_WORD2).... 4111
73.4.7 Error Injection Channel Descriptor n, Word3 (EICHD0_WORD3 - EICHD28_WORD3).... 4112
73.4.8 Error Injection Channel Descriptor n, Word4 (EICHD0_WORD4 - EICHD27_WORD4).... 4114
73.4.9 Error Injection Channel Descriptor n, Word5 (EICHD0_WORD5 - EICHD7_WORD5).... 4115
73.4.10 Error Injection Channel Descriptor n, Word6 (EICHD0_WORD6 - EICHD7_WORD6).... 4117
73.4.11 Error Injection Channel Descriptor n, Word7 (EICHD0_WORD7 - EICHD7_WORD7).... 4118
73.4.12 Error Injection Channel Descriptor n, Word8 (EICHD0_WORD8 - EICHD7_WORD8).... 4119
73.4.13 Error Injection Channel Descriptor n, Word0 (EICHD8_WORD0 - EICHD9_WORD0).... 4120
73.4.14 Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0).... 4121
73.4.15 Error Injection Channel Descriptor 11, Word0 (EICHD11_WORD0).... 4122
73.4.16 Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1).... 4123
73.4.17 Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0).... 4124
73.4.18 Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1).... 4125
73.4.19 Error Injection Channel Descriptor n, Word0 (EICHD13_WORD0 - EICHD15_WORD0).... 4126
73.4.20 Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1).... 4127
73.4.21 Error Injection Channel Descriptor n, Word1 (EICHD14_WORD1 - EICHD16_WORD1).... 4128
73.4.22 Error Injection Channel Descriptor 16, Word0 (EICHD16_WORD0).... 4129
73.4.23 Error Injection Channel Descriptor 17, Word0 (EICHD17_WORD0).... 4130
73.4.24 Error Injection Channel Descriptor 17, Word1 (EICHD17_WORD1).... 4131
73.4.25 Error Injection Channel Descriptor 18, Word0 (EICHD18_WORD0).... 4132
73.4.26 Error Injection Channel Descriptor 18, Word1 (EICHD18_WORD1).... 4133
73.4.27 Error Injection Channel Descriptor n, Word0 (EICHD19_WORD0 - EICHD21_WORD0).... 4134
73.4.28 Error Injection Channel Descriptor 19, Word1 (EICHD19_WORD1).... 4135
73.4.29 Error Injection Channel Descriptor n, Word1 (EICHD20_WORD1 - EICHD22_WORD1).... 4136
73.4.30 Error Injection Channel Descriptor 22, Word0 (EICHD22_WORD0).... 4137
73.4.31 Error Injection Channel Descriptor 23, Word0 (EICHD23_WORD0).... 4138
73.4.32 Error Injection Channel Descriptor 23, Word1 (EICHD23_WORD1).... 4139
73.4.33 Error Injection Channel Descriptor 24, Word0 (EICHD24_WORD0).... 4140
73.4.34 Error Injection Channel Descriptor 24, Word1 (EICHD24_WORD1).... 4141
73.4.35 Error Injection Channel Descriptor n, Word0 (EICHD25_WORD0 - EICHD27_WORD0).... 4142
73.4.36 Error Injection Channel Descriptor 25, Word1 (EICHD25_WORD1).... 4143
73.4.37 Error Injection Channel Descriptor n, Word1 (EICHD26_WORD1 - EICHD27_WORD1).... 4144
73.4.38 Error Injection Channel Descriptor 28, Word1 (EICHD28_WORD1).... 4145
73.4.39 Error Injection Channel Descriptor 29, Word1 (EICHD29_WORD1).... 4146
73.4.40 Error Injection Channel Descriptor 30, Word1 (EICHD30_WORD1).... 4147
73.5 EIM_0 register descriptions.... 4148

73.5.1 EIM_0 memory map....4149
73.5.2 Error Injection Module Configuration Register (EIMCR)....4150
73.5.3 Error Injection Channel Enable register (EICHEN)....4150
73.5.4 Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)....4154
73.5.5 Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)....4154
73.5.6 Error Injection Channel Descriptor 0, Word3 (EICHD0_WORD3)....4155
73.5.7 Error Injection Channel Descriptor 0, Word4 (EICHD0_WORD4)....4156
73.5.8 Error Injection Channel Descriptor 0, Word5 (EICHD0_WORD5)....4157
73.5.9 Error Injection Channel Descriptor 0, Word6 (EICHD0_WORD6)....4158
73.5.10 Error Injection Channel Descriptor 0, Word7 (EICHD0_WORD7)....4159
73.5.11 Error Injection Channel Descriptor 0, Word8 (EICHD0_WORD8)....4160
73.6 EIM_1 register descriptions....4161
73.6.1 EIM_1 memory map....4163
73.6.2 Error Injection Module Configuration Register (EIMCR)....4167
73.6.3 Error Injection Channel Enable register (EICHEN)....4168
73.6.4 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD14_WORD1)....4174
73.6.5 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD14_WORD2)....4176
73.6.6 Error Injection Channel Descriptor n, Word3 (EICHD0_WORD3 - EICHD14_WORD3)....4177
73.6.7 Error Injection Channel Descriptor n, Word4 (EICHD0_WORD4 - EICHD14_WORD4)....4178
73.6.8 Error Injection Channel Descriptor n, Word5 (EICHD0_WORD5 - EICHD14_WORD5)....4180
73.6.9 Error Injection Channel Descriptor n, Word6 (EICHD0_WORD6 - EICHD14_WORD6)....4181
73.6.10 Error Injection Channel Descriptor n, Word7 (EICHD0_WORD7 - EICHD14_WORD7)....4182
73.6.11 Error Injection Channel Descriptor n, Word8 (EICHD0_WORD8 - EICHD14_WORD8)....4184
73.7 EIM_2 register descriptions....4185
73.7.1 EIM_2 memory map....4186
73.7.2 Error Injection Module Configuration Register (EIMCR)....4188
73.7.3 Error Injection Channel Enable register (EICHEN)....4189
73.7.4 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD5_WORD1)....4193
73.7.5 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD5_WORD2)....4194
73.7.6 Error Injection Channel Descriptor n, Word3 (EICHD0_WORD3 - EICHD5_WORD3)....4195
73.7.7 Error Injection Channel Descriptor n, Word4 (EICHD0_WORD4 - EICHD5_WORD4)....4196
73.7.8 Error Injection Channel Descriptor n, Word5 (EICHD0_WORD5 - EICHD5_WORD5)....4197
73.7.9 Error Injection Channel Descriptor n, Word6 (EICHD0_WORD6 - EICHD5_WORD6)....4198
73.7.10 Error Injection Channel Descriptor n, Word7 (EICHD0_WORD7 - EICHD5_WORD7)....4199
73.7.11 Error Injection Channel Descriptor n, Word8 (EICHD0_WORD8 - EICHD5_WORD8)....4200
73.8 EIM_3 register descriptions....4201
73.8.1 EIM_3 memory map....4202
73.8.2 Error Injection Module Configuration Register (EIMCR)....4204
73.8.3 Error Injection Channel Enable register (EICHEN)....4204
73.8.4 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD3_WORD1)....4208
73.8.5 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD3_WORD2)....4209
73.8.6 Error Injection Channel Descriptor n, Word3 (EICHD0_WORD3 - EICHD3_WORD3)....4210
73.8.7 Error Injection Channel Descriptor n, Word4 (EICHD0_WORD4 - EICHD3_WORD4)....4211
73.8.8 Error Injection Channel Descriptor n, Word5 (EICHD0_WORD5 - EICHD3_WORD5)....4212
73.8.9 Error Injection Channel Descriptor n, Word6 (EICHD0_WORD6 - EICHD3_WORD6)....4213
73.8.10 Error Injection Channel Descriptor n, Word7 (EICHD0_WORD7 - EICHD3_WORD7)....4214
73.8.11 Error Injection Channel Descriptor n, Word8 (EICHD0_WORD8 - EICHD3_WORD8)....4215
73.9 EIM_LLCE register descriptions....4216
73.9.1 EIM_LLCE memory map....4217
73.9.2 Error Injection Module Configuration Register (EIMCR)....4218
73.9.3 Error Injection Channel Enable register (EICHEN)....4219
73.9.4 Error Injection Channel Descriptor n, Word0 (EICHD0_WORD0 - EICHD6_WORD0)....4223
73.9.5 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD6_WORD1)....4224
73.10 EIM_MISC register descriptions....4225
73.10.1 EIM_MISC memory map....4227

73.10.2 Error Injection Module Configuration Register (EIMCR)..... 4229
73.10.3 Error Injection Channel Enable register (EICHEN)..... 4229
73.10.4 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD4_WORD1)..... 4236
73.10.5 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD15_WORD2)... 4237
73.10.6 Error Injection Channel Descriptor n, Word3 (EICHD0_WORD3 - EICHD14_WORD3)... 4238
73.10.7 Error Injection Channel Descriptor n, Word4 (EICHD2_WORD4 - EICHD4_WORD4)..... 4240
73.10.8 Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1).....4241
73.10.9 Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1).....4242
73.10.10 Error Injection Channel Descriptor n, Word1 (EICHD7_WORD1 - EICHD8_WORD1)... 4243
73.10.11 Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1).....4244
73.10.12 Error Injection Channel Descriptor n, Word1 (EICHD10_WORD1 - EICHD11_WORD1)4245
73.10.13 Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1).....4246
73.10.14 Error Injection Channel Descriptor n, Word1 (EICHD13_WORD1 - EICHD14_WORD1)4247
73.10.15 Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1).....4248
73.10.16 Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1).....4249
73.11 EIM_PFE0 register descriptions..... 4249
73.11.1 EIM_PFE0 memory map..... 4252
73.11.2 Error Injection Module Configuration Register (EIMCR)..... 4254
73.11.3 Error Injection Channel Enable register (EICHEN)..... 4255
73.11.4 Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0).....4265
73.11.5 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD1_WORD1)..... 4266
73.11.6 Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0).....4267
73.11.7 Error Injection Channel Descriptor n, Word2 (EICHD1_WORD2 - EICHD28_WORD2)... 4268
73.11.8 Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0).....4269
73.11.9 Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1).....4270
73.11.10 Error Injection Channel Descriptor 2, Word3 (EICHD2_WORD3).....4271
73.11.11 Error Injection Channel Descriptor 2, Word4 (EICHD2_WORD4).....4272
73.11.12 Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0).....4273
73.11.13 Error Injection Channel Descriptor n, Word1 (EICHD3_WORD1 - EICHD4_WORD1)... 4274
73.11.14 Error Injection Channel Descriptor 4, Word0 (EICHD4_WORD0).....4275
73.11.15 Error Injection Channel Descriptor n, Word0 (EICHD5_WORD0 - EICHD6_WORD0)... 4276
73.11.16 Error Injection Channel Descriptor n, Word1 (EICHD5_WORD1 - EICHD7_WORD1)... 4277
73.11.17 Error Injection Channel Descriptor 7, Word0 (EICHD7_WORD0).....4278
73.11.18 Error Injection Channel Descriptor 8, Word0 (EICHD8_WORD0).....4279
73.11.19 Error Injection Channel Descriptor n, Word1 (EICHD8_WORD1 - EICHD9_WORD1)... 4280
73.11.20 Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0).....4281
73.11.21 Error Injection Channel Descriptor n, Word0 (EICHD10_WORD0 - EICHD11_WORD0)4282
73.11.22 Error Injection Channel Descriptor n, Word1 (EICHD10_WORD1 - EICHD12_WORD1)4283
73.11.23 Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0).....4284
73.11.24 Error Injection Channel Descriptor 13, Word0 (EICHD13_WORD0).....4285
73.11.25 Error Injection Channel Descriptor n, Word1 (EICHD13_WORD1 - EICHD14_WORD1)4286
73.11.26 Error Injection Channel Descriptor 14, Word0 (EICHD14_WORD0).....4287
73.11.27 Error Injection Channel Descriptor n, Word0 (EICHD15_WORD0 - EICHD16_WORD0)4288
73.11.28 Error Injection Channel Descriptor n, Word1 (EICHD15_WORD1 - EICHD17_WORD1)4289
73.11.29 Error Injection Channel Descriptor 17, Word0 (EICHD17_WORD0).....4290
73.11.30 Error Injection Channel Descriptor 18, Word0 (EICHD18_WORD0).....4291
73.11.31 Error Injection Channel Descriptor n, Word1 (EICHD18_WORD1 - EICHD19_WORD1)4292
73.11.32 Error Injection Channel Descriptor 19, Word0 (EICHD19_WORD0).....4293
73.11.33 Error Injection Channel Descriptor n, Word0 (EICHD20_WORD0 - EICHD21_WORD0)4294
73.11.34 Error Injection Channel Descriptor n, Word1 (EICHD20_WORD1 - EICHD22_WORD1)4295
73.11.35 Error Injection Channel Descriptor 22, Word0 (EICHD22_WORD0).....4296
73.11.36 Error Injection Channel Descriptor 23, Word0 (EICHD23_WORD0).....4297
73.11.37 Error Injection Channel Descriptor n, Word1 (EICHD23_WORD1 - EICHD24_WORD1)4298
73.11.38 Error Injection Channel Descriptor 24, Word0 (EICHD24_WORD0).....4299
73.11.39 Error Injection Channel Descriptor n, Word0 (EICHD25_WORD0 - EICHD26_WORD0)4300

73.11.40 Error Injection Channel Descriptor n, Word1 (EICHD25_WORD1 - EICHD27_WORD1)4301
73.11.41 Error Injection Channel Descriptor 27, Word0 (EICHD27_WORD0).....4302
73.11.42 Error Injection Channel Descriptor 28, Word0 (EICHD28_WORD0).....4303
73.11.43 Error Injection Channel Descriptor n, Word1 (EICHD28_WORD1 - EICHD29_WORD1)4304
73.11.44 Error Injection Channel Descriptor 29, Word0 (EICHD29_WORD0).....4305
73.11.45 Error Injection Channel Descriptor n, Word0 (EICHD30_WORD0 - EICHD31_WORD0)4306
73.11.46 Error Injection Channel Descriptor n, Word1 (EICHD30_WORD1 - EICHD31_WORD1)4307
73.12 EIM_PFE1 register descriptions..... 4308
73.12.1 EIM_PFE1 memory map..... 4310
73.12.2 Error Injection Module Configuration Register (EIMCR)..... 4311
73.12.3 Error Injection Channel Enable register (EICHEN)..... 4312
73.12.4 Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0).....4318
73.12.5 Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1).....4319
73.12.6 Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0).....4320
73.12.7 Error Injection Channel Descriptor n, Word1 (EICHD1_WORD1 - EICHD2_WORD1)..... 4321
73.12.8 Error Injection Channel Descriptor n, Word2 (EICHD1_WORD2 - EICHD14_WORD2)... 4322
73.12.9 Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0).....4323
73.12.10 Error Injection Channel Descriptor n, Word0 (EICHD3_WORD0 - EICHD4_WORD0)... 4324
73.12.11 Error Injection Channel Descriptor n, Word1 (EICHD3_WORD1 - EICHD5_WORD1)... 4325
73.12.12 Error Injection Channel Descriptor 5, Word0 (EICHD5_WORD0).....4326
73.12.13 Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0).....4327
73.12.14 Error Injection Channel Descriptor n, Word1 (EICHD6_WORD1 - EICHD7_WORD1)... 4328
73.12.15 Error Injection Channel Descriptor 7, Word0 (EICHD7_WORD0).....4329
73.12.16 Error Injection Channel Descriptor n, Word0 (EICHD8_WORD0 - EICHD9_WORD0)... 4330
73.12.17 Error Injection Channel Descriptor n, Word1 (EICHD8_WORD1 - EICHD13_WORD1). 4331
73.12.18 Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0).....4332
73.12.19 Error Injection Channel Descriptor 11, Word0 (EICHD11_WORD0).....4333
73.12.20 Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0).....4334
73.12.21 Error Injection Channel Descriptor n, Word0 (EICHD13_WORD0 - EICHD14_WORD0)4335
73.12.22 Error Injection Channel Descriptor n, Word3 (EICHD13_WORD3 - EICHD14_WORD3)4336
73.12.23 Error Injection Channel Descriptor n, Word4 (EICHD13_WORD4 - EICHD14_WORD4)4337
73.12.24 Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1).....4338
73.13 EIM_PFE2 register descriptions..... 4339
73.13.1 EIM_PFE2 memory map..... 4341
73.13.2 Error Injection Module Configuration Register (EIMCR)..... 4344
73.13.3 Error Injection Channel Enable register (EICHEN)..... 4345
73.13.4 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD3_WORD1)..... 4354
73.13.5 Error Injection Channel Descriptor n, Word1 (EICHD4_WORD1 - EICHD5_WORD1)..... 4355
73.13.6 Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1).....4356
73.13.7 Error Injection Channel Descriptor n, Word1 (EICHD7_WORD1 - EICHD8_WORD1)..... 4357
73.13.8 Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1).....4358
73.13.9 Error Injection Channel Descriptor n, Word1 (EICHD10_WORD1 - EICHD11_WORD1). 4359
73.13.10 Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1).....4360
73.13.11 Error Injection Channel Descriptor n, Word1 (EICHD13_WORD1 - EICHD14_WORD1)4361
73.13.12 Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1).....4362
73.13.13 Error Injection Channel Descriptor n, Word0 (EICHD16_WORD0 - EICHD21_WORD0)4363
73.13.14 Error Injection Channel Descriptor n, Word1 (EICHD16_WORD1 - EICHD30_WORD1)4364
73.13.15 Error Injection Channel Descriptor n, Word2 (EICHD16_WORD2 - EICHD30_WORD2)4366
73.13.16 Error Injection Channel Descriptor 22, Word0 (EICHD22_WORD0).....4367
73.13.17 Error Injection Channel Descriptor 22, Word3 (EICHD22_WORD3).....4368
73.13.18 Error Injection Channel Descriptor 22, Word4 (EICHD22_WORD4).....4369
73.13.19 Error Injection Channel Descriptor 22, Word5 (EICHD22_WORD5).....4370
73.13.20 Error Injection Channel Descriptor 22, Word6 (EICHD22_WORD6).....4371
73.13.21 Error Injection Channel Descriptor 22, Word7 (EICHD22_WORD7).....4372
73.13.22 Error Injection Channel Descriptor 22, Word8 (EICHD22_WORD8).....4373

73.13.23 Error Injection Channel Descriptor n, Word0 (EICHD23_WORD0 - EICHD30_WORD0)4374
73.13.24 Error Injection Channel Descriptor 31, Word0 (EICHD31_WORD0).....4375
73.13.25 Error Injection Channel Descriptor 31, Word1 (EICHD31_WORD1).....4376
73.14 EIM_PFE3 register descriptions..... 4377
73.14.1 EIM_PFE3 memory map..... 4379
73.14.2 Error Injection Module Configuration Register (EIMCR)..... 4383
73.14.3 Error Injection Channel Enable register (EICHEN)..... 4384
73.14.4 Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0).....4393
73.14.5 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD8_WORD1)..... 4394
73.14.6 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD31_WORD2)... 4396
73.14.7 Error Injection Channel Descriptor n, Word3 (EICHD0_WORD3 - EICHD10_WORD3)... 4397
73.14.8 Error Injection Channel Descriptor n, Word4 (EICHD0_WORD4 - EICHD10_WORD4)... 4398
73.14.9 Error Injection Channel Descriptor n, Word5 (EICHD0_WORD5 - EICHD10_WORD5)... 4399
73.14.10 Error Injection Channel Descriptor n, Word6 (EICHD0_WORD6 - EICHD10_WORD6). 4400
73.14.11 Error Injection Channel Descriptor n, Word7 (EICHD0_WORD7 - EICHD10_WORD7). 4401
73.14.12 Error Injection Channel Descriptor n, Word8 (EICHD0_WORD8 - EICHD10_WORD8). 4402
73.14.13 Error Injection Channel Descriptor n, Word0 (EICHD1_WORD0 - EICHD8_WORD0)... 4403
73.14.14 Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0).....4404
73.14.15 Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1).....4405
73.14.16 Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0).....4406
73.14.17 Error Injection Channel Descriptor n, Word1 (EICHD10_WORD1 - EICHD18_WORD1)4407
73.14.18 Error Injection Channel Descriptor n, Word0 (EICHD11_WORD0 - EICHD18_WORD0)4408
73.14.19 Error Injection Channel Descriptor 19, Word0 (EICHD19_WORD0).....4410
73.14.20 Error Injection Channel Descriptor 19, Word1 (EICHD19_WORD1).....4411
73.14.21 Error Injection Channel Descriptor n, Word0 (EICHD20_WORD0 - EICHD31_WORD0)4412
73.14.22 Error Injection Channel Descriptor n, Word1 (EICHD20_WORD1 - EICHD31_WORD1)4413
73.15 EIM_PFE4 register descriptions..... 4414
73.15.1 EIM_PFE4 memory map..... 4415
73.15.2 Error Injection Module Configuration Register (EIMCR)..... 4416
73.15.3 Error Injection Channel Enable register (EICHEN)..... 4417
73.15.4 Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0).....4422
73.15.5 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD9_WORD1)..... 4423
73.15.6 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD9_WORD2)..... 4425
73.15.7 Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0).....4426
73.15.8 Error Injection Channel Descriptor n, Word0 (EICHD2_WORD0 - EICHD9_WORD0)..... 4427
73.16 EIM_PFE5 register descriptions..... 4428
73.16.1 EIM_PFE5 memory map..... 4430
73.16.2 Error Injection Module Configuration Register (EIMCR)..... 4433
73.16.3 Error Injection Channel Enable register (EICHEN)..... 4434
73.16.4 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD1_WORD1)..... 4443
73.16.5 Error Injection Channel Descriptor n, Word1 (EICHD2_WORD1 - EICHD3_WORD1)..... 4444
73.16.6 Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1).....4445
73.16.7 Error Injection Channel Descriptor n, Word1 (EICHD5_WORD1 - EICHD6_WORD1)..... 4446
73.16.8 Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1).....4447
73.16.9 Error Injection Channel Descriptor n, Word0 (EICHD8_WORD0 - EICHD13_WORD0)... 4448
73.16.10 Error Injection Channel Descriptor n, Word1 (EICHD8_WORD1 - EICHD27_WORD1). 4449
73.16.11 Error Injection Channel Descriptor n, Word2 (EICHD8_WORD2 - EICHD28_WORD2). 4451
73.16.12 Error Injection Channel Descriptor 14, Word0 (EICHD14_WORD0).....4452
73.16.13 Error Injection Channel Descriptor 15, Word0 (EICHD15_WORD0).....4453
73.16.14 Error Injection Channel Descriptor 16, Word0 (EICHD16_WORD0).....4454
73.16.15 Error Injection Channel Descriptor 17, Word0 (EICHD17_WORD0).....4455
73.16.16 Error Injection Channel Descriptor 18, Word0 (EICHD18_WORD0).....4456
73.16.17 Error Injection Channel Descriptor 19, Word0 (EICHD19_WORD0).....4457
73.16.18 Error Injection Channel Descriptor 20, Word0 (EICHD20_WORD0).....4458
73.16.19 Error Injection Channel Descriptor 21, Word0 (EICHD21_WORD0).....4459

73.16.20 Error Injection Channel Descriptor 22, Word0 (EICHD22_WORD0)....4460
73.16.21 Error Injection Channel Descriptor 23, Word0 (EICHD23_WORD0)....4461
73.16.22 Error Injection Channel Descriptor 24, Word0 (EICHD24_WORD0)....4462
73.16.23 Error Injection Channel Descriptor 25, Word0 (EICHD25_WORD0)....4463
73.16.24 Error Injection Channel Descriptor 25, Word3 (EICHD25_WORD3)....4464
73.16.25 Error Injection Channel Descriptor 25, Word4 (EICHD25_WORD4)....4465
73.16.26 Error Injection Channel Descriptor 26, Word0 (EICHD26_WORD0)....4466
73.16.27 Error Injection Channel Descriptor 27, Word0 (EICHD27_WORD0)....4467
73.16.28 Error Injection Channel Descriptor 28, Word0 (EICHD28_WORD0)....4468
73.16.29 Error Injection Channel Descriptor n, Word1 (EICHD28_WORD1 - EICHD29_WORD1)4469
73.16.30 Error Injection Channel Descriptor 29, Word0 (EICHD29_WORD0)....4470
73.16.31 Error Injection Channel Descriptor n, Word0 (EICHD30_WORD0 - EICHD31_WORD0)4471
73.16.32 Error Injection Channel Descriptor n, Word1 (EICHD30_WORD1 - EICHD31_WORD1)4472
73.17 EIM_PFE6 register descriptions.... 4473
73.17.1 EIM_PFE6 memory map.... 4474
73.17.2 Error Injection Module Configuration Register (EIMCR).... 4474
73.17.3 Error Injection Channel Enable register (EICHEN).... 4475
73.17.4 Error Injection Channel Descriptor n, Word0 (EICHD0_WORD0 - EICHD1_WORD0)..... 4479
73.17.5 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD1_WORD1)..... 4480
73.18 Functional description....4480
73.19 Glossary.... 4481

Chapter 74 Cyclic Redundancy Check (CRC) Unit....4482
74.1 Introduction....4482
74.2 Main features....4482
74.3 Block diagram....4482
74.4 Signal description.... 4482
74.4.1 Peripheral bus interface.... 4482
74.5 CRC register descriptions....4483
74.5.1 CRC memory map.... 4483
74.5.2 Configuration Register (CFG1 - CFG3)....4484
74.5.3 Input Register (INP1 - INP3).... 4485
74.5.4 Current Status Register (CSTAT1 - CSTAT3).... 4486
74.5.5 Output Register (OUTP1 - OUTP3).... 4487
74.6 Functional description....4488
74.7 Use cases....4490
74.7.1 Programming example....4490
74.7.2 Register programming....4491
74.7.2.1 Configuration Register values....4492
74.7.2.2 Current Status Register initial values....4492
74.7.2.3 Examples of input data and expected output data....4492

Chapter 75 Debug....4494
75.1 Introduction....4494
75.2 Block diagram....4494
75.2.1 S32G accelerator subsystem debug block diagram....4495
75.3 Glossary.... 4496
75.4 Features.... 4497
75.5 Debug....4499
75.5.1 Test and Debug Access Port (TAP) connectivity.... 4499
75.5.2 Debug Access Port (DAP) TAP....4499
75.5.3 System JTAG Controller (JTAGC).... 4500
75.5.3.1 Chip JTAG ID....4502

75.5.3.2 JTAG Data Communication (JDC) module....4502
75.6 APB memory map....4502
75.6.1 S32G accelerator debug APB memory map....4510
75.7 Trace....4512
75.7.1 Trace modules and connectivity....4512
75.7.1.1 S32G accelerator trace modules and connectivity....4513
75.7.2 Trace assignments....4514
75.7.2.1 S32G accelerator trace assignments....4515
75.8 Embedded Cross Trigger (ECT)....4516
75.8.1 CTI assignments....4516
75.8.1.1 S32G accelerator subsystem CTI assignments....4520
75.9 NoC trace....4523
75.9.1 S32G accelerator subsystem NoC trace....4529
75.9.2 Observability....4531
75.9.2.1 Concepts....4531
75.9.2.1.1 Packet and transaction probes....4531
75.9.2.1.2 Observers....4532
75.9.2.2 Operation....4532
75.9.2.2.1 Logging errors....4532
75.9.2.2.2 Tracing packets....4533
75.9.2.2.3 Collecting statistics....4536
75.9.2.2.4 Transaction profiling operation....4537
75.9.2.2.5 Activating observer debug output....4538
75.9.2.3 Technical reference....4538
75.9.2.3.1 Observation probes....4538
75.9.2.3.2 Observers....4539
75.10 Debug and trace under functional reset....4547
75.11 MDM_AP register descriptions....4548
75.11.1 MDM_AP memory map....4549
75.11.2 MDM_AP status (STATUS)....4549
75.11.3 MDM_AP control (CONTROL)....4552
75.11.4 WIR enable (WIR_EN)....4557
75.11.5 WIR release (WIR_REL)....4557
75.11.6 MDM_AP status 2 (STATUS2)....4558
75.11.7 Cortex-A53 A0 debug (CORTEX_A53_A0_IP_DBG)....4560
75.11.8 MDM_AP control 2 (CONTROL2)....4561
75.11.9 Cortex-A53 A1 debug (CA53_A1_IP_DBG)....4562
75.11.10 Cortex-A53 B0 debug (CORTEX_A53_B0_IP_DBG)....4563
75.11.11 Cortex-A53 B1 debug (CORTEX_A53_B1_IP_DBG)....4564
75.11.12 Cortex M7_0 debug (CORTEX_M7_0_IP_DBG)....4564
75.11.13 Cortex M7_1 debug (CORTEX_M7_1_IP_DBG)....4565
75.11.14 Cortex M7_2 debug (CORTEX_M7_2_IP_DBG)....4566
75.11.15 DAP enable control (DAP_EN_CTRL)....4567
75.12 Debug resets....4568
75.13 Security....4568
75.13.1 Arm authentication signals....4569
75.13.2 Security fuses....4569
75.14 Programming checklists....4569
75.14.1 JTAG switching sequence....4569
75.14.2 FCCU considerations....4569
75.14.3 Lockstep considerations....4569
75.15 Pin interface....4569
75.15.1 Debug port pin descriptions....4570
75.15.2 Trace port pin descriptions....4570
75.16 Timestamp distribution network....4571

75.16.1 S32G accelerator timestamp distribution network....4572
75.17 Low power debug....4572
75.18 References....4573

Chapter 76 JTAG Controller (JTAGC)....4574
76.1 Chip-specific JTAGC information....4574
76.1.1 JTAGC device identification register....4574
76.2 Introduction....4574
76.2.1 Block diagram....4574
76.2.2 Features....4574
76.2.3 Modes of operation....4575
76.2.3.1 Reset....4575
76.2.3.2 IEEE 1149.1–2001 defined test modes....4575
76.2.3.3 Bypass mode....4575
76.3 External signal description....4575
76.3.1 Test clock input (TCK)....4576
76.3.2 Test data input (TDI)....4576
76.3.3 Test data output (TDO)....4576
76.3.4 Test mode select (TMS)....4576
76.3.5 JCOMP JTAG compliancy....4576
76.4 Register description....4576
76.4.1 Instruction register....4576
76.4.2 Bypass register....4577
76.4.3 Device identification register....4577
76.4.4 Boundary scan register....4578
76.5 Functional description....4578
76.5.1 JTAGC reset configuration....4578
76.5.2 IEEE 1149.1-2001 (JTAG) TAP....4578
76.5.3 TAP controller state machine....4579
76.5.3.1 Enabling the TAP controller....4579
76.5.3.2 Selecting an IEEE 1149.1-2001 register....4580
76.5.4 JTAGC block instructions....4580
76.5.4.1 IDCODE instruction....4580
76.5.4.2 SAMPLE/PRELOAD instruction....4580
76.5.4.3 SAMPLE instruction....4580
76.5.4.4 EXTEST external test instruction....4580
76.5.4.5 HIGHZ instruction....4581
76.5.4.6 CLAMP instruction....4581
76.5.4.7 ACCESS_AUX_x instructions....4581
76.5.4.8 BYPASS instruction....4581
76.5.5 Boundary scan....4581
76.6 Initialization/application information....4581

Chapter 77 JTAG Data Communication (JDC)....4582
77.1 Introduction....4582
77.2 Overview....4582
77.3 Signal description....4582
77.4 JDC register descriptions....4583
77.4.1 JDC memory map....4583
77.4.2 Module Configuration Register (MCR)....4583
77.4.3 Module Status Register (MSR)....4584
77.4.4 JTAG Output Data Register (JOUT_IPS)....4586
77.4.5 JTAG Input Data Register (JIN_IPS)....4586

77.5 Non-memory-mapped register definition.... 4587
77.5.1 JTAG output data register (JOUT).... 4587
77.5.2 JTAG input data register (JIN).... 4588
77.6 Functional description.... 4588
77.6.1 JTAG register access.... 4589
77.6.1.1 JDC block instructions.... 4589
77.7 Secure challenge/response connection procedure.... 4589
77.8 Glossary.... 4590

Chapter 78 Aurora Trace Port (ATP) Subsystem.... 4591
78.1 Introduction.... 4591
78.1.1 Features.... 4591
78.2 External signal description.... 4592
78.3 Overview.... 4592
78.4 ATP register descriptions.... 4593
78.4.1 ATP memory map.... 4593
78.4.2 PLL Control (PLLC).... 4594
78.4.3 PLL Status (PLLS).... 4595
78.4.4 PLL Divider (PLLDIV).... 4597
78.4.5 PLL Calibration 2 (PLLCAL_2).... 4598
78.4.6 PLL Clock MUX (PLLCKMUX).... 4599
78.4.7 AL Status (ALS).... 4600
78.4.8 AL General Control (ALGC).... 4602
78.4.9 AL Training Control (ALTC).... 4604
78.4.10 LVDS TX IO Configuration (LVDSTX).... 4607
78.4.11 LVDS RX IO Configuration (LVDSRX).... 4609
78.4.12 LVDS TX OBE Configuration (LVDSTXOBE).... 4610
78.4.13 CIA Control (CIAC).... 4611
78.4.14 Aurora Trace Port Enable (ATPE).... 4613
78.4.15 Integration Mode Control (ITCTRL).... 4614
78.4.16 Claim Tag Set (CLAIMSET).... 4614
78.4.17 Claim Tag Clear (CLAIMCLR).... 4615
78.4.18 Device Affinity 0 (DEVAFF0).... 4616
78.4.19 Device Affinity 1 (DEVAFF1).... 4616
78.4.20 Lock Access Register (LAR).... 4617
78.4.21 Lock Status Register (LSR).... 4618
78.4.22 Authentication Status (AUTHSTATUS).... 4619
78.4.23 Device Architecture (DEVARCH).... 4620
78.4.24 Device Configuration 2 (DEVID2).... 4621
78.4.25 Device Configuration 1 (DEVID1).... 4621
78.4.26 Device Configuration (DEVID).... 4622
78.4.27 Device Type Identifier (DEVTYPE).... 4623
78.4.28 Peripheral Identification Register 4 (PIDR4).... 4624
78.4.29 Peripheral Identification Register 0 (PIDR0).... 4624
78.4.30 Peripheral Identification Register 1 (PIDR1).... 4625
78.4.31 Peripheral Identification Register 2 (PIDR2).... 4626
78.4.32 Peripheral Identification Register 3 (PIDR3).... 4627
78.4.33 Component Identification Register 0 (CIDR0).... 4628
78.4.34 Component Identification Register 1 (CIDR1).... 4628
78.4.35 Component Identification Register 2 (CIDR2).... 4629
78.4.36 Component Identification Register 3 (CIDR3).... 4630
78.5 Functional description.... 4631
78.5.1 CoreSight interface adaptor.... 4631
78.5.1.1 CIA functional blocks.... 4631

78.5.1.2 TPIU interface module....4632
78.5.1.3 Buffer management module....4632
78.5.1.4 AL interface module....4632
78.5.1.5 Register configuration module....4632
78.5.1.6 Clock gating module....4632
78.5.2 Aurora link....4633
78.5.2.1 AL functional blocks....4633
78.5.2.2 AL link training....4634
78.5.2.3 AL static training....4634
78.5.2.4 AL transmission procedure....4635
78.5.2.5 AL clock compensation considerations....4635
78.5.3 Aurora PHY....4636
78.5.3.1 AP modes of operation....4636
78.5.3.2 AP functional blocks....4636
78.5.4 Aurora PLL....4637
78.5.4.1 PLL clock configuration....4637
78.6 Initialization sequence....4638

Appendix A Revision History....4641
A.1 About_This_Manual module changes....4641
A.2 Introduction changes....4641
A.3 Arm modules changes....4641
A.4 FlexNoC module changes....4641
A.5 Chip-specific Ncore information changes....4641
A.6 Ncore module changes....4641
A.7 MCM module changes....4642
A.8 A53_CLUSTER_GPR module changes....4642
A.9 CM7_CLUSTER_GPR module changes....4642
A.10 Chip-specific eDMA information changes....4642
A.11 eDMA3 module changes....4642
A.12 DMA_CRC module changes....4642
A.13 Chip-specific DMAMUX information changes....4642
A.14 DMAMUX module changes....4642
A.15 Chip-specific XRDC information changes....4643
A.16 XRDC module changes....4643
A.17 Application Extension-Specific SIUL2 Changes....4643
A.18 SIUL2 module changes....4643
A.19 SEMA42 module changes....4643
A.20 SRC module changes....4643
A.21 S32G_MAIN_GPR module changes....4644
A.22 S32G_STDBY_GPR module changes....4644
A.23 REG_PROT module changes....4644
A.24 Chip-specific [S32G] information changes....4645
A.25 WKPU module changes....4645
A.26 Clocking changes....4645
A.27 MC_CGM module changes....4646
A.28 DFS module changes....4646
A.29 PLLDIG module changes....4646
A.30 FXOSC module changes....4646
A.31 Reset changes....4647
A.32 Chip-specific MC_RGM information changes....4647
A.33 MC_RGM module changes....4647

A.34 Boot changes..... 4647
A.35 Power management changes..... 4648
A.36 Chip-specific MC_ME information changes..... 4648
A.37 MC_ME module changes..... 4649
A.38 MC_PCU changes..... 4649
A.39 Chip-specific PMC information changes..... 4649
A.40 PMC module changes..... 4649
A.41 Chip-specific [S32G] information changes..... 4649
A.42 SRAMC module changes..... 4649
A.43 DDR module changes..... 4649
A.44 DDR_GPR module changes..... 4650
A.45 Chip-specific QuadSPI changes..... 4650
A.46 QuadSPI module changes..... 4650
A.47 uSDHC module changes..... 4650
A.48 Chip-specific STM information changes..... 4651
A.49 STM module changes..... 4651
A.50 Chip-specific SWT information changes..... 4651
A.51 SWT module changes..... 4651
A.52 Chip-specific PIT information changes..... 4651
A.53 PIT_RTI module changes..... 4651
A.54 Chip-specific FTM information changes..... 4651
A.55 FTM module changes..... 4651
A.56 RTC module changes..... 4652
A.57 I^2C module changes..... 4652
A.58 Chip-specific GMAC changes..... 4652
A.59 PFE module changes..... 4652
A.60 Chip-specific FlexRay information changes..... 4652
A.61 FlexRay module changes..... 4652
A.62 LINFlexD module changes..... 4652
A.63 Chip-specific SPI information changes..... 4652
A.64 Chip-specific FlexCAN information changes..... 4652
A.65 FlexCAN module changes..... 4652
A.66 S32G CAN Timestamp changes..... 4653
A.67 Chip-specific SerDes information changes..... 4653
A.68 SERDES_GPR module changes..... 4653
A.69 LLCE module..... 4653
A.70 Chip-specific USBOTG changes..... 4653
A.71 USBOTG module changes..... 4653
A.72 Chip-specific CTU changes..... 4654
A.73 CTU module changes..... 4654
A.74 Chip-specific ADC information changes..... 4654
A.75 SAR_ADC module changes..... 4654
A.76 Chip-specific TMU information changes..... 4655
A.77 TMU changes..... 4655
A.78 Security changes..... 4655
A.79 Chip-specific MU information changes..... 4655
A.80 MU changes..... 4655
A.81 OCOTP Controller module changes..... 4655
A.82 FBXC module changes..... 4656
A.83 Safety changes..... 4656
A.84 Chip-specific ERM information changes..... 4656
A.85 ERM module changes..... 4656
A.86 FCCU chip-specific changes..... 4656
A.87 FCCU module changes..... 4656
A.88 Chip-specific CMU_FC information changes..... 4657

A.89 Chip-specific CMU_FM information changes....4657
A.90 SELFTEST_GPR module changes....4657
A.91 SELFTEST_GPR_TOP module changes....4657
A.92 Chip-specific STCU2 information changes....4657
A.93 STCU2 module changes....4658
A.94 SBSW module changes....4658
A.95 Chip-specific EIM information changes....4659
A.96 EIM changes....4659
A.97 CRC module changes....4659
A.98 Chip-specific JTAGC information changes....4659
A.99 JTAGC module changes....4659
A.100 CJTAG module changes....4659
A.101 ATP module changes....4659
A.102 JDC module changes....4659

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab62fc1

Chapter 1
About This Manual

1.1 Audience

This reference manual (RM) is intended for system software, hardware developers, and applications programmers who need to develop products using this chip. It assumes that its users understand operating systems, microprocessor system design, and basic principles of software and hardware.

1.2 Organization

This manual has two main sets of chapters.

- Chapters in the first set contain information that applies to all components on the chip.
- Chapters in the second set are organized into functional groupings that detail particular areas of functionality.
  - Examples of these groupings are clocking, timers, and communication interfaces.
  - Each grouping includes chapters that provide a technical description of individual modules.

1.2.1 Attachments

This manual includes key information in the files attached to it. For example, memory map and I/O details. Use the content in these attachments in conjunction with this manual's content.

NOTE

Select the paperclip icon on the left side of the PDF window to see the list of attachments.

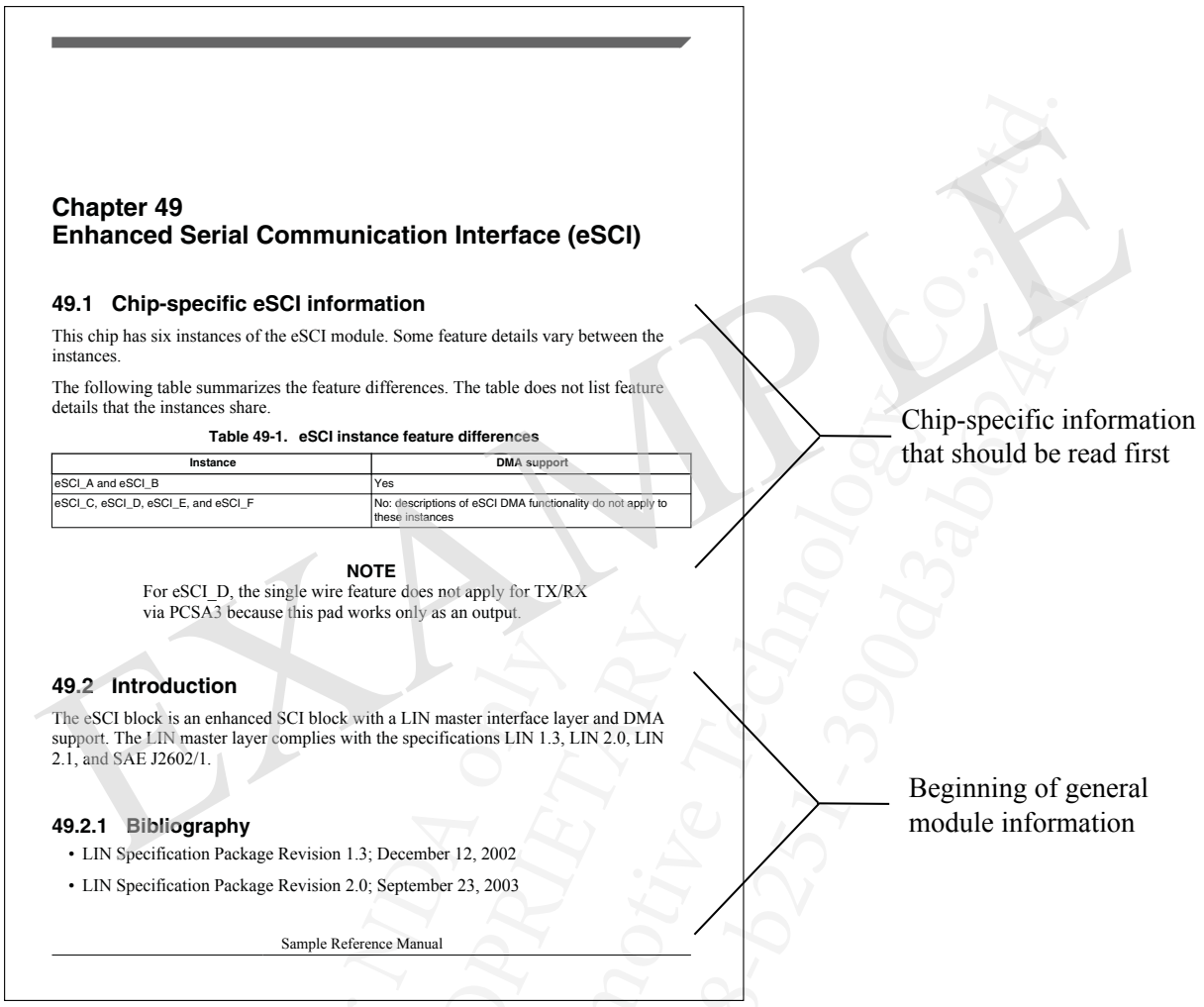
1.3 Module descriptions

Each module chapter has two main parts:

- The first section, *chip-specific [module name]* information, provides details such as the number of module instances on the chip and connections between that module and the other ones. Read this section *first* because its content is crucial for understanding the information in the other sections of the chapter.
- The subsequent sections provide general information about the module, including its signals, registers, and functional description.

The following figure shows you an example of this demarcation.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a-b251-390dRab624c1



Chapter 49
Enhanced Serial Communication Interface (eSCI)

49.1 Chip-specific eSCI information

This chip has six instances of the eSCI module. Some feature details vary between the instances.

The following table summarizes the feature differences. The table does not list feature details that the instances share.

Table 49-1. eSCI instance feature differences

Instance	DMA support
eSCI_A and eSCI_B	Yes
eSCI_C, eSCI_D, eSCI_E, and eSCI_F	No: descriptions of eSCI DMA functionality do not apply to these instances

NOTE
For eSCI_D, the single wire feature does not apply for TX/RX via PCSA3 because this pad works only as an output.

49.2 Introduction

The eSCI block is an enhanced SCI block with a LIN master interface layer and DMA support. The LIN master layer complies with the specifications LIN 1.3, LIN 2.0, LIN 2.1, and SAE J2602/1.

49.2.1 Bibliography

- LIN Specification Package Revision 1.3; December 12, 2002
- LIN Specification Package Revision 2.0; September 23, 2003

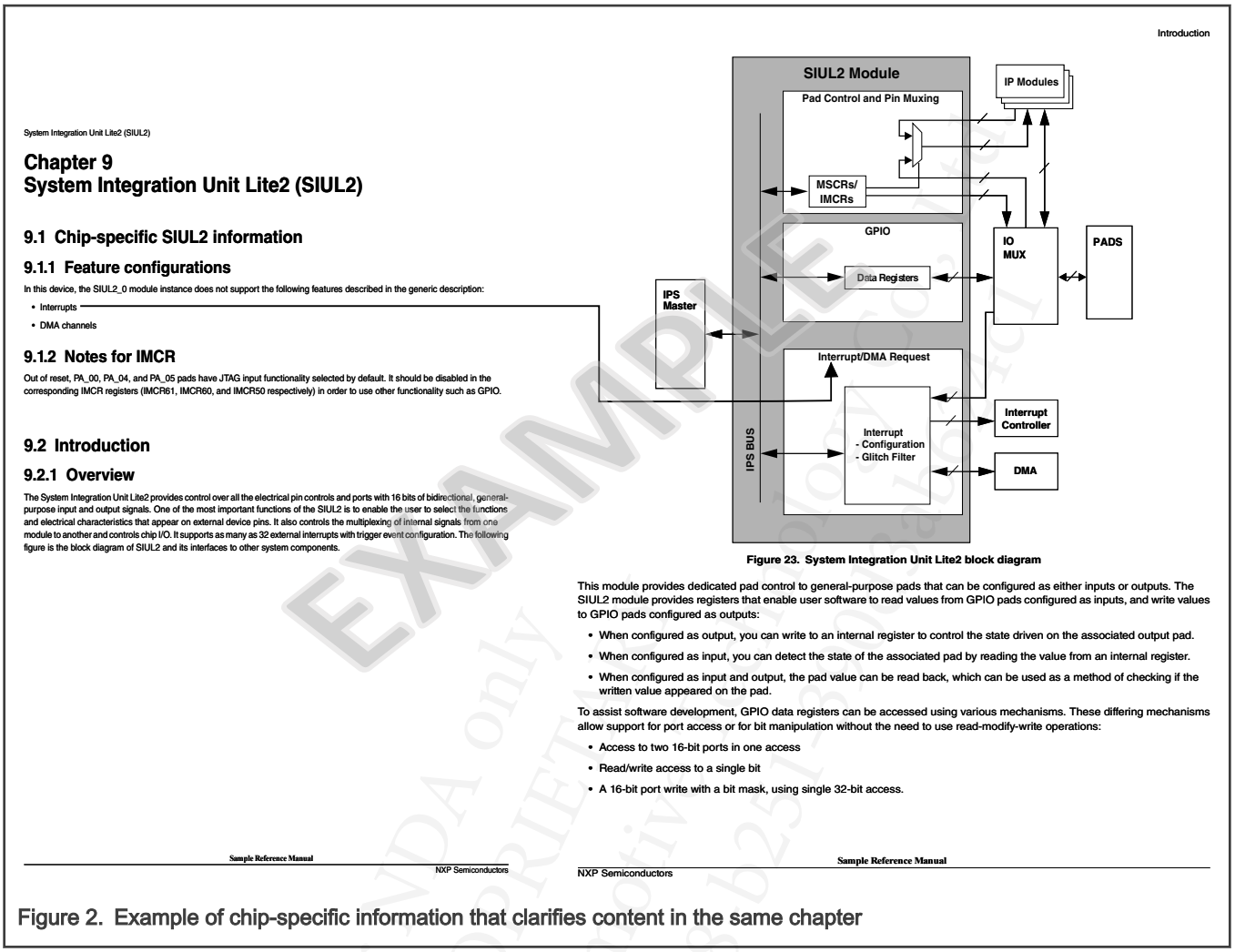
Sample Reference Manual

Figure 1. Example of chapter chip-specific information and general module information

1.3.1 Chip-specific information that clarifies content in the same chapter

The following figure shows an example of chip-specific information that clarifies general module information presented later in the chapter. In this case, the chip-specific register reset values supercede the reset values that appear in the register diagram.

EXAMPLE
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48e8-b251-390d3ab9e4c4



Introduction

System Integration Unit Lite2 (SIUL2)

Chapter 9
System Integration Unit Lite2 (SIUL2)

9.1 Chip-specific SIUL2 information

9.1.1 Feature configurations

In this device, the SIUL2_0 module instance does not support the following features described in the generic description:

- Interrupts
- DMA channels

9.1.2 Notes for IMCR

Out of reset, PA_00, PA_04, and PA_05 pads have JTAG input functionality selected by default. It should be disabled in the corresponding IMCR registers (IMCR61, IMCR60, and IMCR50 respectively) in order to use other functionality such as GPIO.

9.2 Introduction

9.2.1 Overview

The System Integration Unit Lite2 provides control over all the electrical pin controls and ports with 16 bits of bidirectional, general-purpose input and output signals. One of the most important functions of the SIUL2 is to enable the user to select the functions and electrical characteristics that appear on external device pins. It also controls the multiplexing of internal signals from one module to another and controls chip I/O. It supports as many as 32 external interrupts with trigger event configuration. The following figure is the block diagram of SIUL2 and its interfaces to other system components.

Figure 23. System Integration Unit Lite2 block diagram

This module provides dedicated pad control to general-purpose pads that can be configured as either inputs or outputs. The SIUL2 module provides registers that enable user software to read values from GPIO pads configured as inputs, and write values to GPIO pads configured as outputs:

- When configured as output, you can write to an internal register to control the state driven on the associated output pad.
- When configured as input, you can detect the state of the associated pad by reading the value from an internal register.
- When configured as input and output, the pad value can be read back, which can be used as a method of checking if the written value appeared on the pad.

To assist software development, GPIO data registers can be accessed using various mechanisms. These differing mechanisms allow support for port access or for bit manipulation without the need to use read-modify-write operations:

- Access to two 16-bit ports in one access
- Read/write access to a single bit
- A 16-bit port write with a bit mask, using single 32-bit access.

Sample Reference Manual
NXP Semiconductors

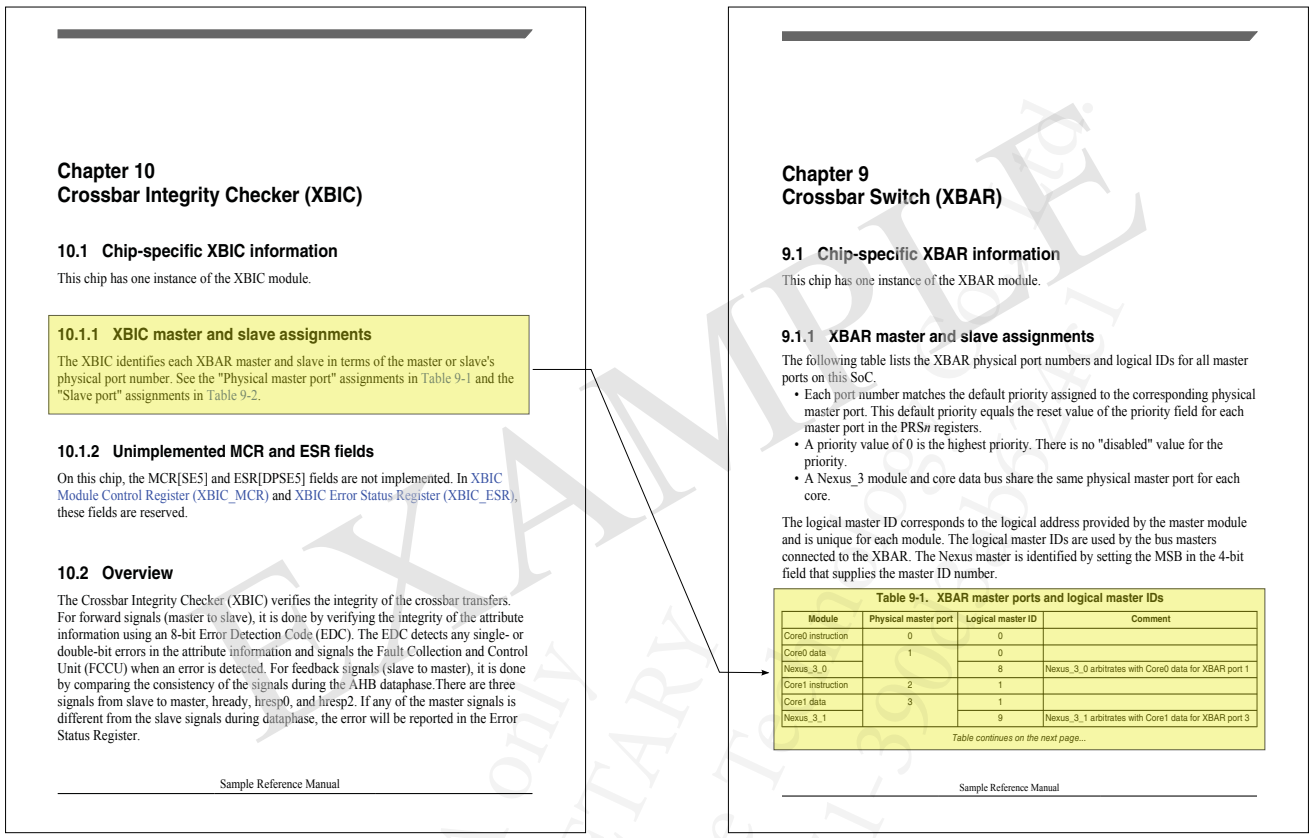
Sample Reference Manual
NXP Semiconductors

Figure 2. Example of chip-specific information that clarifies content in the same chapter

1.3.2 Chip-specific information that refers to a different chapter

Related chip-specific information may be provided in different chapters of the manual. The following figure shows an example of two such connected pieces of information. In this case, read both before you proceed.

EXAMPLE
Provided under NDA only
COMPANY CONFIDENTIAL
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-8a8b-b251-3b0d1a2c4d1



Chapter 10
Crossbar Integrity Checker (XBIC)

10.1 Chip-specific XBIC information

This chip has one instance of the XBIC module.

10.1.1 XBIC master and slave assignments

The XBIC identifies each XBAR master and slave in terms of the master or slave's physical port number. See the "Physical master port" assignments in Table 9-1 and the "Slave port" assignments in Table 9-2.

10.1.2 Unimplemented MCR and ESR fields

On this chip, the MCR[SE5] and ESR[DPSE5] fields are not implemented. In XBIC Module Control Register (XBIC_MCR) and XBIC Error Status Register (XBIC_ESR), these fields are reserved.

10.2 Overview

The Crossbar Integrity Checker (XBIC) verifies the integrity of the crossbar transfers. For forward signals (master to slave), it is done by verifying the integrity of the attribute information using an 8-bit Error Detection Code (EDC). The EDC detects any single- or double-bit errors in the attribute information and signals the Fault Collection and Control Unit (FCCU) when an error is detected. For feedback signals (slave to master), it is done by comparing the consistency of the signals during the AHB dataphase.There are three signals from slave to master, hready, hresp0, and hresp2. If any of the master signals is different from the slave signals during dataphase, the error will be reported in the Error Status Register.

Sample Reference Manual

Chapter 9
Crossbar Switch (XBAR)

9.1 Chip-specific XBAR information

This chip has one instance of the XBAR module.

9.1.1 XBAR master and slave assignments

The following table lists the XBAR physical port numbers and logical IDs for all master ports on this SoC.

- Each port number matches the default priority assigned to the corresponding physical master port. This default priority equals the reset value of the priority field for each master port in the PRS*n* registers.
- A priority value of 0 is the highest priority. There is no "disabled" value for the priority.
- A Nexus_3 module and core data bus share the same physical master port for each core.

The logical master ID corresponds to the logical address provided by the master module and is unique for each module. The logical master IDs are used by the bus masters connected to the XBAR. The Nexus master is identified by setting the MSB in the 4-bit field that supplies the master ID number.

Table 9-1. XBAR master ports and logical master IDs

Module	Physical master port	Logical master ID	Comment
Core0 instruction	0	0	
Core0 data	1	0	
Nexus_3_0		8	Nexus_3_0 arbitrates with Core0 data for XBAR port 1
Core1 instruction	2	1	
Core1 data	3	1	
Nexus_3_1		9	Nexus_3_1 arbitrates with Core1 data for XBAR port 3

Table continues on the next page...

Sample Reference Manual

Figure 3. Example of chip-specific information that refers to a different chapter

1.4 Register descriptions

Module chapters present register information in the following:

- Memory maps, which contain:
  - An offset from the module's base address
  - The mnemonic and name of each register
  - The width of each register (in bits)
  - The reset value of each register
- Register figures
- Field-description tables
- Associated text

The following figure shows register figure conventions used throughout the manual.

EXAMPLE
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-39c0b8b1

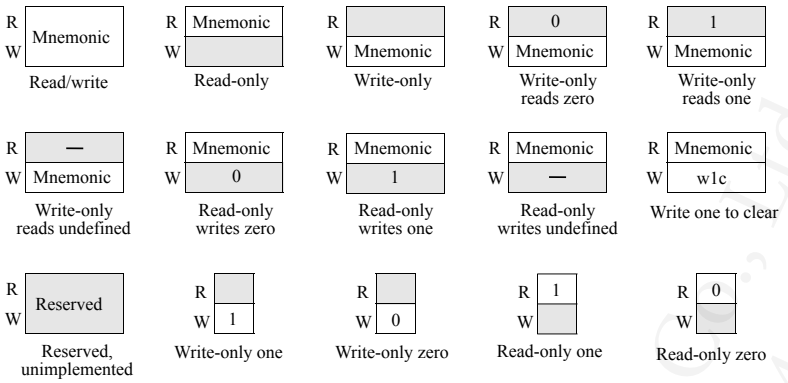


Figure 4. Register figure conventions

NOTE

Reset values of reserved locations documented in this manual are subject to change and must not be used for diagnostic purposes.

1.5 Conventions

1.5.1 Notes and cautions

Specific information is provided as part of notes and cautions throughout this manual.

NOTE

Emphasizes information that deserves extra attention.

CAUTION

Informs you of situations that could lead to highly undesirable outcomes—such as damage to the chip or irreversible malfunction.

1.5.2 Numbering systems

The following suffixes identify different numbering systems:

Table 1. Numbering systems

This suffix	Identifies a
b	Binary number. For example, the binary equivalent of the number 5 is mentioned as 101b. In some cases, *0b* is prefixed to binary numbers.
d	Decimal number. Decimal numbers are followed by this suffix only when there is a possibility of confusion. In general, decimal numbers are used without a suffix.
h	Hexadecimal number. For example, the hexadecimal equivalent of the number 60 is mentioned as 3Ch. In some cases, *0x* is prefixed to hexadecimal numbers.

1.5.3 Typographic notation

The following typographic notations are used throughout this document:

Table 2. Typographic notation

Example	Description
x and other italicized text	The italized, lowercase *x* is used as a placeholder for replaceable numbers. In general, italicized text is used for titles of publications and for emphasis. Additionally, italics could be used for metasymbols in syntax descriptions. Plain lowercase letters are used as placeholders for single letters and numbers.
`code font`	Fixed-width font (such as Courier) used for code. It is used for a letter, word, or phrase that you want the user to type. For example, "Type `Read` and press Enter." This type of font is also used for instruction mnemonics, directives, symbols, subcommands, parameters, operators, computer-language elements, code listings, commands that appear in running text, and for sample code. Instruction mnemonics and directives in text and tables are mentioned in all caps; for example, BSR.
SR[SCM]	A mnemonic in square brackets represents the name of a register field. This example refers to the Scaling Mode (SCM) field in the Status Register (SR).
REVNO[6:4], XAD[7:0]	Numbers in brackets that are separated by a colon represent either: • A subset of a register's named field For example, REVNO[6:4] refers to bits 6-4 that are part of the COREREV field occupying bits 6-0 of the REVNO register. • A continuous range of individual signals of a bus For example, XAD[7:0] refers to signals 7-0 of the XAD bus.
MOD.REG	A period between a module abbreviation and a register mnemonic indicates that the register is located in that module. For example, SIUL2.MSCRn means that the MSCRn register is located in the SIUL2 module.

1.5.4 Special terms

The following terms have special meanings.

Table 3. Special terms

Term	Meaning
Asserted	Refers to the state of a signal as follows: • An active-high signal is asserted when high (1). • An active-low signal is asserted when low (0).
Deasserted	Refers to the state of a signal as follows: • An active-high signal is deasserted when low (0). • An active-low signal is deasserted when high (1). In some cases, deasserted signals are described as *negated*.
Reserved	Refers to a memory space, register, field, or programming setting. Writes to a reserved location can result in unpredictable functionality or behavior. You must: • Not modify the default value of a reserved programming setting, such as the reset value of a reserved register field.

Table continues on the next page...

Table 3. Special terms (continued)

Term	Meaning
	• Consider undefined locations in memory to be reserved. • Not use the reset values of reserved locations documented in this manual for diagnostic purposes. They subject to change.
Write 1 to clear (w1c)	Refers to the access type of a register field that is used to clear the field by writing the value 1 to it.
Undefined (u)	Refers to undefined reset values

1.6 Editorial changes

Each new release of this document includes editorial improvements such as:

- Spelling
- Grammar
- Punctuation
- Voice
- Tense
- Capitalization
- Formatting
- Punctuation
- Presentation
- Navigation

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 2
Introduction

2.1 Overview

S32G2 is a family of high-performance vehicle network processors that combine controller area network (CAN), local interconnect network (LIN), and FlexRay networking with high-data-rate Ethernet networking. It also combines a functional safe-core infrastructure with MPU cores and includes high-level security features.

S32G2 family includes the following variants:

- S32G234M
- S32G233A
- S32G254A
- S32G274A

This document primarily represents the features offered by the superset S32G274A. To compare the features of the S32G2 family variants, please see the #unique_15 .

2.2 Applications

S32G2 targets the applications that support the following functions:

- Central gateways and domain controllers connecting various networks and translating their protocols
- Firmware over-the-air (FOTA) masters controlling secure software image downloads and their distribution to the engine control units (ECUs) in the network
- Secure key management
- Smart antennas
- High-performance central compute nodes

2.3 Block diagram

See the following figure that represents the block diagram for S32G2.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-43a8-b251-390d3b624c1

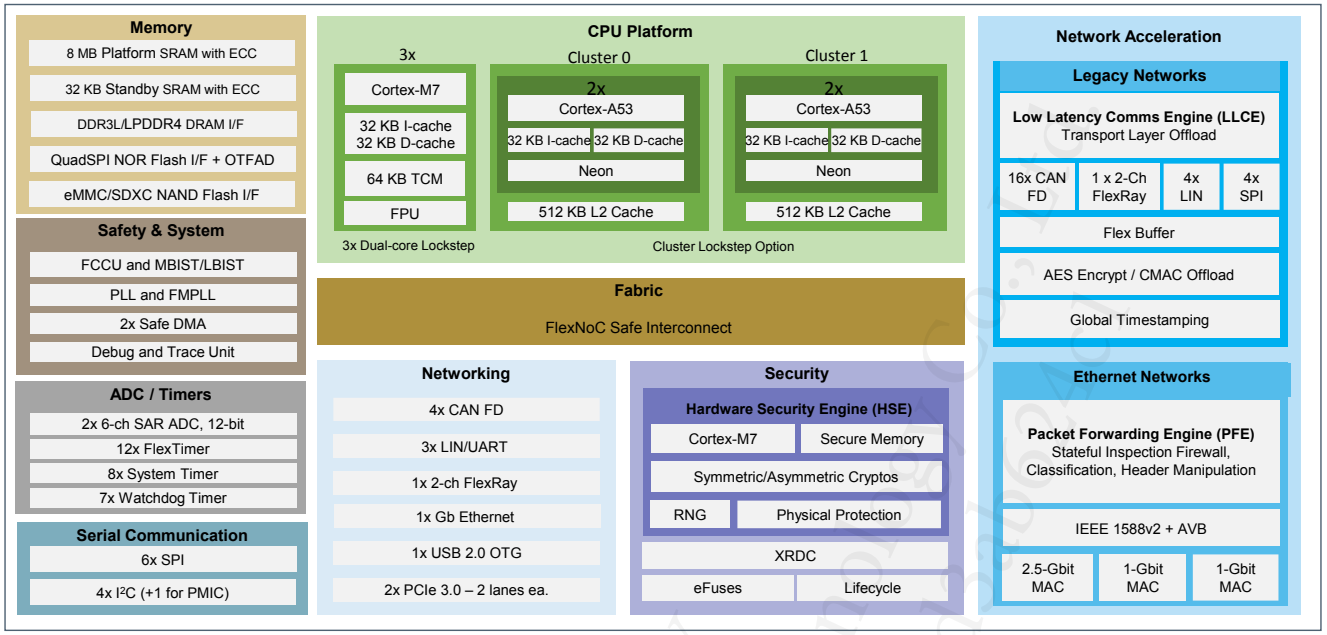


Figure 5. Block diagram

2.4 Module functional categories

The modules related to S32G2 on this device are grouped into functional categories.

NOTE

The module names in the following table may change in future releases.

Table 4. Module functional categories

Module categories	Module names
System	• Arm® Cortex-A53 • Arm Cortex-M7 • System NoC • Accelerator NoC • NCore • Enhanced Direct Memory Access (eDMA) • DMA CRC • Direct Memory Access Multiplexer (DMAMUX) • Extended Resource Domain Controller (XRDC) • System Integration Unit Lite2 (SIUL) • Semaphores2 (SEMA42) • Miscellaneous System Control Module (MSCM) • Wakeup Unit (WKPU)

Table continues on the next page...

Table 4. Module functional categories (continued)

Module categories	Module names
Clocking	• Clock Generation Module (MC_CGM) • Fast Internal RC Oscillator Digital Interface (FIRC) • Slow Internal RC Oscillator (SIRC) • Digital Frequency Synthesizer (DFS) • Phase-Locked Loop (PLL) for Aurora • Dual PLL Digital Interface (PLLDIG) • Fast Crystal Oscillator Digital Controller (FXOSC)
Reset and boot	• Reset Generation Module (MC_RGM)
Power management	• Mode Entry Module (MC_ME) • Power Control Unit (MC_PCU) • Power Management Controller (PMC) digital • Supply Presence Detector (SPD)
Memories and memory interfaces	• System RAM (SRAM) • DDR DRAM CTRL • DDR DRAM PHY • External flash memory interface • Quad Serial Peripheral Interface (QuadSPI) • On-the-Fly AES Decryption module (OTFAD) • Ultra Secured Digital Host Controller (uSDHC)
Timer	• Software Watchdog Timer (SWT) • System Timer Module (STM) • Periodic Interrupt Timer (PIT) • FlexTimer module (FlexTimer) • Real Time Clock (RTC)
Communication interface	• Inter-Integrated Circuit (I^2C) • Ethernet • FlexRay communication controller (FlexRay) • LINFlexD • CAN (FlexCAN) • De-serial Serial Peripheral Interface (DSPI) • Peripheral Component Interconnect Express (PCIe) controller • PCIe Gen3 PHY

Table continues on the next page...

Table 4. Module functional categories (continued)

Module categories	Module names
	• Low Latency Communications Engine (LLCE) • Ethernet Packet Forwarding Engine (PFE) • Universal Serial Bus (USB) • Universal Serial Bus OTG Controller (USBOTG)
Analog	• Analog-to-Digital Converter (SAR_ADC) • Cross Trigger Unit (CTU) • Temperature Monitor Unit (TMU) digital • TMU analog • TMU sense • Thermal diode • Thermal RESCAL circuit • Secure Digital (SD)
Security	• On-Chip One-Time Programmable (OCOTP) Controller • External Interface Module (EIM) • Register Protection (REG_PROT) • SRAM Controller (OCPSRAMC) • Hardware Security Engine (HSE_H)
Safety	• Error Reporting Module (ERM) • Fault Collection and Control Unit (FCCU) • Redundancy Control and Checker Unit (RCCU) • Clock Monitor Unit (CMU) • Self-Test Control Unit (STCU2) • Safety by Software (SBSW) • Error Injection Module (EIM) • Cyclic Redundancy Check (CRC)
Debug and trace	• JTAG Controller (JTAGC) • IEEE 1149.7 Compact JTAG Test Access Port Controller (CJTAG) • JTAG Data Controller (JDC) • Aurora Trace Port (ATP)

2.5 Feature comparison

The following table compares the features of these devices in the S32G2 family: S32G234M, S32G233A, S32G254A, and S32G274A.

Provided under NDA only
COMPANY PROPRIETARY
Beijing Huichuang Technology Co., Ltd.
cac46251-1621-4c38-8251-390d3ab624c1

Table 5. Feature comparison

Feature	S32G234M	S32G233A	S32G254A	S32G274A
Compute and bus modules				
CPUs: applications and processors	N/A	Cluster 0: Single Arm Cortex-A53		Cluster 0: Dual Arm Cortex-A53
	N/A	Cluster 1: Single Arm Cortex-A53		Cluster 1: Dual Arm Cortex-A53
L1 cache	N/A	32 KB I-cache/32 KB D-cache per Cortex-A53		
L2 cache	N/A	512 KB per cluster		
Cache coherency interconnect	N/A	Yes		
Interrupt controller supported by Cortex-A53	N/A	Generic interrupt controller (GIC)-500		
Maximum frequency of Cortex-A53	N/A	1 GHz		
Functional safety	N/A	Configurable: ASIL D (lockstep clusters) ASIL B (two independent clusters)		
CPUs: real-time	3x Arm Cortex-M7 (lockstep)	1x Arm Cortex-M7 (lockstep)	3x Arm Cortex-M7 (lockstep)	
L1 cache	32 KB I-cache/32 KB D-cache per Cortex-M7			
Data tightly-coupled memory (D-TCM)	64 KB per Cortex-M7			
Interrupt controller supported by Cortex-M7	3x nested vectored interrupt controller (NVIC)	1x NVIC	3x NVIC	
Maximum frequency of Cortex-M7	400 MHz			
Direct memory access (DMA)	2x Safe eDMA (supporting lockstep) with 32 channels per eDMA			
DMAMUX	Supporting 128 inputs per DMA			
Debug: Run control	Arm CoreSight JTAG (IEEE 1149.1)			
Debug: Trace	Aurora 4-lane			
SWT	7			
STM	8			
Memory modules				
Internal RAM	8 MB	6 MB	8 MB	
RAM ports	16 ports, interleaved 64-byte			

Table continues on the next page...

Table 5. Feature comparison (continued)

Feature	S32G234M	S32G233A	S32G254A	S32G274A
Dynamic random-access memory (DRAM)	N/A	LPDDR4 and DDR3L		
DRAM physical interface (PHY)	N/A	x32		
QuadSPI	1 (supports two identical devices)			
Secured digital host controller (SDHC)	1			
Fuses	8K bank			
Standby SRAM with error correction code (ECC)	32 KB			
Security modules				
Security subsystem	Hardware Security Engine (HSE_H)			
Resource isolation	XRDC supporting 8 domains, 32 entries			
Arm TrustZone (TZ)	N/A	Yes		
Lifecycle	Yes			
Secure debug	Yes			
OTFAD	Yes			
Communication interface modules				
Comms acceleration	Low Latency Communications Engine (LLCE)			
CAN/CAN with flexible data-rate (CAN FD)	16 (LLCE) + 4			
FlexRay 2.1 (dual channel)	1 (LLCE) + 1			
LIN (UART)	4 (LLCE) + 3			
Ethernet acceleration	Packet Forwarding Engine (PFE)			
Ethernet MACs	4			
Ethernet I/F	MII / RMII / RGMII / SGMII			
PCI express (PCIe)	1x Gen 2 controller (X1, X2 modes) (PCIe_1 only)	2x Gen 3 controller (X1, X2 modes)		
SERDES	2-Lanes (configurable PCIe/SGMII) (SerDes_1 only)	4-Lanes (configurable PCIe/SGMII)		
USB 2.0 on-the-go (OTG) (ULPI I/F)	N/A	1		

Table continues on the next page...

Table 5. Feature comparison (continued)

Feature	S32G234M	S32G233A	S32G254A	S32G274A
I^2C	4 + 1 power management integrated circuits (PMIC)			
Serial peripheral interface (SPI)	4 (LLCE) + 6			
CRC	1			
Generic modules				
PIT	2			
SAR_ADC	2 x 6 channels, 12-bit			
FlexTimer module (FTM)	2 x 6 channels			
CTU	1			
SEMA42	1			
Clocking, power, and reset modules				
FIRC	48 MHz			
SIRC	32 kHz			
FXOSC	20–40 MHz			
PLL	5			
Low-power mode	Yes			
Real time clock (RTC)/application programming interface (API)	Yes			
Wake-up	24 wake-up sources			
Miscellaneous				
Package	525 flip chip plastic ball grid array (525 FC-PBGA), 19mm x19mm, 0.8mm			

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a8-b251-390d8ab624c1

Chapter 3
Memory Map

3.1 Introduction

See the device Memory map tables attached to this document as an Excel file. Locate the paperclip symbol on the left side of the PDF window, and click on it.

NOTE

Accesses to reserved space is not allowed and will result in indeterminate results.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 4
Signal Description

4.1 IOMUX and signal details

See the IOMUX spreadsheet attached to this document. Locate the paperclip symbol on the left side of the PDF window, and click it. Double-click on the Excel file to open it.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 5
Arm Modules

5.1 Core complex platform components

The following table lists the platform components which comprise the core complex.

Table 6. Core complex platform components

Component	Description	Reference
Arm Cortex-A53 MPCore clusters complex	• Two Arm Cortex-A53 clusters, two cores per cluster, up to 1 GHz • NoC Interconnect fabric • Ncore Cache Coherent Interconnect	• Overview: Cortex-A53 cluster complex • Cortex-A53 implementation details: Arm Cortex-A53 MPCore • Cortex-A53 documentation: *Arm Cortex-A53 MPCore Processor Technical Reference Manual, Revision r0p4* at www.arm.com
Arm® Cortex®-M7 processor clusters	• Three Arm Cortex-M7 delayed lockstep clusters, two cores per cluster, up to 400 Mhz • An in-order issue, super-scalar pipeline with dynamic branch prediction	• Overview: Cortex-M7 Core Introduction • Cortex-M7 documentation: *Arm Cortex-M7 Processor Technical Reference Manual, DDI 0489D Revision r1p1* at www.arm.com
Arm CoreLink Generic Interrupt Controller (GIC-500)	• The Arm GIC architecture divides interrupt management and processing among a distributor block and one or more CPU interface blocks. • The GIC-500 includes a register set for managing interrupt sources, interrupt behavior, and interrupt routing.	• Overview: Arm CoreLink Generic Interrupt Controller (GIC-500) • *Arm CoreLink GIC-500 Generic Interrupt Controller Technical Reference Manual* at infocenter.arm.com
Device debug and trace interfaces	Based on Arm CoreSight™ architecture supplemented with the Secured JTAG controller (SJC) to support security features	• Arm debug documentation: *Arm Debug Interface Architecture Specification ADIv5.0 to ADIv5.2* and *Arm CoreSight Architecture Specification* at www.arm.com

5.2 Cortex-A53 cluster complex

The Arm Cortex-A53 complex provides a high-performance host processing environment. It also supports lower power run modes on the chip through frequency scaling and power gating on unused non-primary cores.

The GIC-500 module provides interrupt management and routing outside of the processors.

5.2.1 Arm Cortex-A53 MPCore

The Arm® Cortex®-A53 cluster is a mid-range, low-power Arm processor that implements the Armv8-A architecture. Each chip includes two identical Arm Cortex-A53 clusters. The implementation options for each chip are shown below.

Preliminary Information COMPANY CONFIDENTIAL Beijing HJ Automotive Technology Co., Ltd. c8c62e8-1821-48a8-b251-5903e624c1

Table 7. Arm Cortex-A53 cluster configuration by chip

Implementation option	Quad	
	Cluster 0	Cluster 1
Number of cores	2	
L1 data cache size	• 32 KB 2-way set associative instruction cache • 32 KB 4-way set associative data cache	
L2 cache	Included	
L2 cache size	512 KB	
SCU-L2 cache protection	Included	
Advanced SIMD and floating-point extension[1]	Included	
Cryptography extension	Included	
CPU cache protection	Included	
AMBA 5 CHI or AMBA 4 ACE interface	AMBA 4 ACE	
v7 or v8 debug memory map	v8	

1. Also, referred to as FPU and Neon extension

Additionally, the standard features include:

- A Snoop Control Unit (SCU) to maintain L1 cache coherency among the cores
- A generic timer to schedule events and trigger interrupts based on an incrementing counter value. It provides generation of timer events as interrupt outputs and generation of event streams.
- Instruction tracing capability through Embedded Trace Macrocell
- A cross-trigger module

Each core includes:

- Arm v8-A instruction set architecture and both AArch32 and AArch64 execution states
- Asymmetric/symmetric multi-processing capability
- A Floating Point Unit (FPU) implemented with the Arm Vector Floating Point v3 architecture with 16 double-precision registers (VFPv3-D16), compliant with the ANSI/IEEE Std 754-1985 and IEEE Standard for Binary Floating-Point Arithmetic (IEEE 754)
- A Cortex-A53 Neon Media Processing Engine (MPE), which extends the capabilities of the FPU with an implementation of the Arm Neon Advanced SIMD instruction set for further acceleration of media and signal processing functions. It includes an additional register set supporting a rich set of SIMD operations over 8, 16, and 32-bit integer and 32-bit floating-point data types.
- In-order 8-stage dual issue pipeline implementation
- 512-KB L2 cache with ECC protection including software control for L2 cache flush.
- MMU that provides fine-grained memory system control through a set of virtual-to-physical address mappings and memory attributes. For example, cacheability and shareability, held in page tables.
- Debug resources for supporting trace and cross-trigger functions[1]
- A set of timer registers

[1] The device debug and trace is based on the Arm CoreSight™ architecture. For more information on the CoreSight Debug components, see the Arm® CoreSight™ Technical Reference Manual.

- A Generic Interrupt Controller (GIC) CPU interface, in conjunction with the external (to the clusters) GIC-500 Generic Interrupt Controller, acts as a centralized resource for supporting and managing interrupts.
  - Includes registers for managing interrupt sources, interrupt behavior, and interrupt routing to the Cortex-A53
  - Supports the Arm architecture security extensions
  - Supports enabling, disabling, and generating cluster interrupts from hardware (peripheral) interrupt sources
  - Supports generation of software interrupts
  - Supports interrupt masking and prioritization
  - Supports virtualization of interrupts for each Cortex-A53 CPU
  - Supports Interrupt Translation Service (ITS) that provides ID translation and core migration for message-based interrupts

The Cortex-A53 supports cache protection in the form of ECC or parity on all the RAM instances:

- L2 Cache Tag Array is protected with the Single-Bit Error Correction, Double-bit Error Detection (SECDED) capability through ECC protection. On a single-bit upset, the tag is rewritten with the corrected data and the access is retried. On a multi-bit upset, the tag is invalidated. The event is reported in the L2MERRSR register. Uncorrectable errors in the L2 Cache Tag Array cause the nINTERRIRQ interrupt line to be asserted.
- L2 Cache Data Array is protected with the SECDED capability through ECC protection on L2 Cache Tag and Data arrays. On a single-bit upset, the data is rewritten in-line. On a multi-bit upset, the tag is invalidated. The event is reported in the L2MERRSR register. Uncorrectable errors in the L2 Data Array because of an instruction fetch, TLB pagewalk, or load instruction result in either a precise data abort or a prefetch abort, and the event is reported to the L2MERRSR register. Uncorrectable errors in the L2 Data Array because of cache eviction cause the nINTERRIRQ interrupt line to be asserted. The event is reported in the L2MERRSR register.
- L1 Instruction Cache Tag arrays are protected with the SED capability through the parity coverage. In the event of a parity mismatch, both lines in the cache set are invalidated, then the line requested is refetched. The event is reported to the CPUMERRSR register.
- L1 Data Cache Tag arrays are protected with the Single-Error Detection (SED) capability through the parity coverage. In the event of a parity mismatch, the line is cleaned then evicted from the L1. The correct address for eviction is retrieved from the SCU duplicate tags. The event is reported to the CPUMERRSR register.
- L1 Instruction Cache Data arrays are protected with SED capability through the parity coverage. In the event of a parity mismatch, both lines in the cache set are invalidated, then the line requested is refetched. The event is reported to the CPUMERRSR register.
- L1 Data Cache Data arrays are protected with SECDED capability through ECC protection. On a single-bit upset, the data is rewritten in-line and the line is invalidated from the L1 cache. The event is reported to the CPUMERRSR register. On a multi-bit upset, the line is invalidated from the L1 cache and either causes an abort or the nINTERRIRQ interrupt line to be asserted. The event is reported to the CPUMERRSR register.

NOTE

nINTERRIRQ is mapped as an SPI in interrupt controllers and also reported as an NCF in FCCU.

5.2.2 Arm CoreLink Generic Interrupt Controller (GIC-500)

The Arm GIC architecture divides interrupt management and processing among a distributor block and one or more CPU interface blocks. The GIC-500 module is the distributor block. It provides central interrupt management and interrupt routing for both the clusters. Each processor core has a GIC CPU interface, which enables GIC-500 to route interrupts to either individual or multiple cores, depending on the interrupt type.

Freescale Semiconductor, Inc.
COMPANY PROPRIETARY
Beijing CH Electronics Technology Co., Ltd.
c8c2e5-5-821485-2-1-3-3-624-1

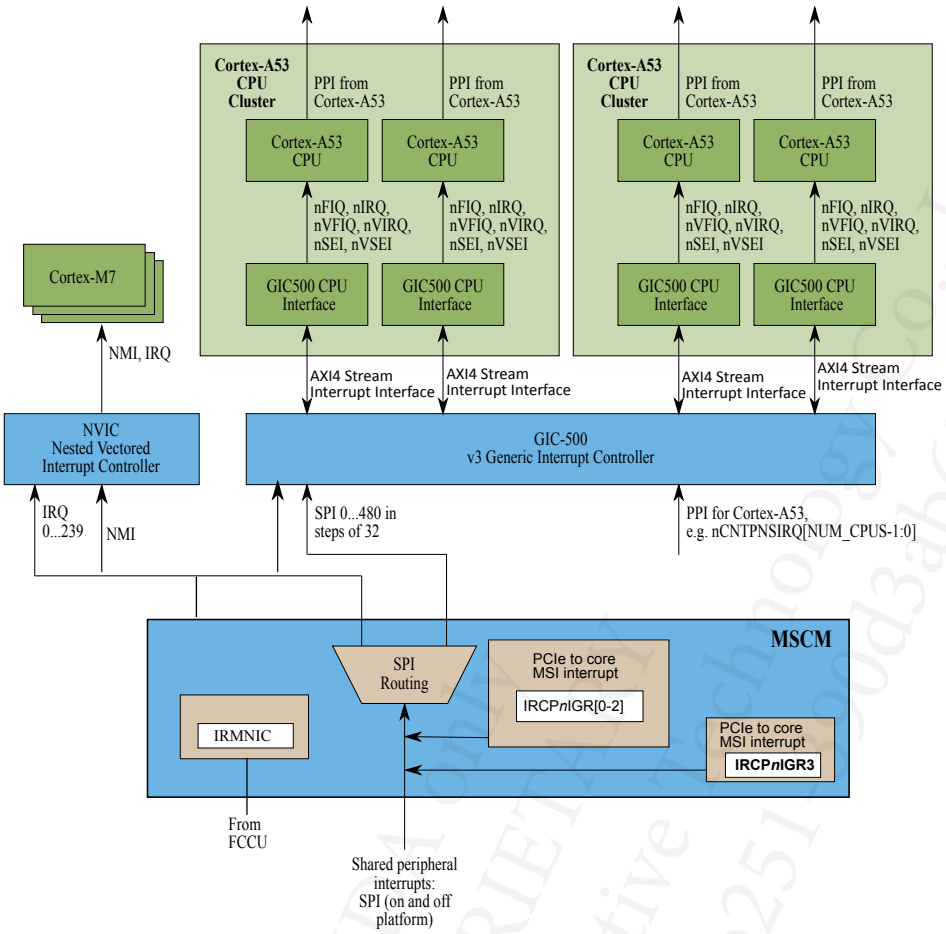


Figure 6. Interrupt architecture

The GIC-500 module includes a register set for managing interrupt sources, interrupt behavior, and interrupt routing. Support is included for:

- Enabling, disabling, and generating processor interrupts from the hardware (peripheral) interrupt sources
- 16 software-generated interrupts (SGIs)
- Interrupt masking and prioritization
- Uniprocessor and multiprocessor environments
- Wakeup events in power-management environments
- Locality-specific peripheral interrupts (LPIs)
- Shared peripheral interrupts (SPIs)
- 16 private peripheral interrupts (PPIs)

Provided under NXP ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e5-1891-48a8-b251-90d3ab624c1

NOTE

- Only the system register access method is supported (that is, by programming SRE = 1) for accessing GIC CPU Interface registers and memory mapped access is not supported. The Technical Reference Manual (TRM) provides more details on the access method of GIC CPU interface registers.
  - Cortex-A53 Technical Reference Manual

    http://infocenter.arm.com/help/topic/com.arm.doc.ddi0500f/DDI0500F_cortex_a53_r0p4_trm.pdf
- The operating mode of GIC-500 modules (lockstep/degraded) can be selected using the A53 GPR06[GIC500_LOCKSTEP_EN] register field.
- Error injection in GIC500 memories might result in the lockstep failure reported to FCCU if the two GIC500 modules are operating in the lockstep mode.

Table 8. GIC registers reset

Offset	Name	Type	Reset	Description
0x000	GICD_CTLR	RW	0x00000030	Distributor Control Register
0x004	GICD_TYPER	RO	0x007B0471	Interrupt Controller Type Register
0x008	GICD_IIDR	RO	0x0001143B	Distributor Implementer Identification Register
0x084 - 0xC4	GICD_IGROUPR0-17	RW	0x00000000	Interrupt Group Registers
0x104- 0x148	GICD_ISENABLER1-17	RW	0x00000000	Interrupt Set-Enable Registers
0x184- 0x1C8	GICD0_ICENABLER17	RW	0x00000000	Interrupt Clear-Enable Registers
0x204-0x248	GICD_ISPENDR1-17	RW	0x00000000	Interrupt Set-Pending Registers
0x284-0x2C8	GICD_ICPENDR1-17	RW	0x00000000	Interrupt Set-Pending Registers
0x304-0x348	GICD_ISACTIVERnd0-17	RW	0x00000000	GICv2 Interrupt Set-Active Registers
0x384-0x3C8	GICD_ICACTIVER0-17	RW	0x00000000	Interrupt Clear-Active Registers
0x420-0x63C	GICD_IPRIORITYR8-143	RW	0x00000000	Interrupt Priority Registers
0xC08-C8C	GICD_ICFGR2-35	RW	0x00000000	Interrupt Configuration Registers
0x6100-0x7EF8	GICD_IROUTER32-991	RW	0x00000000	SGI Clear-Pending Registers
0xFFE0	GICD_PIDR0	RO	0x00000092	Identification Register 0-4
0xFFE4	GICD_PIDR1	RO	0x000000B4	Identification Register 0-4

Table continues on the next page...

Table 8. GIC registers reset (continued)

Offset	Name	Type	Reset	Description
0xFFE8	GICD_PIDR2	RO	0x0000003B	Identification Register 0-4
0xFFEC	GICD_PIDR3	RO	0x00000000	Identification Register 0-4
0xFFD0	GICD_PIDR4	RO	0x00000044	Identification Register 0-4
0xFFF0	GICD_CIDR0	RO	0x0000000D	Component Identification Register 0-3
0xFFF4	GICD_CIDR1	RO	0x000000F0	Component Identification Register 0-3
0xFFF8	GICD_CIDR2	RO	0x00000005	Component Identification Register 0-3
0xFFFC	GICD_CIDR3	RO	0x000000B1	Component Identification Register 0-3

5.3 Operation details

5.3.1 Operation mode

The chip supports two modes of operation for Cortex-A53 clusters.

- **Decoupled mode**: The decoupled or performance mode allows the two Cortex-A53 clusters to operate in an independent manner. All the four cores are available for the software application.
- **Lock step mode**: In this mode, the two Cortex-A53 clusters operate in delayed lockstep configuration, that is, core0 of cluster0 is in lock step with core0 of cluster1 and core1 of cluster0 is in lock step with core1 of cluster1. Only cluster0 is available in the system for software application while cluster1 works as a checker or redundant module. The clock, reset, or other configurations of cluster1 and its cores, as available in MC_RGM, MC_ME, and A53 GPR registers, are not used in this mode.

The default mode of operation for the Cortex-A53 clusters is the decoupled mode. The selection between lockstep and decoupled mode is controlled by the A53 GPR06[CA53_LOCKSTEP_EN] register bit.

NOTE

The operating mode should be changed only when A53 reset domain is in the reset state.

5.3.2 Booting Cortex-A53 cores in 32-bit mode

The default boot mode of Cortex-A53 core is AArch64. Booting in AArch32 mode shall be done through the exception level in software, that is, by moving from the higher exception level to lower exception level.

5.3.3 Cluster clock gating

For information about turning on the Cortex-M7 application cores or the Cortex-A53 cluster cores, see the section "Core turn-on sequence" in the Reset chapter of this document. The Technical Reference Manual (TRM) provides more detail about core shutdown procedures.

- Cortex-A53 Technical Reference Manual

  http://infocenter.arm.com/help/topic/com.arm.doc.ddi0500f/DDI0500F_cortex_a53_r0p4_trm.pdf

- Cortex-M7 Technical Reference Manual

  http://infocenter.arm.com/help/topic/com.arm.doc.ddi0489b/DDI0489B_cortex_m7_trm.pdf

5.3.4 Cortex-A53 system counter

The A53 system counter value is driven from a 64-bit counter generated of a divided clock of FXOSC_CLK. The divide factor is controlled by GPR00[CA53_COUNTER_CLK_DIV_VAL]. The divide value equals to GPR00[CA53_COUNTER_CLK_DIV_VAL]+1.

The system counters get reset along with Cortex-A53 clusters whenever A53 reset domain is reset following the sequence described in section "Software reset partition turn-off flowchart" of Reset chapter.

5.3.5 Cortex-A53 system error interrupt

NMIs from FCCU and WKPU are routed as system error interrupts to the Cortex-A53 cores. The reporting of the NMIs to each of the four cores can be enabled through the IRNMIC register inside MSCM.

5.3.6 Cortex-A53 wake-up from WFE

Executing the SEV instruction from any of the Cortex-A53 core generates a wake-up event for all the Cortex-A53 cores available in the chip. Cortex-M7 cores can also generate a wake-up event for Cortex-A53 cores in a cluster using GPR01 in A53_GPR (bit 0 – cluster0 and bit 1 – cluster1). The GPR mechanism is not used by the Cortex-A53 cores and executing SEV instruction suffices. GPR01[bit 1] is not used when Cortex-A53 clusters are operating in the lockstep mode as only cluster0 is operational effectively.

5.3.7 Cortex-A53 PMUEVENT bus signals observation

The chip contains observers to capture and count the number of times events are signaled on PMUEVENT bus of Cortex-A53 cores. This information can be particularly useful to Cortex-M7 cores for debug or safety application. There is one PMUEVENT bus observer per Cortex-A53 core available in the chip. There are 30 counters present inside a PMUEVENT bus observer, one counter per bus signal. Each counter has a control register and a status register.

Base address of PMUEVENT observer

- A53 cluster0 core0 PMUEVENT bus observer - 0x4004_8100
- A53 cluster0 core1 PMUEVENT bus observer - 0x4004_8200
- A53 cluster1 core0 PMUEVENT bus observer - 0x4004_8300
- A53 cluster1 core1 PMUEVENT bus observer - 0x4004_8400

Procedure to read event count on PMUEVENT bus signals

The software should follow the following procedure to enable the counter and read the count of a number of times an event is signaled on a PMUEVENT bus signal "a" of a Cortex-A53 core. Here, "a" ranges from 0 to 29.

1. Program fields of PMUEVENT Observer Control Register (CTLa) to values mentioned below.
   - REVTINPSEL = 2'b01
   - CTRSTRTSEL = 2'b01
   - CTRSTPSEL = 2'b10
   - CPYCOUNT = 1'b1
   - CLRCNTR = 1'b0

2. On a PMU interrupt or FCCU interrupt (due to memory errors related PMUEVENTs, see FCCU NCF list list attached to this document), read the count value in the status register - PMUEVENT Observer Status Register (STATa).
3. Program CTLa[CLRCNTR] to 1'b1 in order to clear the counter and the status register.

5.3.7.1 PMUEVENT observer register descriptions

NOTE

For more information on the base address of PMUEVENT observer, see the section base address of PMUEVENT observer.

5.3.7.1.1 PMUEVENTOBSERVER memory map

PMUEVENTOBSERVER base address: 0h

Offset	Register	Width (In bits)	Access	Reset value
0h - 74h	PMUEVENT Observer Control Register (CTL0 - CTL29)	32	RW	0000_000Ch
80h - F4h	PMUEVENT Observer Status Register (STAT0 - STAT29)	32	RW	0000_0000h

5.3.7.1.2 PMUEVENT Observer Control Register (CTL0 - CTL29)

Offset

For a = 0 to 29:

Register	Offset
CTLa	0h + (a × 4h)

Function

This register controls PMUEVENT observer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								Reserved	Reserved						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						CLRCNTR	CPYCOUNT	CTRSTPSEL		CTRSTRTSEL		Reserved		REVTINPSEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c84e2e5-4821-48a8-b251-3903a2624c1

Fields

Field	Function
31-24 —	Reserved
23 —	Reserved
22-10 —	Reserved
9 CLRCNTR	This bit is used to clear the counter value. 0b - Do not clear the counter value. 1b - Clear the counter value.
8 CPYCOUNT	This bit is used to copy the counter value to the status register for software to read. 0b - Do not copy the counter value to the status register 1b - Copy the counter value to the status register
7-6 CTRSTPSEL	Input to select the counter stop trigger 00b - Reserved 01b - Falling edge detect signal used as stop trigger 10b - Reserved 11b - Reserved
5-4 CTRSTRTSEL	Input to select the counter start trigger 00b - Reserved 01b - Input to select the counter start trigger 10b - Reserved 11b - Reserved
3-2 —	Reserved
1-0 REVTINPSEL	Rise event input select to counter 00b - Reserved 01b - Rise level of signal used as input to counter 10b - Reserved 11b - Reserved

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-b251-390c3ab624c1

5.3.7.1.3 PMUEVENT Observer Status Register (STAT0 - STAT29)

Offset

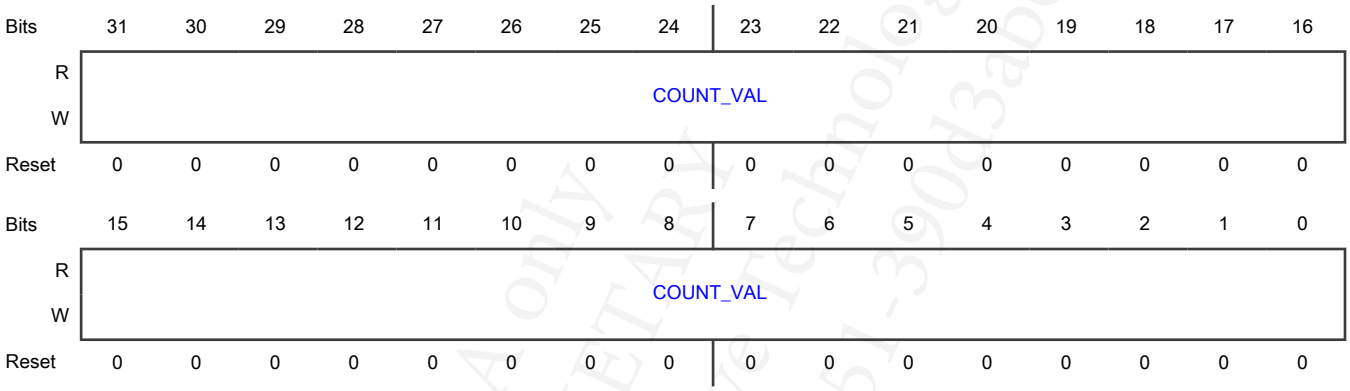
For a = 0 to 29:

Register	Offset
STATa	80h + (a × 4h)

Function

This register shows the PMUEVENT observer status.

Diagram



Fields

Field	Function
31-0 COUNT_VAL	Number of events counted by the counter corresponding to PMUEVENT[a]

5.3.8 Cortex-A53/GIC500 lockstep operation

The following AArch64 Generic Timer registers must be initialized before entering Lockstep mode. The GIC Timer registers must be symmetrically initialized to the same values across both Cortex-A53 processor clusters:

- CNTKCTL_EL1 - Counter-timer Kernel Control Register
- CNTFRQ_EL0 Counter-timer Frequency Register
- CNTPCT_EL0 - Counter-time Physical Count Register
- CNTVCT_EL0 - Counter-timer Virtual Count Register
- CNTP_TVAL_EL0 - Counter-timer Physical Timer TimerValue Register
- CNTP_CTL_EL0 - Counter-timer Physical Timer Control Register
- CNTP_CVAL_EL0 - Counter-timer Physical Timer Compare Value Register
- CNTP_TVAL_EL0 - Counter-timer Physical Timer TimerValue Register
- CNTV_CTL_EL0 - Counter-timer Virtual Timer Control Register

Reproduction under NDA only
COMPANY PROPRIETARY
Delivered by Automotive Technology Co., Ltd.
S-1625-4101-4888-b251-39013ab624c1

- CNTV_CVAL_EL0 - Counter-timer Virtual Timer Compare Value Register
- CNTVOFF_EL2 - Counter-timer Virtual Offset Register
- CNTHCTL_EL2 - Counter-timer Hypervisor Control Register
- CNTHP_TVAL_EL2 - Counter-timer Hypervisor Physical Timer TimerValue Register
- CNTHP_CTL_EL2 - Counter-timer Hypervisor Physical Timer Control Register
- CNTHP_CVAL_EL2 - Counter-timer Hypervisor Physical Time Compare Value Register
- CNTPS_TVAL_EL1 - Counter-timer Physical Secure Timer Timer Value Register
- CNTPS_CTL_EL1 - Counter-timer Physical Secure Timer Control Register
- CNTPS_CVAL_EL1 - Counter-timer Physical Secure Timer Compare Value Register

The Cortex-A53 clusters are configurable to run either independently or in a delayed lockstep configuration. In the lockstep mode, one cluster is considered the main cluster and the second cluster is considered the checker cluster. From a software perspective, the two clusters operate as a single cluster in the lockstep mode. Cortex-A53 Cluster0 is the main execution cluster, where Cortex-A53 Cluster1 follows execution of Cortex-A53 Cluster0. The replicated processing elements within each cluster contain:

- Cores
- L1 Cache Controller
- L2 Cache Controller
- Snoop Control Unit (SCU)

Equivalent operation of replicated resources is supervised by comparators (RCCU) on all functional signals leaving the cluster for the rest of the MCU. Any operational deviations between the supervised signals causes the FCCU to be notified of the discrepancy.

The Checker Cluster does not have a direct connection to the system. All of the outputs of Checker Cluster that target the system (as well as any other non-duplicated resource, like cache memories) terminates in an RCCU for analysis, and all the inputs to Checker Cluster from the system are siphoned from the main cluster system inputs. An abstract view of the implementation including delays and RCCUs is shown in the following figure, but does not show the local memory.

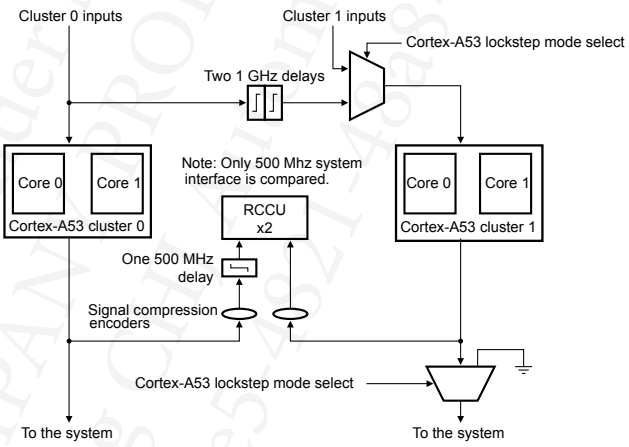


Figure 7. Cortex-A53 Integration

Similarly, the GIC500 interrupt controller for the Cortex-A53 can be optionally run in a delayed lockstep configuration. In the lockstep mode, one GIC500 instance is considered the main component and the second GIC500 instance is considered the checker component. From a software perspective, two components operate as a single interrupt controller in the lockstep mode. GIC500_0 is the main interrupt controller component, where GIC500_1 follows execution of GIC500_0.

Equivalent operation of replicated resources is supervised by comparators (RCCU) on all functional signals leaving the cluster for the rest of the MCU. Any operational deviations between the supervised signals causes the FCCU to be notified of the discrepancy.

The checker GIC500_1 does not have a direct connection to the system. All of the outputs of checker GIC500_1 that target terminates in an RCCU for analysis, and all the inputs to Checker GIC500_1 from the system are siphoned from the Main Cluster system inputs. An abstract view of the implementation including delays and RCCUs is shown in figure, but does not show the local memory.

NOTE

There are controls for selecting to run the Cortex-A53 clusters in either lockstep mode or performance mode. However, in either mode the GIC500 continues to run in lockstep mode with GIC500_0 as the main interrupt controller component.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c0

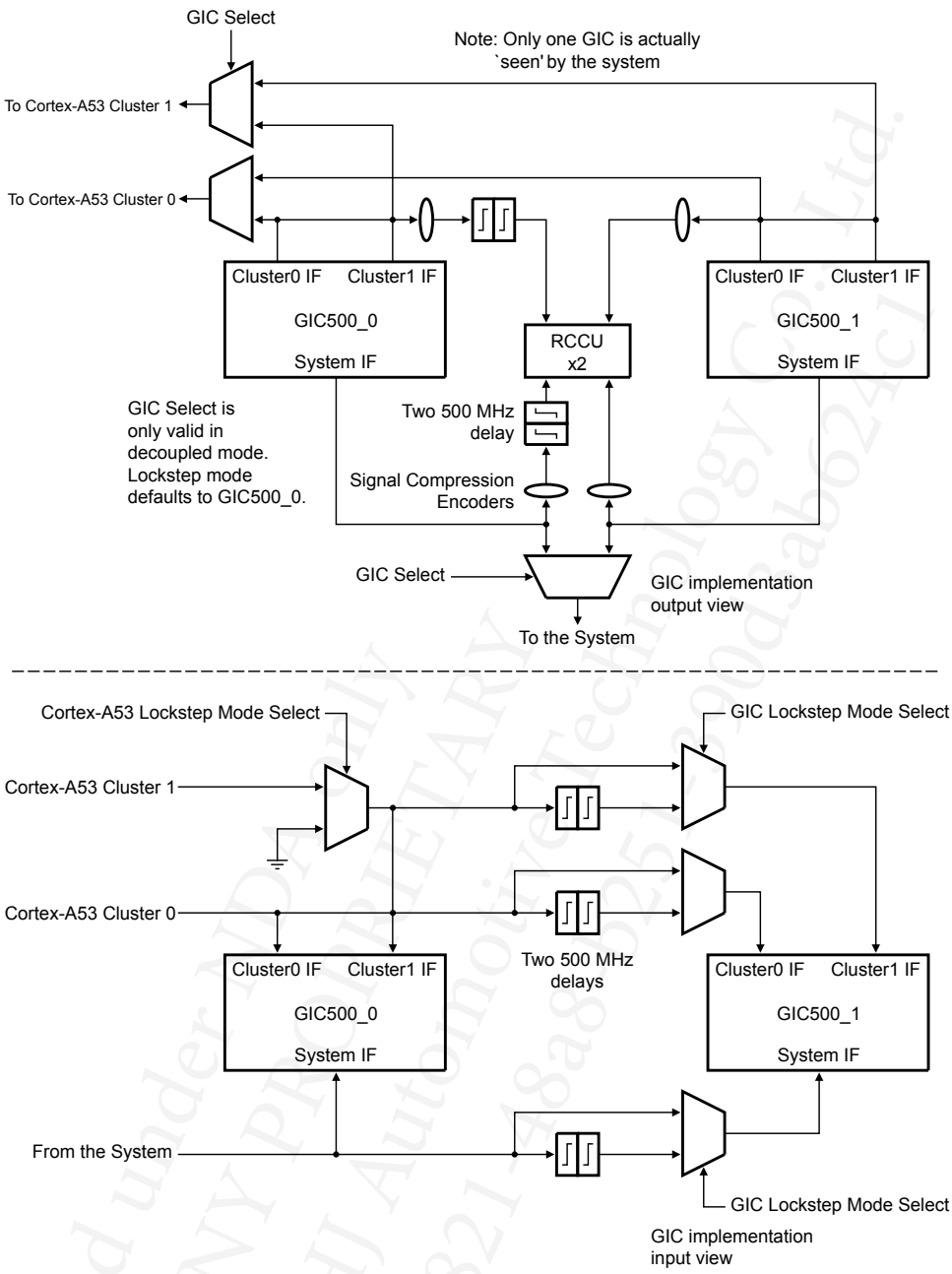


Figure 8. GIC500 Integration

5.3.9 Cortex-M7 lockstep operation

The checker Cortex-M7 does not have a direct connection to the system. All of the outputs of checker CM7 that target the system (as well as any other non-duplicated resource, like cache memories) will terminate in an RCCU for analysis, and all the inputs to Checker CM7 from the system will be siphoned from the Main Core system inputs. An abstract view of the implementation including delays and RCCUs is shown in the following figure.

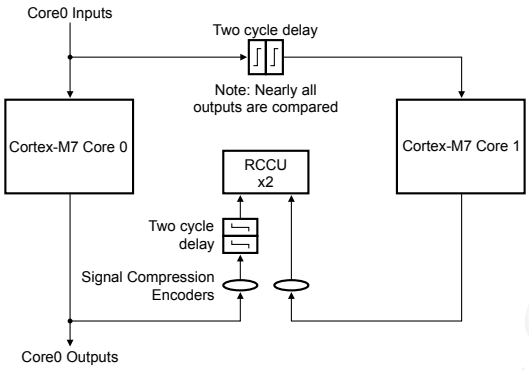


Figure 9. Cortex-M7 Integration

5.4 Cortex-M7 overview

5.4.1 Cortex-M7 core introduction

The Cortex-M7 processor is a highly efficient, high-performance embedded processor that features low interrupt latency, low-cost debug, and has backwards compatibility with existing Cortex-M profile processors. The processor has an in-order super-scalar pipeline and supports dual-issue of load/load and load/store instruction pairs to multiple memory interfaces.

Memory interfaces that the processor supports include:

- Tightly Coupled Memory (TCM) interface
- Harvard instruction and data caches and AXI master (AXIM) interface
- Dedicated low-latency AHB-Lite peripheral (AHBP) interface

The processor includes a Memory Protection Unit (MPU) that you can configure to protect regions of memory. The MPU contains support for up to 16 regions.

The Cortex-M7 processor includes floating-point arithmetic functionality, with support for single-precision arithmetic.

The Cortex-M7 core includes several memory interfaces:

- 64-bit Data Tightly Coupled Memory (DTCM), which is split into two interleaved 32-bit interfaces named D0TCM and D1TCM. RAM is expected to be connected to these interfaces and provides zero wait state data fetches.
- 64-bit AXI AMBA4 memory interface with a 32K instruction cache and a 32K data cache for efficient access to external resources
- 32-bit AHBP Peripheral port for interfacing with low latency on chip peripherals
- 32-bit AHBS Slave port for interfacing with slaves such as DMA

The Cortex-M7 core also includes:

- Nested Vectored Interrupt Controller (NVIC)
- Debug and Trace Unit (Coresight components)

The Cortex-M7 processor is intended for deeply embedded applications that require fast interrupt response features.

With D-Cache enabled, Cortex-M7 can randomly perform speculative load from any Normal type memory space through its AXIM bus.

5.4.1.1 Buses, interconnects, and interfaces

The Arm Cortex-M7 core includes four buses, as described in the following table.

Bus name	Description
AXI Master (AXIM) bus	This bus connects to NoC and provides the Cortex-M7 high-bandwith access to memory and peripherals.
AHB Slave (AHBS) bus	The AHBS interface provides system access to Cortex-M7 TCM.
AHB Peripheral (AHBP) bus	This bus connects to NoC and provides the Cortex-M7 low-latency D-side access to peripherals.
Private Peripheral bus (PPB)	The PPB bus provides access to these modules: • Arm modules such as the NVIC, ETCM, ITM, DWT, and ROM tables • Miscellaneous Control Module (MCM)

5.4.2 Debug facilities

This device has extensive debug capabilities including run control and tracing capabilities. The standard Arm debug port that supports JTAG.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab62461

Chapter 6
NoC Subsystem

6.1 Introduction

The NoC is an interconnect matrix which supports communication between various initiators and targets. A NoC instance on the Common Chassis supports communication to and from system resources including core clusters, cores, peripherals, and system RAM. A NoC instance in the Accelerator subsystem supports communication to and from modules in that subsystem.

6.2 Functional description

6.2.1 Observability

6.2.1.1 Concepts

The technology, which allows activity within NoC interconnects to be observed and reported, provides the following services:

- Error detection, logging, and reporting.
- Packet tracing.
- Traffic or link statistics computation.
- Transaction latency profiling.
- Events or statistics reporting to the system via industry standard third-party interface, MIPI® STPv2 over the ARM® ATB Coresight™ trace interface.

To provide these services, NoC technology uses task-specific probes placed on probe points located in the datapath topology. Traced packets and observed events are gathered by observers, which can issue interrupts and forward detailed data to supported third-party interfaces.

6.2.1.1.1 Packet and transaction probes

NoC packet probes provide the following features:

- Run-time programmable selection of packet probe points for probes connected to multiple points.
- Run-time programmable filtering for packet tracing.
- Recording of traffic and link statistics.
- Event counting of signals external to the NoC.
- Generation of trace or statistic packets, or both, for the observation network.
- Optional time stamping of packets.

NoC transaction probes provide the following features:

- Run-time programmable selection of transaction probe points.
- Transaction profiling in terms of latency or number of pending transactions.
- Event counting of signals external to the NoC.
- Generation of statistic packets for the observation network.
- Optional time stamping of packets.

NoC error probes detect packets in error and forward them to the associated observer via the observation network. The error packets are recorded by error loggers in the observer. An error logger can be programmed to assert an output signal when an error packet is received. The signal can be used as a system interrupt. Observers can contain one or more loggers and associated interrupt signals, each dedicated to specific error codes or error security levels.

By default, a new error can only be logged when the previous one is cleared by proper register access. If an error probe reports new errors and the associated logger is already full, then subsequent errors are dropped. Alternatively, the error logger can be programmed to queue up error packets and only accept them once the currently logged error has been cleared.

6.2.1.1.2 Observers

Data collected by certain NoC probes can be transferred to NoC observer units via the observation network.

Observers can be connected to NoCs via an ATB™ interface, with MIPI STPv2 encapsulation and time stamping. Collected data can be forwarded either to an off-chip interface, or on-chip storage buffer for analysis by an on-chip debugger.

6.2.1.1.2.1 Time stamping the observation subsystem

Data collected by certain NoC probes can be transferred to NoC observer units via the observation network.

Observers can be connected to NoCs via an ATB® interface, with MIPI STPv2 encapsulation and time stamping. Collected data can be forwarded either to an off-chip interface, or on-chip storage buffer for analysis by an on-chip debugger.

6.2.1.1.2.2 Time stamping probed events

The timestamp is added to the observation packet, regardless of whether the probe is configured for tracing or statistic collection.

The timestamp remains attached to the observation packet until the packet reaches the observer. This timestamp value is extracted and the difference between the current timestamp and the previous timestamp value, which was sent in the STPv2 stream, is calculated. This delta value is appended to the first STPv2 packet embedding the ATB packet.

6.2.1.2 Operation

6.2.1.2.1 Logging errors

6.2.1.2.1.1 Reconstructing transaction addresses

6.2.1.2.1.1.1 Calculating initiator – target transaction addresses

Once the initiator–target mapping associated with the logged packet has been identified, the complete transaction address, either from the perspective of the initiator or the target, can be calculated.

6.2.1.2.1.1.1.1 Initiator transaction address

From the perspective of the initiator, calculate InitAddr by:

```
InitAddr = Init localAddress | Addr
```

where:

- Addr – Logged packet transport address Addr given in register ErrLog3 and, additionally, ErrLog4, in the case where the address exceeds 32 bits.
- Init localAddress – Value from the identified initiator-target mapping in subsection "Aperture values” of the structure information file.

6.2.1.2.1.1.1.2 Target transaction address

The transaction address from the perspective of the target, calculate TargAddr by:

```
TargAddr = Targ localAddress | Addr
```

Where the definitions of the terms are the same as used for calculating the initiator transaction address (see Initiator transaction address).

6.2.1.2.2 Tracing packets

6.2.1.2.2.1 Basic packet tracing

To trace packets, the packet probe must be programmed as follows:

1. Set field TraceEn of register MainCtl to 1 to enable forwarding of traced packets to the connected observer. Optionally set field PayloadEn of register MainCtl to 1 if the packet payload should be included in the trace.
2. Set register TracePortSel to the value corresponding to the probe point of interest.
3. Set field GlobalEn of register CfgCtl to 1.
4. Set register FilterLut to -1 (all bits set).

NOTE

A lookup table acts as a simple gate when there are no filters.

After competing the programming sequence above, filters must be configured (see Filtering packets).

6.2.1.2.2.2 Filtering packets

Each packet probe supports up to four packet filters.

The following list comprises the complete set of filter registers, where the value for filter *n* ranges from 0 to 3.

- Filters_N_RouteIdBase
- Filters_N_Status
- Filters_N_RouteIdMask
- Filters_N_Length
- Filters_N_AddrBase_Low
- Filters_N_Urgency
- Filters_N_AddrBase_High
- Filters_N_UserBase
- Filters_N_WindowSize
- Filters_N_UserMask
- Filters_N_UserBaseHigh
- Filters_N_UserMaskHigh
- Filters_N_Opcode

6.2.1.2.2.2.1 Filtering on an address range

Packets which target a given address range of a specific mapping may be filtered by programming the set of filter registers.

Table 9. Packet filtering registers

Register	Value
Filters_N_RouteIdBase	RouteId subfields Initflow, TargFlow, and TargSubRange.
Filters_N_RouteIdMask	Set bits to 1 if RouteId fields are to be used by filter.
Filters_N_AddrBase_Low/ High	Address offset from initiator or target local address.
Filters_N_WindowSize	Log_2 of address range.

6.2.1.2.2.2.2 Selecting packets based on packet opcode

Packets can be selected based on the opcode field (Opc) in the packet header using register Filters_N_Opcode. In the following example, the filter is required to select all packets with Opc equal to RD and all packets belonging to a locked sequence.

Program the register fields of Filters_N_Opcode as shown in Table 10.

Table 10. Packet filtering registers

Field name	Setting
RdEn	1
LockEn	1

The other register fields are set to 0. Figure 10 shows a sequence of packets with the resulting filter hits.

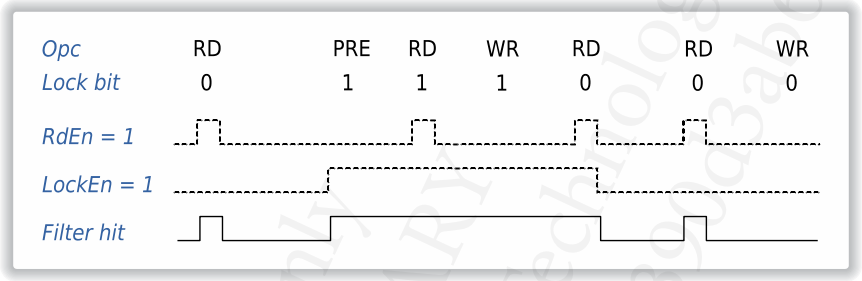


Figure 10. Sequence of packets

6.2.1.2.2.2.3 Excluding packet header fields from filter criteria

To ignore a specific packet header field in the filter, set its associated register with the value in shown Table 11.

Table 11. Excluding packet header field register settings

Register name	Setting
Filters_N_RouteIdMask	0
Filters_N_WindowSize	-1[1]
Filters_N_Opcode	Fh
Filters_N_Status	3h
Filters_N_Length	Fh
Filters_N_Urgency	0
Filters_N_UserMask (if present)	0
Filters_N_UserMaskHigh (if present)	0

1. Setting register Filters_N_WindowSize to –1 programs the filter to accept packets which target anywhere in the full address range supported by the transport network. Hence the address is effectively excluded from the filter criteria.

6.2.1.2.2.2.4 Tracing all packets

If it is required to trace all packets, then the probe's filtering function must be programmed as follows:

1. Program one of the filters to accept all packet header field values. Set the filter registers to the values given in the table in the preceding section "Excluding packet header fields from filter criteria".

2. Program register FilterLut to select the output of filter programmed in the previous step.

6.2.1.2.2.3 Tracing error packets

Error packets, that is packets with header field Status set to ERR, are not traced by default. To include error packets in the trace output, write MainCtl[ErrEn] = 1 and MainCtl[ErrEn] = 1.

6.2.1.2.2.4 Combining filters

A look-up table (LUT) is instantiated in the probe, which allows a hit to be generated on any logical combination of packet filters. The logic function is programmed via register FilterLut. The value programmed in the register corresponds to the truth table applied on the filter outputs.

For example, with two filters F0 and F1, sixteen possible functions can be defined, a subset of which is shown in the following table.

Table 12. Example subset of sixteen possible functions

Filter hit			Required filter combination					
F1	F0		F0	F1	F1 & F0	F1 | F0	F1 ^ F0	~F0
0	0		0	0	0	0	0	1
0	1		1	0	0	1	1	0
1	0		0	1	0	1	1	1
1	1		1	1	1	1	0	0
			Value to program in FilterLut					
			Ah	Ch	8h	Eh	6h	5h

NOTE

In this example, setting register FilterLut to Fh will generate a LUT hit for any combination of filter hits, effectively bypassing the filters.

6.2.1.2.2.5 Configuring and managing trace alarm events

Trace alarm events can be separately enabled, recorded, and cleared using registers TraceAlarmEn, TraceAlarmStatus, and TraceAlarmClr respectively. In each register, a dedicated bit is associated with each filter hit; the MSB is associated with the LUT hit.

The trace alarm logic can be configured to drive output signal TraceAlarm if one or more trace alarm events are recorded in register TraceAlarmStatus. Signal TraceAlarm can be used directly as an interrupt source, or can be routed through a sideband manager.

To use the trace alarm feature

1. Enable hits from selected filters 0 to (nFilter – 1) to generate a trace alarm by setting register TraceAlarmEn bits 0 to (nFilter – 1) to 1.
   - Optionally, enable LUT hits to generate a trace alarm by setting the MSB of TraceAlarmEn.
2. Set field AlarmEn to 1 in register MainCtl in order to activate signal TraceAlarm when a trace alarm occurs.

If TraceAlarm is asserted, the interrupt service routine should:

1. Identify the source or sources of the trace alarm by reading register TraceAlarmStatus.
2. Clear the source or sources of the trace alarm by setting the corresponding bits to 1 in register TraceAlarmClr. Signal TraceAlarm deasserts when all bits in TraceAlarmStatus have been cleared.

6.2.1.2.3 Collecting statistics

6.2.1.2.3.1 Counting packets over fixed period

The following programming sequence counts packets at a given probe point using statistic counter 0.

1. Set field StatEn to 1 in register MainCtl.
2. Set register Counters_0_PortSel to the value corresponding to the probe point of interest.
3. Set register Counters_0_Src to 6h (PKT) to count packets.
4. Specify the period during which the packets should be counted by setting register StatPeriod to: log2 (interval expressed in number of probe clock cycles)
5. Set field GlobalEn of register CfgCtl to 1 to enable packet counting.

Once time $2^{StatPeriod}$ has elapsed, the number of packets counted is dumped to the observer.

6.2.1.2.3.2 Counting filtered packets over fixed period

The following programming sequence counts hits from packet filter 2 using statistic counter 0.

NOTE

The important distinction between this sequence and the one in a related programming example (see Counting packets over fixed period) is that the probe port is selected using register TracePortSel rather than Counters_0_PortSel

1. Set field StatEn to 1 in register MainCtl.
2. Set register TracePortSel to the value corresponding to the probe port of interest.
3. Set register Counters_0_Src to 0xE (FILT2) to count packets from filter 2.
4. Specify the period during which the packets should be counted by setting register StatPeriod to: log2(interval expressed in number of probe clock cycles)
5. Set field GlobalEn of register CfgCtl to 1 to enable packet counting.

Once time $2^{StatPeriod}$ has elapsed, the number of filtered packets counted is dumped to the observer.

6.2.1.2.3.3 Measuring bandwidth

The following programming sequence example shows how a packet probe can be used to measure the average bandwidth, over a given interval, at a probe point.

Some important points to note about this example are:

- Statistics counters are chained together to support the maximum theoretical bandwidth. Counter 0 is configured to count bytes; counter 1 increments when counter 0 rolls over.
- The counter values are dumped to an observer after time $2^{StatPeriod}$.
- A maximum bandwidth threshold is defined. A statistics alarm signal is asserted if this threshold is exceeded when statistics data is dumped. The comparison is made against the concatenated value of the two counters.

The programming sequence is as follows:

1. Set field StatEn to 1 in register MainCtl.
2. Set register Counters_0_PortSel to the value corresponding to the probe point of interest.
3. To count bytes, set register Counters_0_Src to 8h (BYTES).
4. To increment when counter 0 wraps, set register Counters_1_Src to 10h (CHAIN).

5. Specify the period during which the bytes should be counted by setting register StatPeriod to: log2 (interval expressed in number of probe clock cycles)
6. Set register Counters_0_AlarmMode to 2h (MAX) and Counters_1_AlarmMode to 0h (OFF). A statistics alarm is raised if the combined count of counter 0 and counter 1 is greater than the value specified in register StatAlarmMax.
7. Set register StatAlarmMax to a value corresponding to the upper bandwidth limit required to trigger the statistics alarm.
8. Set field AlarmEn to 1 in register MainCtl. Ensure bit 0 of register StatAlarmEn is set to 1 (default value) if the alarm output signal StatAlarm is required.
9. Set field GlobalEn of register CfgCtl to 1 to enable the counting of bytes.

In the event StatAlarm is asserted, the interrupt service routine should:

1. Optionally read the number of packets contained in registers Counters_0_Val and Counters_1_Val.
2. Clear the statistics alarm by setting bit 0 to 1 in StatAlarmClr. The signal StatAlarm deasserts.

NOTE

Statistics counters are frozen and the dump timer is suspended until bit 0 is set to 1 in register StatAlarmClr.

6.2.1.2.4 Transaction profiling operation

The programming sequence examples in this section constitute, for a given category of use cases, the obligatory base from which to develop sequences to handle specific needs within the category.

6.2.1.2.4.1 Measuring latency

The following example of a run-time programming sequence shows how a transaction probe can be used to measure transaction latency.

The measurement specification is:

- For init0, monitor delays for read transactions only.
- For init1, monitor delays for read and write transactions.

6.2.1.2.4.2 Programming the packet probe statistics subsystem

The statistics counters in the packet probe unit of the transaction probe must be programmed to count the delay events recorded by the transaction profiler unit.

1. Map the five delay event bins, associated with transactions issued by init0, from the transaction profiler unit to statistics counters in the packet probe unit by programming the following registers:
   - Counters_0_Src = 20h => Counts the number of transactions from init0 with a delay between 0 and 1.
   - Counters_1_Src = 21h => Counts the number of transactions from init0 with a delay between 2 and 4.
   - Counters_2_Src = 22h => Counts the number of transactions from init0 with a delay between 5 and 9.
   - Counters_3_Src = 23h => Counts the number of transactions from init0 with a delay between 10 and 99.
   - Counters_4_Src = 24h => Counts the number of transactions from init0 with a delay greater than 100.
2. Map the five delay event bins, associated with transactions issued by init1, from the transaction profiler unit to statistics counters in the packet probe unit by programming the following registers:
   - Counters_5_Src = 25h => Counts the number of transactions from init1 with a delay between 0 and 1.
   - Counters_6_Src = 26h => Counts the number of transactions from init1 with a delay between 2 and 4.
   - Counters_7_Src = 27h => Counts the number of transactions from init1 with a delay between 5 and 9.
   - Counters_8_Src = 28h => Counts the number of transactions from init1 with a delay between 10 and 99.

- Counters_9_Src = 29h => Counts the number of transactions from init1 with a delay greater than 100.

3. Set register StatPeriod to $\log_2$(period), that is, the period during which delay events should be counted before being sent to the observer.
4. Set field StatEn of register MainCtl to 1 to enable the statistics counters to count delay events.

6.2.1.2.4.3 Enabling the transaction probe

1. Set field GlobalEn of register CfgCtl to 1 to enable the statistics counters.
2. Set register En to 1 to enable the transaction profiling counters.

6.2.1.2.4.4 Programming the init0 NIU transaction filter

1. Set register Mode to 1 to set the delay mode to LATENCY.
2. Set field RdEn of register Opcode to 1 to monitor read transactions only.

6.2.1.2.4.5 Programming the init1 NIU transaction filter

1. Set register Mode to 1 to set the delay mode to LATENCY.
2. Set fields RdEn and WrEn of register Opcode both to 1 to monitor read and write transactions.

6.2.1.2.5 Activating observer debug output

6.2.1.2.5.1 STPv2 interface

The observers generate STPv2 formatted trace packets. To enable trace output, the following registers must be programmed.

1. Specify the required period between STP synchronization sequences in register AsyncPeriod. Ensure that the value of AsyncPeriod is large enough to allow STPv2 cells, encapsulating Trace / Statistic Packets, to be inserted in between STPv2 synchronization sequences.
2. Ensure that register STPv2En is set to 1 to enable the STPv2 interface.
3. Set register AtbId to a value that is unique not only within the NoC but also within the debug system.
4. Set register AtbEn to 1 to enable the ATB function.

6.2.1.3 Technical reference

NoC observability technology comprises two principle components: observation probes and observer units.

The section provides detailed technical reference information for these components, along with the NoC packet format description needed to analyze error logs and packet trace information.

6.2.1.3.1 Observation probes

This section provides detailed technical reference information for the NoC Observability technology collectively called observation probe units.

6.2.1.3.1.1 Packet probe

The NoC packet probe unit collects trace and statistics data. The unit can be used in conjunction with the NoC transaction profiler unit to create a transaction probe.

6.2.1.3.1.1.1 Packet probe features

The unit has the following main features:

- Support for any transport packet format and serialization.
- Trace mode with programmable packet filters.
- Optional packet payload tracing.
- Programmable flow-control modes: Overflow Management and Intrusive
- Statistics can be collected and sent to an observer either periodically or after manual triggering.
- Alarms on both trace filters and statistics counters.
- Optional time stamping of probed events.

6.2.1.3.1.2 Overflow flow-control for tracing

When not using Intrusive Mode (see register field IntrusiveMode of register MainCtl), an overflow packet is added after the stalled packet(s), which reports how many observation packets have been lost.

This Overflow packet has the following format:

- Opc = WR, Status = ERR.
- Debug = (probeId, Urgency= 0, Mod=1).
- RouteId = RouteID of the packet that triggered the overflow.
- Addr[8:0] = number of lost or uncompleted observation packets, minus one.
- Len: 0 if no payload lost; 1 otherwise.
- Lock = Addr[8:0] = Security = User = 0.

In case of payload overflow, the current packet is completed according to its type; then an overflow packet is also sent:

- Write request packets are fully completed with the correct number of words, all having null Data and BE.
- A single data word with null data that terminates the tracing of the read payload for that transaction completes Read response and Read continuous packets.

6.2.1.3.1.3 Transaction filter

The NoC transaction filter unit uses programmable criteria to select request and response events.

The unit is integrated into the NIU and is connected to the NIU generic-to-transport unit. More than one transaction filter can be connected to a NoC transaction profiler unit.

6.2.1.3.1.4 Transaction profiler

The NoC transaction profiler unit receives events from one or more transaction filters and organizes them into histogram bins, defined by programmable thresholds.

The unit is connected to a standard Packet Probe via the unit external event input ports.

6.2.1.3.2 Observers

This section provides detailed technical reference information for the NoC Observability technology Observer unit.

6.2.1.3.2.1 Error logging

Errors that occur in packets observed by probes can be logged.

6.2.1.3.2.1.1 Features

- Error filtering can be based on selected error codes, user and security flags and sideband signals.
- An optional fault interrupt can be asserted on an error.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c46235-4821-4a38-b251-390d3ab624c1

- Can be used in conjunction with trace probe filtering to monitor a specific port.

6.2.1.3.2.1.2 NoC error codes

The following table lists all error codes that are relevant to NoC interconnect operation.

Table 13. NoC error codes

Code	Value	Source	Type
SLV	0	Target	Target error detected by slave.
DEC	1	Initiator NIU	Address decode error.
UNS	2	Target NIU	Unsupported request.
DISC	3	Power disconnect	Disconnected target or domain.
SEC	4	Initiator NIU or firewall	Security violation.
HIDE	5	Firewall	Hidden security violation, reported as ok to initiator.
TMO	6	Target NIU	Timeout.
RSV	7	None	Reserved.

6.2.1.3.2.2 STPv2 stream generation

The NoC STPv2 adapter in the observer encapsulates ATB packets and generates synchronization sequences to provide a MIPI-STPv2 compliant STP stream. Time stamps can optionally be included in the stream.

6.2.1.3.2.2.1 Features

- Compliant with AMBA®-3 ATB and MIPI® STPv2 specifications.
- Supports ATB data widths from 8 to 64 bits.
- Supports STPv2 opcodes ASYNC, VERSION, NULL, NULL_TS, D64, D64MTS, and MERR.
- Synchronization sequence generation with optional time stamp.
- Encapsulation of probe time stamps.
- Supports all time stamp length encodings.

NOTE

For details, see MIPI® Alliance Specification for System Trace Protocol (STP), version 2.00.01, 24 February 2010.-STPv2 specification.

6.2.1.3.2.2.2 Synchronization sequence generation

The synchronization sequence permits the SoC debug system to identify the STPv2 packet boundary. A time stamp can be optionally inserted in the sequence to which the SoC debug system may resynchronize itself.

The synchronization sequence is sent when one of the following events occurs:

- After a reset.
- On the rising edge of input signal SyncReq.
- When the number of clock cycles since the last synchronization sequence is between 2AsyncPeriod and 223 inclusive, and register AsyncPeriod is greater than 0.

NOTE

The synchronization sequences are inserted instead of ATB synchronization cells.

The STPv2 adapter can issue synchronization sequences either with or without time stamps. The examples in figures that follow are represented according to the following key.

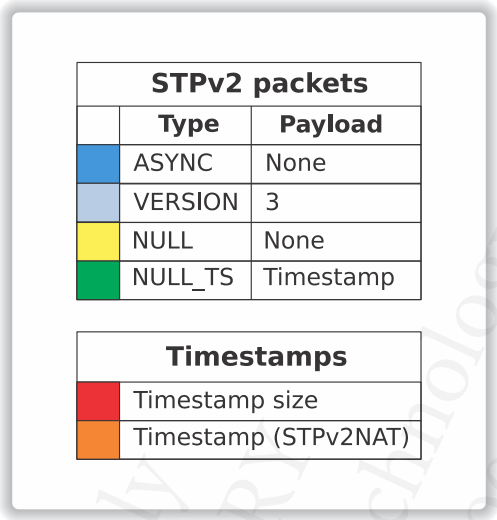


Figure 11. STPv2 synchronization sequence key

6.2.1.3.2.2.2.1 STPv2 synchronization sequence without timestamp

Synchronization sequences are sent without a timestamp when the observer STPv2 parameter wTimeStamp is set to 0. In this case, the sequence comprises the STPv2 packets ASYNC and VERSION, as well as NULL packets for padding. Such a sequence is shown in the following figure.

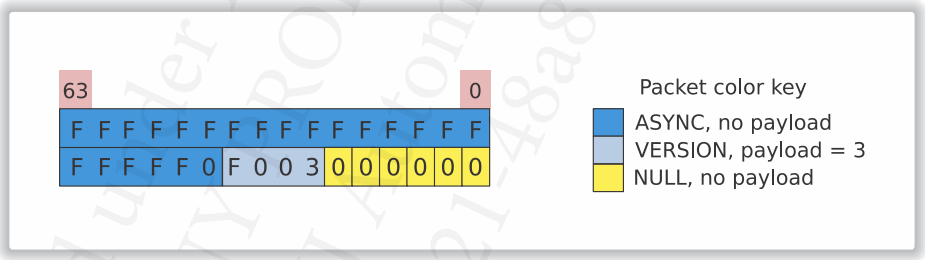


Figure 12. STPv2 synchronization sequence without timestamp

NOTE

The sequence in the figure assumes a data width of 64 for the ATB debug port, as set by parameter wData (Specification: Interface: Observation: debugOutput).

6.2.1.3.2.2.2.2 STPv2 synchronization sequence with timestamp

If wTimeStamp is greater than zero, synchronization sequences are sent with time stamps. The sequence comprises the STPv2 packets ASYNC, VERSION, NULL, and NULL_TS. The time stamp value, received on observer input signal TimeStamp and extended to 64 bits, is sent in the payload of NULL_TS.

The synchronization sequence with time stamp is shown in the following figure.

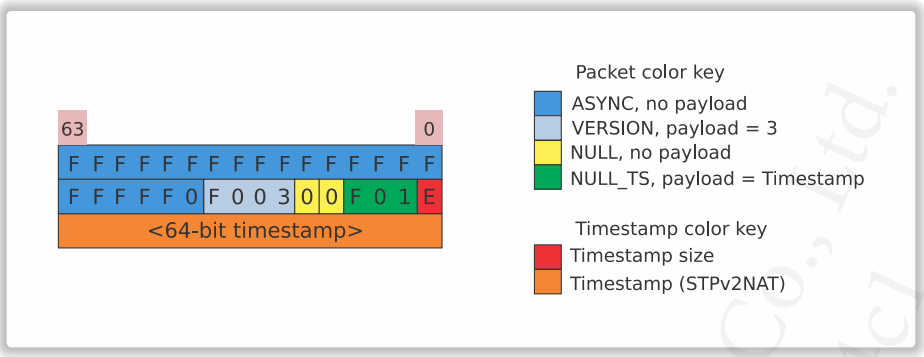


Figure 13. STPv2 synchronization sequence with timestamp

In the figure:

- Regardless of the width of the time stamp, specified by wTimeStamp, the time stamp is extended to 64 bits.
- All 16 nibbles of the time stamp are significant, indicated by the time stamp size nibble being set to Eh.
- The protocol version is set to 3 in the payload of packet VERSION. This indicates that the time stamp is encoded using the uncompressed STPv2NAT format.

6.2.1.3.2.2.3 Encapsulation of ATB packets

The STPv2 adapter encapsulates 64–bit ATB cells into the payload of 64–bit STPv2 data packets. The first ATB cell of an observation packet is encapsulated into a D64MTS STPv2 packet; D64 STPv2 packets are used for subsequent cells of the same observation packet. If observer parameter wTimeStamp is set to a value greater than 0, a Timestamp Size nibble and the time stamp are appended to the D64MTS packet. If wTimeStamp is 0, only the Timestamp Size nibble, set to a value of 0, is appended.

If an overflow packet is received by the observer, an MERR STPv2 packet is sent, indicating an STPv2 master error, prior to the encapsulation of the probed data into D64MTS and D64 packets.

6.2.1.3.2.2.3.1 Timestamp compression

When wTimeStamp is greater than 0, the number of bits sent in the Timestamp field of D64MTS packets depends on the difference between the time stamp from the current observation packet and the previous time stamp value sent in the STPv2 stream. The previous time stamp may be either itself the time stamp of the previous observation packet or the absolute time stamp sent in an STPv2 synchronization sequence.

Only nibbles which are different between the current and previous time stamps are sent, left-aligned, in the Timestamp field of the D64MTS packet. The field is preceded by a Timestamp Size nibble, which indicates the field size, in nibbles. The encoding of TimeStamp Size is defined in Section 7.2 of MIPI STP specification version 2.00.01.

The following table shows several examples of time stamp compression values.

Table 14. Timestamp compression values

Timestamp value		STP timestampe	
Previous	Current	Timpstamp Size	Compressed timestamp
0000_0000_1111_2222h	0000_0000_1111_2222h	0	—
0000_0000_1111_2222h	0000_0000_1111_2234h	2	34h
0000_0000_1234_5678h	0000_0000_1254_5601h	6	545601h
0123_0000_0000_0000h	0226_0000_0002_0030h	16[1]	0226_0000_0002_0030h
0000_0000_0012_3459h	0000_0000_0012_3457h	1	7h[2]

1. The Timestamp Size nibble is encoded for 16 nibbles even though only 15 nibbles are different because there is no encoding for 15 (see MIPI STP specification).
2. It is possible for the time stamp value of the packet being processed to be less than the time stamp of the previous packet. This can occur if an observation packet, which has an earlier time stamp than a packet from another probe, is delayed due to buffering in the observation network.

6.2.1.3.2.3 Observer registers

Observer functions, such as error logging, ATB formatting, and STPv2 encapsulation, are run-time programmable via registers. For a given function, in addition to the specific registers for that function, the register set includes standard identification registers .

See "APB memory map" in the Debug Subsystem chapter in the compute complex RM.

See "Memory maps/register definitions" in this document for reset values of registers.

6.2.1.3.2.3.1 Error logging registers

Registers related to error logging are summarized in the following table.

Address offset	Name	Type	Retention
0008h	FaultEn	Control	Yes
000Ch	ErrVld	Status	No
0010h	ErrClr	Pulse	No
0014h	ErrLog0	Status	No
0018h	ErrLog1	Status	No
001Ch	ErrLog2	Status	No
0020h	ErrLog3	Status	No
0024h	ErrLog4	Status	No
0028h	ErrLog5	Status	No
002Ch	ErrLog6	Status	No
0030h	ErrLog7	Status	No
0034h	ErrLog8	Status	No
0038h	StallEn	Control	Yes

6.2.1.3.2.3.1.1 FaultEn

When 1-bit register FaultEn is set to 1, error reporting signal Fault is enabled and asserted when register ErrVld is 1. When FaultEn is cleared to 0, signal Fault is driven to 0.

- Bit – (0)

Related signals:

- Fault

6.2.1.3.2.3.1.2 ErrVld

Register ErrVld indicates that the ErrLog registers contain valid error information.

- Bit – (0)

6.2.1.3.2.3.1.3 ErrLog0

Register ErrLog0 logs fields from the NoC packet transport protocol header of the logged packet.

The MSB field indicates the transport protocol version number.

The following table shows the field assignments. All other bits are reserved and read as 0.

Table 15. ErrLog0 field assignments

Bit range	Field name	Type	Access
0	Lock	Status	Read only
4:1	Opc	Status	Read only
10:8	ErrCode	Status	Read only
27:16	Len	Status	Read only
31	Format	Status	Read only

6.2.1.3.2.3.1.3.1 Lock

Register field Lock contains the packet header bit Lock.

6.2.1.3.2.3.1.3.2 Opc

Register field Opc contains the packet header field Opc.

6.2.1.3.2.3.1.3.3 Len1

Register field Len1 contains the packet header field Len1.

6.2.1.3.2.3.1.3.4 ErrCode

Register field ErrCode contains the packet header field ErrCode.

If the packet field does not exist, the register field contains 0.

6.2.1.3.2.3.1.3.5 Format

Register field Format specifies the version of the NoC packet transport protocol .

6.2.1.3.2.3.1.4 ErrLog1

Register Errlog1 contains the least significant part of the NoC packet transport protocol packet header field RouteId of the logged error. Up to 32 bits will be stored, depending on the width of RouteId. The field is right-aligned in the register, and unused bits are read as 0.

The RouteId value can be used to identify the initiator–target mapping associated with the logged error, in order to calculate the complete address.

When decoding RouteId sub-field SeqId, bit index 0 corresponds to the value for packet header field Opc when:

- The transaction comes from an interface with a protocol, such as AHB, AXI, or AMBA, that allows response disorder based on transaction opcodes.
- The NoC supports other interface protocols that do not.

A description of the sub-fields comprising RouteId is provided in related reference documentation.

- Type – Status

6.2.1.3.2.3.1.5 ErrLog2

Register Errlog2 contains the most significant part of the NoC packet transport protocol packet header field RouteId of the logged error. Up to 16 MSBs of RouteId are stored, depending on the width of RouteId. The field is right-aligned in the register, and unused bits are read as 0.

The RouteId value can be used to identify the initiator–target mapping associated with the logged error, in order to calculate the complete address.

A description of the sub-fields comprising RouteId is provided in related reference documentation.

This register is only present if the width of RouteId is greater than 32 bits.

- Type – Status

6.2.1.3.2.3.1.6 ErrLog3

Register Errlog3 contains the least significant part of the NoC packet transport protocol packet header field Addr of the logged error. Up to 32 bits will be stored, depending on the width of Addr. The field is right-aligned in the register and unused bits are read as 0.

Because ErrLog3 only contains the transport address offset within the particular mapped region, its value must be used in conjunction with the local addresses of the initiator–target mapping, in order to reconstitute the complete transaction address. The mapping is identified using RouteId, logged in ErrLog1 and optionally ErrLog2.

6.2.1.3.2.3.1.7 ErrLog4

Register Errlog4 contains the most significant part of the NoC packet transport protocol packet header field Addr of the logged error. Up to 16 MSBs of Addr are stored, depending on the width of Addr. The field is right-aligned in the register and unused bits are read as 0.

Because ErrLog4 only contains the transport address offset within the particular mapped region, its value, concatenated with ErrLog3, must be used in conjunction with the local addresses of the initiator–target mapping in order to reconstitute the complete transaction address. The mapping is identified using RouteId, logged in ErrLog1, and optionally, ErrLog2.

This register is only present if the width of Addr is greater than 32 bits.

6.2.1.3.2.3.1.8 ErrLog5

Register Errlog5 contains the least significant part of the NoC packet transport protocol packet header field User of the logged error.

Up to 32 bits will be stored, depending on the width of User. The field is right-aligned in the register, and unused bits are read as 0.

6.2.1.3.2.3.1.9 ErrLog6

Register Errlog6 contains the midrange part of field User contained in the packet header of the packet transport protocol. Bit User (32) up to bit User (63) will be stored, depending on the width of User. The field is right-aligned in the register, and unused bits are read as 0.

The register is present only if the width of User is greater than 32 bits.

6.2.1.3.2.3.1.10 ErrLog7

Register Errlog7 contains the NoC packet transport protocol packet header field Security of the logged error. The field is right-aligned in the register and unused bits are read as 0.

6.2.1.3.2.3.1.11 ErrLog8

Register Errlog8 contains the most significant part of the NoC packet transport protocol packet header field User of the logged error. Bits User(64) up to User(95) are stored, depending on the width of User. The field is right-aligned in the register, and unused bits are read as 0.

The register is present only if the width of User is greater than 64 bits.

6.2.1.3.2.3.1.12 StallEn

Register StallEn controls the stall mode of the NoC error logger.

When set to 1, error packets (status = ERR) are stalled, that is, queued at the input to the error logger. The packets are only passed into the error logger once the currently logged error packet has been cleared by setting register ErrClr to 1.

When set to 0, only the first error packet is stored in the error logger. Subsequent error packets are lost.

NOTE

The number of error packets which may be queued is 4.

6.2.1.3.2.3.2 ATB endpoints registers

Registers which are used to control the ATB endpoint are summarized in the following table.

Address offset	Name	Type	Retention	Width
0008h	AtbId	Control	Yes	7 bits
000Ch	AtbEn	Control	Yes	1 bit
0010h	SyncPeriod	Control	Yes	3 bits (4:0)

6.2.1.3.2.3.2.1 AtbId

Register AtbId sets the ATB identifier of the observer. The identifier is carried on signal ATId.

Each observer within the NoC must have a unique ID value. The value is used along with probe IDs to identify probes.

NOTE

Values 00h and 70h to 7Fh are reserved for ARM® CoreSight™ debug systems.

6.2.1.3.2.3.2.2 AtbEn

When AtbEn = 1, register AtbEn enables ATB output.

NOTE

Writing AtbEn = 0 is equivalent to a power disconnect on the affected socket: the current transmission of an ATB packet is completed before the ATB output is disabled.

6.2.1.3.2.3.2.3 SyncPeriod

Register SyncPeriod sets the period, in terms of the number of ATB cells sent, between insertion of ATB synchronization cells.

The following table summarizes the relationship between SyncPeriod and the delay between synchronization cells.

Table 16. SyncPeriod and delay between synchronization cells

SyncPeriod value	Period of sync cell insertion
0	No sync cell insertion
1...2	8
3...22	$2^{(SyncPeriod + 1)}$
23...31	Not supported

SyncPeriod is not present when setting parameter debugOutput of the observer to STPv2, since ATB synchronization cells are not generated when using the STPv2 adapter.

6.2.1.3.2.3.3 STPv2 encapsulation registers

Registers relative to STPv2 encapsulation are summarized in the following table.

Address offset	Name	Type	Retention	Width
0008h	AsyncPeriod	Control	Yes	5 bits (4:0)
000Ch	STPv2En	Control	Yes	1 bit (bit 0)

SyncPeriod is not present when setting parameter debugOutput of the observer to STPv2, since ATB synchronization cells are not generated when using the STPv2 adapter.

6.2.1.3.2.3.3.1 AsyncPeriod

Register AsyncPeriod defines the period, in observer clock cycles, between transmission of STPv2 synchronization sequences.

The following table summarizes the relationship between AsyncPeriod and the delay between synchronization sequences.

Table 17. AsyncPeriod and delay between synchronization cells

AsyncPeriod value	Period between synchronization sequences
0	No sync sequence inserted due to AsyncPeriod
1...22	$2^{(SyncPeriod + 1)}$
23...31	2^{23}

SyncPeriod is not present when setting parameter debugOutput of the observer to STPv2, since ATB synchronization cells are not generated when using the STPv2 adapter.

6.2.1.3.2.3.3.2 STPV2En

When STPV2En = 1, register STPV2En enables the STPv2 output.

When STPV2En = 0, the observer debug interface is disabled: neither STPv2 packets nor synchronization sequences are sent.

By default, the reset value is 1 so that STPv2 stream generation is enabled. If it should be enabled explicitly by software, the reset value can be changed to 0 by setting its parameter rstVal (Structure: Parameters: Control Setting).

NOTE

Writing STPv2En =0 has an equivalent effect to a power disconnect on this interface: the STPv2 encapsulation and transmission of an ATB packet will be completed before the observer debug output is disabled.

6.2.2 QoS

6.2.2.1 Concepts

In NoC interconnects, quality of service (QoS) is the statistical allocation of throughput and delay, in terms of bandwidth and latency, to the transactions transported between initiators and targets.

This section describes how QoS mechanisms balance the access of the traffic of different initiators to suit the finite available resources of the interconnect.

6.2.2.1.1 QoS generators

NoC QoS generators are used to prevent initiator sockets from degrading the QoS of other sockets. The generators, in order of increasing functionality:

- Fixed – Assigns a fixed urgency levels to packets.
- Limiter – Generates flow control when a pre-defined bandwidth threshold is reached.

- Regulator – Ensures that initiators obtain the required bandwidth from targets.

QoS generators are programmable through registers.

Software can only switch a generator to an operating mode with a lower level of functionality.

For example, regulators can be programmed to operate in either Limiter or Fixed mode.

Bandwidth counters are reset when modes are switched.

6.2.2.1.1.1 Fixed mode

Fixed generators assign a pre-defined urgency level, which can be different for read and write transactions, to each packet transiting the NIU.

The assigned urgency levels can be programmed by software.

6.2.2.1.1.2 Limiter mode

NoC QoS generators in limiter mode ensure that initiators do not send too many requests to targets.

To do this, the generator monitors requests instead of responses because monitoring responses would delay feedback, thus allowing initiators to send many requests until bandwidth is restricted.

6.2.2.1.1.3 Regulator mode

NoC QoS generators in regulator mode ensure that initiators obtain the required bandwidth from targets. When bandwidth is insufficient, requests urgency values must be adjusted before the request-path buffering capacity is saturated. Consequently, regulators monitor response rates, because this is the fastest way of identifying slow processing by targets.

Because regulators measure the NIU response bandwidth, they can restrict initiator bandwidth more evenly than limiters.

When a predefined bandwidth threshold is reached, the urgency, pressure, and hurry levels at the initiator interface decrease. The threshold is determined by field Saturation, according to the following equation:

```
Threshold = (Saturation × 16) × 256 / 2
```

6.2.2.2 Programming

This section provides guidelines for programming NoC QoS hardware features to meet performance requirements in different use cases.

6.2.2.2.1 Programming QoS generators

NoC QoS generators, programmed correctly, ensure that throughput is allocated to initiators according to the needs of the system. These needs can include latency requirements.

6.2.2.2.1.1 Programming fixed generators

An initiator with a fixed type of QoS generator can be programmed to give its data transfers a desired priority. The priority remains constant until reprogrammed.

Read and write data transfers can be given different priorities. For example, system performance can be affected more by reads than by writes, and it is therefore desirable to give higher priority to reads.

The following table lists the registers that control data transfer priority.

Table 18. Registers controlling data transfer priority

Register name	Byte offset	Mode	Field	Bits	Bit offset	Description
Priority	08h	bwdComp = True	P0 / Low	2	0	Priority of writes.

Table continues on the next page...

Table 18. Registers controlling data transfer priority (continued)

Register name	Byte offset	Mode	Field	Bits	Bit offset	Description
			P1 / High	2	8	Priority of reads.
			Mark	1	31	Value of 0 for backwward compatibility.
		bwdComp = False	P0 / Low	log_2(nUrgencyLevel + 1)	0	Priority of writes.
			P1 / High	log_2(nUrgencyLevel + 1)	8	Priority of reads.
			Mark	1	31	Value of 10 for backwward compatibility.
Mode	0Ch		Mode	2	0	0 = Fixed 2 = Bypass
ExtControl	18h		SocketQos En	1	0	Initiator drives QoS
			ExtThrEn	1	1	No function.
			IntClkEn	1	2	No function.

The parameter bwdComp indicate if the old (True) or new (false) layout of the Priority register is used.

A chip operating in different use cases may reprogram all of its initiators accordingly. Because priorities can be changed at any time with no risk to data integrity, reprogramming can be in real-time based on system feedback.

Depending on chip design, priorities might be reset when a part of a chip is powered down. The reset value of all priorities is zero.

6.2.2.2.1.2 Programming limiter generators

Initiators with a limiter type of QoS generator have all of the functionality of a fixed QoS generator, and can be programmed to behave as a throughput limiter.

Limiters monitor initiator socket throughput, and throttles the socket to the programmed Bandwidth register value. This mechanism prevents the initiator from causing congestion for others.

The following table lists the registers that can be used to program limiter generators.

Table 19. Registers to program limiter generators

Register name	Byte offset	Field	Bits	Bit offset	Description
Priority	08h	P0	3	0	Priority of writes.
		P1	3	8	Priority of reads.
Mode	0Ch	Mode	2	0	0 = Fixed 1 = Limiter 2 = Bypass

Table continues on the next page...

Table 19. Registers to program limiter generators (continued)

Register name	Byte offset	Field	Bits	Bit offset	Description
Bandwidth	10h	Bandwidth	if (useExtRef) 13 + $\log_2$(2 × wData / 8) else 9 + $\log_2$(2 × wData / 8)	0	Throughput threshold in 1/256th-byte-per-cycle units.
Saturation	14h	Saturation	10	0	Maximum unregulated burst.
ExtControl	18h	SocketQosEn	1	0	Initiator drives QoS
		ExtThrEn	1	1	Use external threshold.
		IntClkEn	1	2	Use external threshold.

Throughput is programmed by setting register Bandwidth in 1/256th-byte-per-cycle units. Consequently, to program a desired throughput rate BW, in megabytes per second, the register must be set to:

```
Bandwidth = 256 × BW / F   // F is the initiator clock frequency in MHz.
```

where F is the initiator clock frequency in MHz. When using an external reference clock, the register must be set to:

```
Bandwidth = 256 × BW / (2 × Fext)   // Fext is the external clock frequency in MHz.
```

In certain cases, some initiators can generate bursty activity. Register saturation sets the maximum number of bytes that can be requested at full speed before the limiter triggers. The register must be set to the desired number of bytes divided by 16.

6.2.2.2.1.3 Programming regulator generators

An initiator NIU with a regulator type of QoS generator has all the functionality of a limiter QoS generator, and can be programmed to behave as a throughput regulator. A regulator monitors throughput usage of the initiator and adjusts the priority of its data transfers based on how much throughput the initiator has been using. Unlike the limiter's strict throttle, which aggressively blocks a initiator from exceeding a throughput quota, a regulator gently allows the initiator to use excess available throughput, but at a priority level that will ensure that other competing initiators have their necessary share.

Table 20. Registers to program regulator generators

Register name	Byte offset	Field	Bits	Bit offset	Description
Priority	08h	P0	3	0	High throughput priority level (or priority of writes).
		P1	3	8	Low throughput priority level (or priority of reads.
Mode	0Ch	Mode	2	0	0 = Fixed 1 = Limiter 2 = Bypass 3 = Regulator
Bandwidth	10h	Bandwidth	if (useExtRef) 13 + $\log_2$(2 × wData / 8) else	0	Throughput threshold in 1/256th-byte-per-cycle units.

Table continues on the next page...

Table 20. Registers to program regulator generators (continued)

Register name	Byte offset	Field	Bits	Bit offset	Description
			$9 + \log_2(2 \times wData / 8)$		
Saturation	14h	Saturation	10	0	Maximum unregulated burst.
ExtControl	18h	SocketQosEn	1	0	Initiator and regulator drive QoS.
		ExtThrEn	1	1	Use external threshold.
		IntClkEn	1	2	Use special clock reference.

A regulator has the same P0 and P1 fields of the Priority register: P0 stores the priority level to use for all data transfers when throughput is above the threshold, and P1 stores the priority level to use for all data transfers when throughput is below the threshold. Generally, the priority in P0 is a lower number than in P1. Both read and write transfers have the same priority when the QoS generator is configured in regulator mode.

Some initiators assert a priority level for data transfers apart from the regulator. Asserting the SocketQosEn bit of the ExtControl register causes the regulator to use the greater of either the initiator or the QoS generator priority levels. This behavior can be used for initiators that have internal buffering where data transfer priority is a function of the fullness of the buffer.

6.2.3 Resilience features

6.2.3.1 Fault aggregation and reporting

6.2.3.1.1 Unit fault aggregation and reporting

When the various resilience mechanisms detect a hardware fault in a unit, they assert a fault output during one clock cycle. At each NoC unit level, all faults are sent to a single checker/BIST module, which aggregates the faults and reports them to a single controller per NoC power domain, which interacts with system software.

The following figure depicts the functionality of the checker/BIST module (in a case of duplicated unit).

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a4b-b251-5

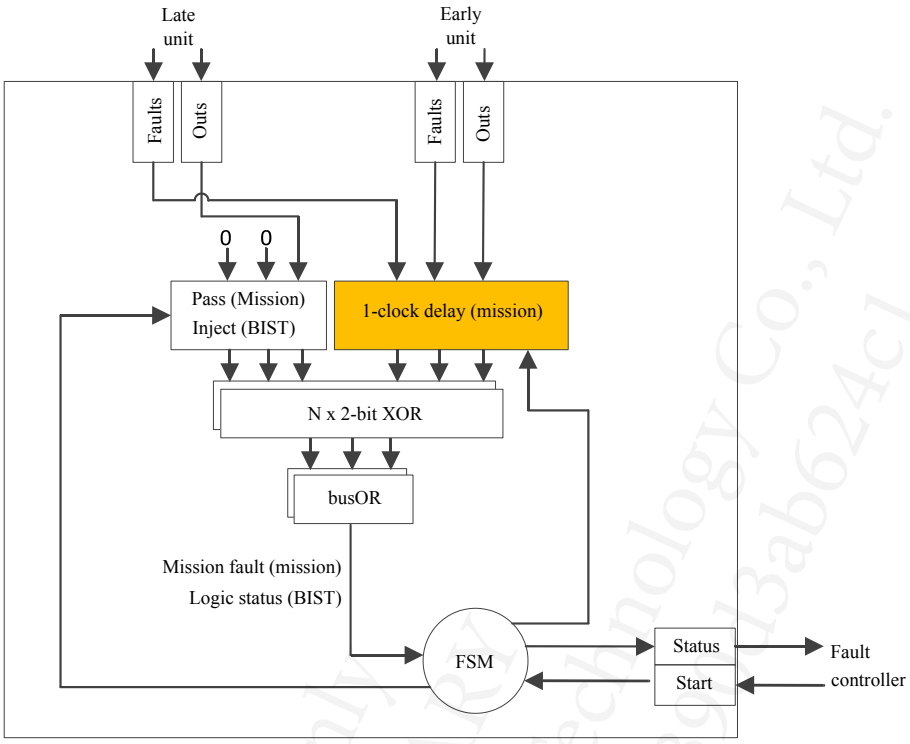


Figure 14. Checker/BIST module functionality

Checker/BIST modules are automatically instantiated by the NoC software synthesis engine in all units that produce faults according to the fault parameter values.

6.2.3.1.1.1 Mission mode

In mission mode, the fault controller reports faults from the checker/BIST modules under its control, typically using the MissionInt output as an interrupt.

The granularity of mission fault reporting is per checker/BIST, i.e. registers indicate which checker/BIT module, and consequently which associated NoC unit has experienced a hardware fault. There is no way to discriminate inside a unit which resilience mechanism triggered the reported fault, neither how many faults were reported.

After reset, the fault controller starts in mission mode.

6.2.3.1.1.2 BIST mode

The checker/BIST module implements a BIST function that detects possible faults in the fault aggregation logic.

The checker/BIST module goes into BIST mode under request by the fault controller on its Start input. In BIST mode, the module does not report unit faults, but performs the following steps:

- Applies patterns in the register that check the fault detection and aggregation logic.
- Reports any failure of the patterns detected by the FSM to the fault controller on the Status output.
- Reports successful completion of the BIST sequence to the fault controller on the Status output and goes back into mission mode, with a Status indicating absence of fault, until a new unit fault is detected by the mission mode function.

6.2.3.1.2 Fault controller

A single fault controller is automatically instantiated per power domain that contains units with fault outputs.

The fault controller performs the following functions:

- In mission mode, reports to the system software the NoC units faults detected by the resilience mechanisms.
- Under system software control, triggers BIST sequences of the fault reporting chain, to detect Latent fault in all checker/BIST modules under its control.

The fault controller has the following single-bit outputs, that can be exported to the NoC boundary:

- MissionInt – Reflects the value of the MissionFault field of the Faults register, when enable is set. When high, indicates that a mission fault has been detected in one of the units, and more information is available in the fault controller registers.
- LatentInt – Reflects the value of the BistDone field of the BistDone status register, when BistDoneEn field of IntEn register is set. When high, indicates that the BIST sequence is done, so the software can read the LatentFault field of the Faults status register to check if BIST sequences have been successful or not. More information is available in the fault controller LatentFault 0-3 status registers.

6.2.3.1.2.1 Mission mode

Mission mode is the normal operating mode of the checker/BIST module, in which it monitors the unit itself and aggregates results into a single Status, stored in a local FSM before being issued to the fault controller.

After reset of the unit, the checker/BIST module is in mission mode, and output Status is 0, indicating absence of a fault.

6.2.3.1.2.2 BIST mode

The fault controller enters BIST mode under software control, using the BistCtl register.

In BIST mode, the controller starts a BIST sequence in parallel on all the checker/BIST modules under its control. Latent fault reporting granularity is per checker/BIST, like mission faults.

NOTE

Until BIST sequence has completed, the system is fully functional but blind to any possible mission fault. The interval during which it is blind to mission fault depends on the timeout register settings of the controller, themselves depending on the largest number of inputs of all checker/BIST modules under its control, see timeout registers below.

After a BIST sequence has completed successfully for all checker/BIST modules, the controller resumes mission mode.

For more information about behavior after BIST failures, see Programming model.

6.2.3.1.2.3 Programming model

The programming model of the fault controller is made complex by the possibility of latent faults in the checker-BIST, or the controller itself, or both. Note that before programming the fault controller, the system should verify the integrity of the service network path by at least reading, for example, its identification registers, and checking their values.

6.2.3.1.2.3.1 Operating in mission mode

To operate in mission mode, the system should either set register IntEn field MissionFaultEn to 1, in order to receive an interrupt on MissionInt controller output, or regularly poll register Faults field MissionFault.

When a mission fault occurs, detected either by the interrupt or the polling method, the system should:

- Verify that Faults field MissionFault is set, when not already done by polling.
- Optionally identify the source of the fault in registers MissionFault 0–3.
- Take appropriate system action, such as running diagnostics.
- Optionally run a BIST sequence to attempt to clear the mission fault. This can permanently succeed if the fault was a transient fault that did not wholly corrupt the state of the system.

NOTE

Unless the sources of faults are reset, or the controller is put in BIST mode, faults continue to be monitored, and other fault sources may become active in registers MissionFault 0–3.

6.2.3.1.2.3.2 Clearing mission faults

If a unit has experienced a transient fault that did not corrupt the whole NoC state, for example, if the fault only affected a data value written into a memory target of the NoC, then by following the appropriate procedure (see To clear mission fault without resetting the faulty unit), it may be possible to clear the fault, although success is not guaranteed.

6.2.3.1.2.3.3 Running successful BIST sequences

A BIST sequence is successful because it correctly detects whether or not there is a latent fault in the fault reporting chain. A BIST sequence will finish successfully even if the FSM of a checker BIST is faulty, in which case a latent fault is reported for the module concerned.

6.2.3.1.2.3.4 Identifying unsuccessful BIST sequences

Identifying unsuccessful BIST sequences When fault controller FSMs are themselves faulty, the BIST sequence may not finish successfully, that is, some unexpected events during the sequence may occur:

- After writing 1 in register BistCtl field BistStart, the polling or interrupt method fails to indicate BIST sequence completion in register BistDone field BistDone before register BistTO1 controller clock cycles.
- After a successful indication that a successful BIST sequence ran correctly in BistDone field BistDone, the controller fails to indicate that it is returned to mission mode in BistDone field MissionMode before BistTO2 controller clock cycles.
- After writing 1 in BistCtl field BistDoneClr, the BistDone field BistDone value is not read back to be 0.

When such events are detected, the fault controller itself must be assumed to have experienced a hardware fault.

Although such a hardware fault does not compromise the operation of the SoC, it may prevent detection of further mission or latent faults, or both.

An attempt can be made to clear mission faults by resetting the power domain containing the controller (see To clear mission fault after resetting the faulty unit). This may be successful if the fault controller experienced a transient fault, because a successful BIST sequence after reset of the power domain indicates that the fault controller is not permanently faulty.

6.2.3.1.2.4 To clear mission fault without resetting the faulty unit

1. Run a BIST sequence. This will clear the state and the Status output of the checker or BIST that reported a fault, assuming that the checker or BIST itself is functional.
2. Write 1 in register IntClr field MissionFaultClr.
3. Either a new mission fault is quickly reported (system dependent), in which case the system is probably permanently faulty, or the system must decide any further action required to resume mission mode.

6.2.3.1.2.5 To clear mission fault after resetting the faulty unit

Since the controlled units are in the same power domain as their controller, resetting the unit usually involves resetting the whole power domain, including the fault controller itself. The action sequence is:

1. Use the domain power controller and bring it to an idle state, then reset the power domain.

   NOTE

   This assumes that the NoC is configured to support such capability, and that the reported hardware fault does not compromise the operation of the power controller.

2. The checker field BIST is reset and the fault controller restarts in mission mode.
3. Either a new mission fault is quickly reported (system dependent), in which case the system is probably permanently faulty, or the system must decide any further action required to resume mission mode.

6.2.3.1.2.6 To run a successful BIST sequence

1. Program registers BistTO1 and BistTO2 to appropriate values. Set register IntEn field BistDoneEn according to the chosen method to detect the end of the BIST sequence: when set to 0, the system must poll register BistDone field BistDone and wait for the field to be set to 1. When set to 1, the system can wait for an interrupt on LatentInt.
2. Write 1 in register BistCtl field BistStart to start the BIST sequence.
3. Once BIST completion is detected by polling or the interrupt method referred to in step 2, check the value of register Faults field LatentFault:
   - If 0, the BIST run was successful and no latent fault was detected.
   - If 1, the BIST run was successful and a latent fault was detected, as indicated by registers LatentFault 0–3.
4. Write 1 into register BistCtl field BistDoneClr.
5. Poll register BistDone fields BistDone and MissionMode until they show values of 0 and 1 respectively. This should happen before BistTO2 clock cycles but after the BIST sequence finishes its run.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 7
Ncore

7.1 Chip-specific Ncore information

7.1.1 Ncore interface units

This table provides a mapping between the Ncore interface units and the modules in this chip with which they are connected.

Ncore interface unit	Chip modules	Type of agent	Notes
CAIU0	A53 Cluster0	Fully-coherent agent	Connects to Ncore via AMBA ACE interface.
CAIU1	A53 Cluster1	Fully-coherent agent	Connects to Ncore via AMBA ACE interface.
NCBU0	SerDes_1, PFE	IO-coherent agent	Originating traffic is routed through the NoC_1 to the Ncore via a shared AXI4 interface.
NCBU1	SerDes_0, GMAC_0	IO-coherent agent	Originating traffic is routed through the NoC_0 to the Ncore via a shared AXI4 interface
CMIU	—	Memory agent	Routes the transactions to the NoC_0

7.2 Introduction

Ncore is a cache-coherent interconnect module. It enables the integration of heterogenous coherent agents and non-coherent agents in a chip. It processes transactions with coherent access semantics from various fully-coherent and IO-coherent masters, targeting shared resources.

7.3 Features

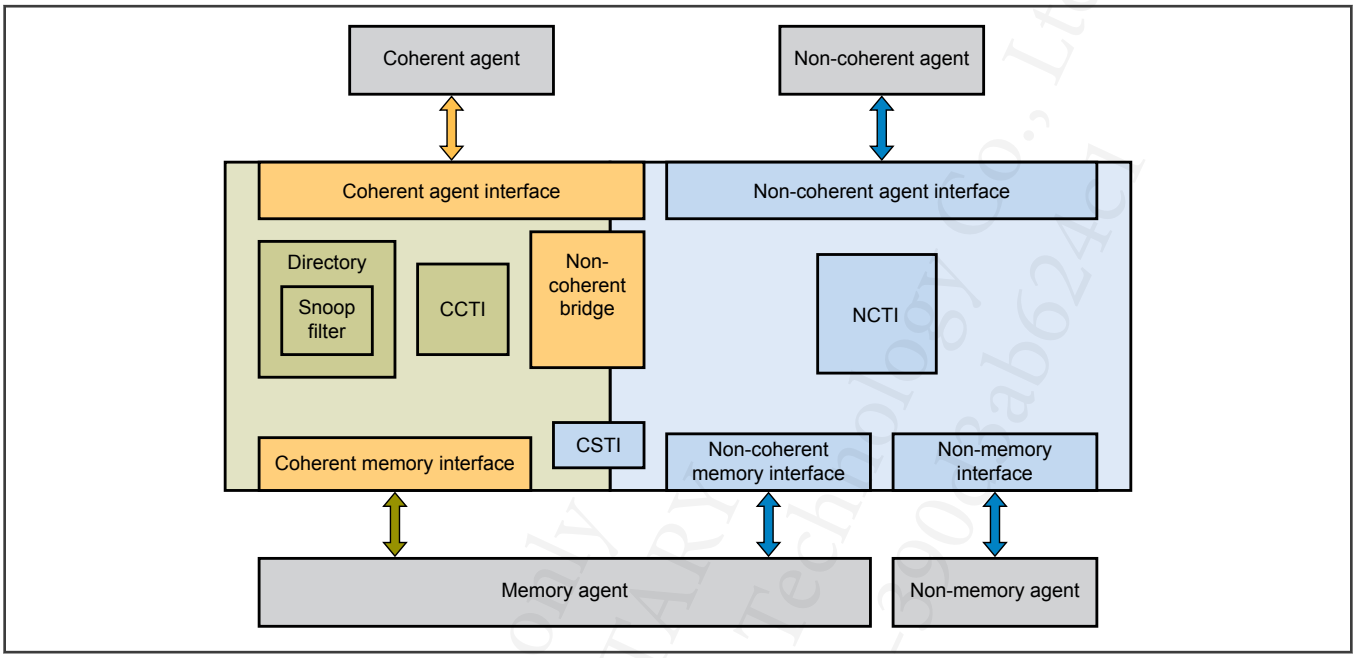
- Configurable heterogenous agents that are:
  - Fully-coherent
  - IO-coherent
  - Non-coherent
- Configurable snoop filter
- Heterogenous coherence models including:
  - Cache state models
  - Transaction processing models
  - Transaction types
- Resiliency package that provides additional functional safety features, such as:
  - Configurable link data protection
  - Hardware unit duplication

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cc462e5-4821-488-b251-390d3a5e4c1

— Memory protection

— Functional safety checker

7.4 Block diagram



7.5 Signal description

Ncore has a single clock domain. All agent interfaces and transport datapath operate synchronously at the same operating frequency.

7.6 Functional description

7.6.1 Caches and cache coherency

Caching is a common performance feature that buffers copies of data (caches) near to the processing units on the chip that manipulates data. Caches consist of relatively small memory arrays and have a short access time. This increases system performance by reducing average memory latency along two dimensions:

- Data which are accessed previously and are likely to be accessed again, and data that are accessed frequently are allocated in the cache. This decreases the waiting time that a processing unit may spend for receiving the data.
- Data are allocated to the caches in quantities (cachelines) larger than the typical size of memory access performed by a processing unit. This prefetch effect reduces the latency for subsequent memory accesses.

The presence of caches creates multiple copies of the same data. All copies must remain consistent. This complete process is termed as cache coherency.

7.6.2 Ncore agents

This chip consists of following set of agents that communicate through the Ncore:

- Master agent: This block initiates read and write transactions to a single physical address. These agents are classified based on the type of transactions they issue:
  - Non-coherent agent: For a given physical address, a non-coherent agent directly accesses the data in the corresponding memory location.

  - Coherent agent: For a given physical address, a coherent agent may access the data either in the corresponding memory location or in one of the caches in the system. Coherent agents are further classified as:
    - Fully-coherent agent: The master agent that performs the coherent transaction obtains a copy of the data that is required to remain coherent with respect to coherent transactions from other master agents.
    - IO-coherent agent: The master agent obtains a copy of the data that is not required to remain coherent.
- Slave agent: This block services read and write transactions for a given set of physical addresses. These agents are classified based on how they access the read and write transaction results:
  - Memory agent: It provides access to memory locations. A read access to the physical address by a memory agent returns the last value written.
  - Non-memory agent: It provides access to other forms of system functionality. A read access to the physical address by a memory agent may not return the last value written, and multiple reads may return different values. A read and write access may cause side effects like raising an interrupt or initiating a hardware state machine.

For mapping between the modules on this chip and their respective Ncore agent classification, see the chip-specific Ncore information.

7.6.3 Ncore interfaces

The Ncore agents connect with Ncore through the following interfaces.

7.6.3.1 Coherent agent interface (CAI)

CAI provides a means for a fully-coherent agent to be connected to the Ncore. The CAI behaves as a fully-coherent slave (ACE slave).

CAI classifies agent transactions as either non-coherent or coherent. Non-coherent agent transactions are delivered directly to the NCTI or an external interconnect. Coherent agent transactions are decoded and translated into individual protocol transactions. Each protocol transaction allocates an entry in an outstanding transaction table (OTT) that tracks the outstanding protocol transactions as they progress through the various phases of the coherence protocol. In addition, coherent read and write transactions also allocate an OTT data buffer for staging read data or write data, respectively.

In CAI, each functional port has an analogous structural unit—Coherent agent interface unit (CAIU).

The CAI in this chip has the following properties:

- ACE protocol
- 32-bit address
- 6-bit transaction ID width
- 128-bit data width
- Supports distributed virtual memory (DVM) on AR and AC channels
- Supports outstanding transactions

The CAI in this chip has the following configuration for supporting outstanding transactions:

- Supports up to 16 outstanding protocol transactions
- Supports 16 data buffers for outstanding protocol transactions, protected by SECDED ECC
- Supports up to four command messages per port and directory pair
- Supports up to four cache-to-cache data messages per port
- Supports heterogeneous mix of coherent and non-coherent transactions sharing the same transaction ID. Only one non-coherent transaction may have the same transaction ID

7.6.3.2 Coherent memory interface (CMI)

CMI provides a mechanism for the Ncore to access a memory agent coherently. The CMI has an analogous structural unit—Coherent memory interface unit (CMIU).

The CMI in this chip has the following properties:

- AXI protocol
- 40-bit address
- 5-bit transaction ID width
- 128-bit data width

7.6.3.3 Non coherent bridge (NCB)

NCB translates a non-coherent transaction into one or more coherent transactions that access a directory cacheline. The NCB behaves as an IO-coherent master.

In an NCB, each functional port has an analogous structural unit—Non-coherent bridge unit (NCBU).

The NCB in this chip has the following properties:

- AXI protocol
- 32-bit address
- 6-bit transaction ID width
- 128-bit data width
- Supports outstanding transactions

The NCB in this chip has the following configuration for supporting outstanding transactions:

- Up to 16 outstanding protocol transactions
- 16 data buffers for outstanding protocol transactions, protected by SECDED ECC

7.6.3.4 Directory

The directory provides a point of serialization for establishing transaction ordering and sequences coherence operations and memory accesses. The directory is configured with a snoop filter. See Snoop filter for more information.

The directory has an analogous structural unit—Directory unit (DIRU).

The directory in this chip has the following properties:

- Supports a 64-byte line size
- Supports up to 48 active protocol transactions
- Four exclusive monitors per port

7.6.3.4.1 Snoop filter

This unit is configured within the DIRU to reduce the system bandwidth needed to enforce coherence. The DIRU accesses the snoop filter in parallel with its regular operations, to determine the state of the caching agent, and thus identifying the required coherence operations.

A snoop filter is configured to store different types of information to make trade-offs between snoop filter area and coherence bandwidth. A snoop filter is of two types, depending upon the type of information stored:

- Null filter
- Tag filter

The null filter stores no information about its associated caching agents. For every coherent transaction, the directory broadcasts snoop messages to the caching agents.

Snoop message - A snoop filter transmits coherency operations to caching agents through snoop messages.

The tag filter is a set-associative structure that stores information about previous cached transactions. This ensures that only the necessary coherence operations are performed on those caching agents. Tag filter achieves this by:

- Tagging entries by physical address
- Storing cacheline validity
- Storing ownership information about its associated caching agents

Snoop filter is further classified into two types based on the information stored in a tag filter entry:

- Presence vector filter:

  Each caching agent is represented by a bit that indicates whether the caching agent possesses a valid copy of the cacheline.
- Owner-sharer filter:

  The tag filter categorizes the caching agents into two types—owner and sharer.

  Owner- Each caching agent that is represented by a pointer field, indicates that the caching agent is an owner. An owner caches a valid copy of data that may be either dirty or unique.

  Sharer- Each caching agent that is represented by a bit, indicates that the caching agent is a sharer. A sharer caches a valid, clean and non-unique copy of data.

The snoop filter in this chip is a tag filter of the owner-sharer type. It has the following features:

- Organized as a 12-way set associative caching agent with 2048 sets per way
- Supports up to eight outstanding snoop messages per directory port and caching agent port pair.

7.6.4 Transport interconnect

The transport interconnect provide a means for the Ncore interfaces to communicate with each other. It also enables communication with and between the non-coherent agents. Ncore uses the following transport interconnects:

- Non-coherent transport interconnect (NCTI): It provides connection between the non-coherent master and the coherent bridges, and between the non-coherent ports of a CAIU. The functionality of NCTI is performed by NoC in this chip.
- Coherent-capable transport interconnect (CCTI): It provides communication between the coherent units to communicate with each other. The functionality of CCTI is performed by NoC in this chip.
- Control and status transport interconnect (CSTI): It transport non-coherent transactions to the control and status registers in the Ncore coherent units.

7.6.5 Storage protection

Storage structures in this chip, such as tag filters and transaction table data buffers, are configured with a protection scheme to enable error detection and correction capability. This protection scheme is called SECDED entry—Single-error correction, double-error detection code per entry

With the SECDED entry storage protection scheme, Ncore can:

- Detect double-bit errors in an entry. These are uncorrectable errors.
- Correct single-bit errors in an entry. These are correctable errors.

Each structural unit connected to the Ncore interfaces logs and signals correctable and uncorrectable errors separately. Ncore aggregates all the error interrupts from the units and outputs two interrupt signals—a combined correctable interrupt signal and a combined uncorrectable interrupt signal.

7.6.5.1 Error types

An error is defined to be a deviation from correct or expected system behavior. Following are the various errors that Ncore interacts with:

- Undetectable error—deviation that Ncore is unable to detect.
- Detectable error—deviation that Ncore is able to detect. These errors are classified as follows:
  - Data corruption error—an uncorrectable error that a transaction encounters in a data storage (for example, data buffers).
  - Address corruption error—an uncorrectable error that a transaction encounters in an address storage (for example, tag filters).
  - Transport error—an uncorrectable error that a transaction encounters in a transport interconnect (for example, CCTI).

7.6.6 Resiliency

Ncore has a highly reliable resiliency feature that enables functional safety and fault tolerance for the chip. It has the following units to support its resiliency feature:

- Fault controller to monitor and log faults detected in the interconnect
- Built-in self-test (BIST) finite state machine (FSM) that enables checking of the Ncore fault detection logic

Ncore has the following properties to support its resiliency feature:

- Duplication of CAI, CMI, DIR and NCB
- Protects datapath packet including packet validity checker, header parity and payload parity
- Protects control register parity
- Transport timeout detection with 32K clocks timeout period

The following figure illustrates a high-level overview of the different elements of Ncore enabled with the resiliency feature:

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8ab62401

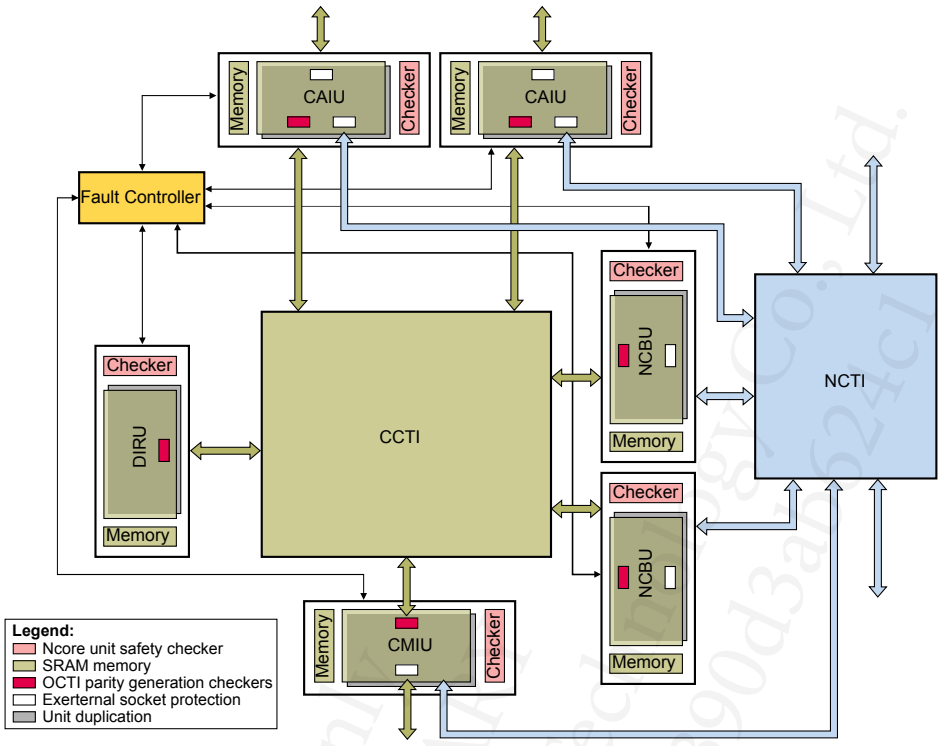


Figure 15. Ncore elements with resiliency

7.6.6.1 Component logic duplication

Ncore uses a mechanism called “component logic duplication” for providing its resiliency feature. This capability replicates the logic for all the Ncore components:

- CAI
- CMI
- DIR
- NCB

It creates logic for two identical modules of the logical component. The two identical modules are functional component and duplicated component. The replicated logic operates in lock-step with the original component logic and it is configured to operate one clock cycle apart.

Ncore uses an additional logic—checking logic—known as safety checker to ensure that the original and replicated logic operates in lockstep.

Ncore reports two types of faults to the fault controller:

- Mission fault

  The checker component receives one cycle delayed version of the same inputs as the functional component. The safety checker module receives the functional component outputs and delays it by one cycle, then compares them with the checker component outputs. Any discrepancy is reported as a mission fault.

  After a mission fault is detected, it remains logged inside the checker component until a BIST sequence clears it (See the BIST sequence).

- Latent fault

The safety checker module implements two identical comparator trees checking simultaneously that the output of the checker component is identical to the delayed version of the output of the functional component. A mismatch between the output of the comparator trees is reported as a latent fault.

7.6.6.2 Fault controller

The fault controller component monitors and logs faults reported by all safety checkers in the system, and drives an interrupt signal to notify the host system that a fault in the coherent interconnect has occurred. The fault controller hosts a register interface which allows you to identify the source of detected faults. (See Functional safety controller (FSC) register descriptions)

You can also use the fault controller to manage the BIST for the fault reporting logic. The fault controller has a BIST FSM that you use to:

- Communicate with the safety checkers
- Check the fault reporting logic
- Clear the mission and latent faults

After any faults are detected, you can execute a BIST sequence to debug the cause of the faults.

The SCBISTC register provides you the option of executing the BIST sequence automatically or manually depending on your requirement. If the automatic running of the BIST sequence fails, you can execute each step one by one to find out which step fails.

See BIST sequence for the procedure.

7.6.6.3 BIST sequence

The BIST sequence is as follows:

1. Reset safety checkers faults

   You reset all the safety checkers in this step.

   This step clears all the mission and latent faults from the safety checkers. On completion of this step, the SCMF*n*[MISSIONFAULT] and SCLF*n*[LATENTFAULT] must read as 0.

2. Force functional component comparator tree error

   You program the functional component output to 1 in this step.

   This step sets all the mission and latent faults from the safety checkers. On completion of this step, the SCMF*n*[MISSIONFAULT] and SCLF*n*[LATENTFAULT] must read as 1.

3. Reset safety checker faults, and force duplicated component comparator tree error

   You reset all the safety checkers and program the duplicated component output to 1 in this step.

   This step clears all the mission faults and sets all the latent faults from the safety checkers.

   On completion of this step, SCMF*n*[MISSIONFAULT] must read as 0 and SCLF*n*[LATENTFAULT] must read as 1.

4. Reset safety checker faults, and force both functional and duplicated component comparator tree error

   You reset all the safety checkers and program both the functional component and duplicated component outputs to 1 in this step.

   This step sets all the mission faults and clears all the latent faults from the safety checkers.

   On completion of this step, SCMF*n*[MISSIONFAULT] must read as 1 and SCLF*n*[LATENTFAULT] must read as 0.

5. Reset safety checkers faults

   You reset all the safety checkers in this step.

   This step clears all the mission and latent faults from the safety checkers. On completion of this step, SCMF*n*[MISSIONFAULT] and SCLF*n*[LATENTFAULT] must read as 0.

NOTE

Performing BIST sequence latches mission and latent faults in FCCU. You must clear the faults by running the FCCU fault clear mechanism. See the FCCU chapter in this Reference Manual for details.

7.6.7 Coherent read/write transactions

The various components in the Ncore work together to communicate and enforce cache coherence on shared transactions. Each coherent transaction initiates a series of protocol transactions and organizes all the operations to carry out the coherent transaction while ensuring the following:

- Maintains coherence amongst all coherent agents in the chip
- Updates the DIRU that oversees the cache state of all coherent agents in the chip

7.6.7.1 Coherent read transactions

To issue a coherent read transaction, you issue a coherent transaction to the CAI. Ncore performs the following operations on this request:

1. The CAI issues a command to the directory.
2. Ncore access the snoop filter in the directory to determine if the read data is resident in the cache of another coherent master connected to Ncore.
3. The directory issues a snoop message to all the coherent masters indicating the type of coherence operation to be performed.
4. The coherent masters receiving the snoop message responds by indicating the result of the snoop transaction.
    - If the data is found in cache, it is called a directory hit.
    - If the data is not found in any cache, it is called a directory miss.

On a directory hit, the snooped agent forwards the data to the requesting agent.

On a directory miss, the directory issues a memory read transaction through the CMI. The directory fetches the requested data from the main memory and forwards it to the requesting agent.

7.6.7.2 Coherent write transactions

To issue a coherent write transaction, you issue a coherent transaction to the CAI. Ncore performs the following operations on this request:

1. The CAI issues a command to the directory.
2. The directory issues a snoop message to all the coherent masters to invalidate the copies of the data resident in all the caches.
3. The directory issues the write to main memory through the CMI.

7.7 Reset and initialization

NOTE

You must perform the following before executing any cache coherence management operation:

- Initialize Ncore
- Move the chip modules connected with CAIU0 and CAIU1 out of reset. See the chip-specific Ncore information to know about the chip modules that are connected with CAIU0 and CAIU1.

Each structural unit must be in operational state, before processing any transaction.

For a component to be in an operational state, each structural unit within the component must be in operational state.

After a component is in operational state, it can be in either of the following states:

- Offline state
- Online state

A component processes coherent transactions only if it is in an online state.

For a component to be in an online state, each structural unit within the component must be in online state.

A component can be in either of the following states with respect to processing transactions:

- Active state
- Inactive state

A component is in the active state when it is tracking state pertaining to outstanding transactions, otherwise it is in the inactive state. Additionally, when a component is in an offline state, it is in the inactive state with respect to coherent transactions; however, with respect to non-coherent transactions, the component may be in either of the states.

In a configuration with only one coherent agent connected to the CAIUs as active, snoop activity is not required to maintain coherency between the CAIUs. So, when you configure to disable any of the coherent agent, you must ensure disabling of snoop messages (by using DIRUCASE0[CASNPEN*n*]) and DVM messages (by using CSADSE0[DVMSNPEN*n*]) to the CAI.

The following sections describes the sequences that you must perform for transitioning the various components to online state.

7.7.1 CAI initialization

Perform the following procedure for transitioning the CAI to online state:

1. Ensure that all the CAIUs within the interface are in operational state: read the CAIUID register here to identify the features and configuration information of the corresponding CAIU.

   Perform this step for each CAIU if the discovery routine is not performed. See Discovery routine for its procedure.
2. For a fully-coherent agent interface, enable snoop messages to the CAI for each DIRU by writing 1 to the DIRUCASE0[CASNPEN*n*] field corresponding to the CAIUs within the interface.
3. For a CAI that supports ACE DVM transactions on the address coherency channel, enable DVM messages to the CAI for each CAIU within the interface by writing 1 to the corresponding CSADSE0[DVMSNPEN*n*] field.

After CAI is transitioned to operational state, it may enter active state with respect to non-coherent transactions, but it remains in offline state until the above procedure completes.

7.7.2 CMI initialization

To transition the CMI to online state, ensure that all the CMIUs within the interface are in operational state: read the CMIUID register to identify the features and configuration information of the corresponding CMIU.

Perform this for each CMIU if the discovery routine is not performed. See Discovery routine for its procedure.

7.7.3 NCB initialization

To transition the NCB to online state, ensure that all the NCBUs within the interface are in operational state: read the NCBUID register to identify the features and configuration information of the corresponding NCBU.

Perform this for each NCBU if the discovery routine is not done. See Discovery routine for its procedure.

7.7.4 Directory initialization

To transition the directory to online state, ensure that all the DIRUs within the interface are in operational state.

The snoop filter (tag filter) in the directory is in offline state. Following is the procedure to transition the snoop filter to online state:

1. Ensure that the directory is in online state.
2. Ensure the following points for each caching agent associated with the snoop filter:
   a. All outstanding coherent transactions that may allocate cacheline in the agent's cache are complete.

b. Entire cache of the agent is clean and invalidated.

c. All the remaining outstanding coherent transactions from the agent are complete.

d. No new coherent transaction is initiated by the agent.

3. Perform an initialize all entries maintenance operation for each snoop filter in each DIRU within the directory by writing the appropriate values to the DIRUSFMC register. See Snoop filter maintenance for the maintenance operations.
4. Poll the DIRUSFMA[MNTOPACTV] field for each snoop filter in each DIRU within the directory until the field value equals 0.
5. Enable snoop filtering for each snoop filter in each DIRU within the directory by writing 1 to the corresponding DIRUSFE[SFENn] field.

After the directory is in online state, agents may issue coherent transactions via CAIs or NCBs in the online state to any CMI in the online state.

7.7.5 Discovery routine

After all the structural units are in operational state, execute the discovery routine to determine the:

- Number of snoop filters
- Number of each type of unit
- Various unit associations

The discovery routine procedure is as follows:

1. Validate the coherent subsystem release version by using the CSID[RELVER] field.
2. Identify the number of snoop filters in the coherent subsystem by using the CSID[NUMSFS] field.
3. Identify the configuration of each snoop filter in the coherent subsystem by using the corresponding CSSFIDRa register.
4. Identify the number of CAIUs, NCBUs, DIRUs, and CMIUs by using the CSUID register.
5. Read the following CAIUID register fields for each CAIU:
    a. CAIID: To identify the CAI with which the CAIU is associated.
    b. CA: To learn whether the CAIU is associated with a caching agent (that is, a fully-coherent agent). If it is associated, then this field also indicates that a corresponding DIRUCASE0[CASNPEN*n*] field is present.
    c. SFID: If you learn that the CAIU is associated with a caching agent, use this field to identify the snoop filter with which the caching agent is associated. Skip to the next step if the CAIU is not associated with a caching agent.
    d. TYPE: To learn whether the CAIU is associated with a DVM agent. If it is associated, then this field also indicates that a corresponding CSADSE[DVMSNPEN*n*] field is present.
6. Read the NCBUID[NCBID] field for each NCBU to identify the NCB associated with it.
7. Read the following CMIUID register fields for each CMIU:
    a. CMIID: To identify the CMI with which the CMIU is associated.

Discovery routine does not take into account the state of coherent agent connected to the CAIUs. If one of the coherent agent associated with a CAIU is in an inactive state, you must configure Ncore to avoid snoop activity towards the inactive caching agent. So, when you transition a CAIU to offline state, you must disable snoop messages (by using DIRUCASE0[CASNPEN*n*]) and DVM messages (by using CSADSE0[DVMSNPEN*n*]) to the CAI.

7.8 Maintenance operations

Use maintenance operations to initialize caching structures in the coherent subsystem and to manage those caching structures relative to non-coherent agents. The set of caching structures includes any snoop filters (tag filters) in DIRUs. Each set include the following registers:

- A maintenance control register (DIRUSFMC)
- A maintenance activity register (DIRUSFMA)
- A pair of maintenance location registers (DIRUSFMLR0 and DIRUSFMLR1)

Perform a maintenance operation by executing the following procedure:

1. Write the appropriate maintenance location registers with the location of the entry on which the operation is to be performed (if required for the operation).
2. Write the appropriate maintenance control register to initiate the maintenance operation.
3. Poll the maintenance operation active field (MNTOPACTV) in the appropriate maintenance activity register until the field value equals 0.

Successful completion of step 3 completes the maintenance operation.

For each DIRU, you can perform only one maintenance operation at a time. Writes to the maintenance operation registers while the corresponding MNTOPACTV field value equals 1 is ignored.

A maintenance operation operates only on the caching structure implemented by a unit.

7.8.1 Snoop filter maintenance

For a given DIRU, following is a list of maintenance operations that you could perform on a snoop filter:

- Initialize all entries: This operation invalidates all entries in the snoop filter and any associated victim buffer, and initializes the error protection bits in the entries.
- Flush all entries: This operation issues recall transactions for all valid entries in the snoop filter and any associated victim buffer, and leaves all entries in the invalid state.
- Flush entry at set and way: This operation issues a recall transaction for a valid entry at the specified physical set and way in the snoop filter, and leaves the entry in the invalid state.
- Flush entry at address: This operation issues a recall transaction for a valid entry at the specified address in any snoop filter and any associated victim buffer, and leaves the entry in the invalid state.
- Flush all victim buffer entries: This operation issues a recall transaction for all valid entries in the victim buffer associated with the designated snoop filter, and leaves all entries in the invalid state

The DIRUSFMC register corresponding to a DIRU contains the following fields to control the maintenance operation on its snoop filter:

- SFMNTOP: Write appropriate value to this field to specify the type of maintenance operation to be performed.
- SFID: This field identifies the snoop filter on which maintenance operation is performed.
- SFSECATTR: This field identifies the security attribute (secure versus non-secure) of the address on which the snoop filter maintenance operation is performed. A value of 1 indicates a secured configuration.

For the flush entry at set and way maintenance operation, the DIRUSFMLR0 register corresponding to a DIRU contains the following fields to identify the snoop filter entry on which maintenance operation is performed:

- MNTSET: This field identifies the physical set on which maintenance operation is performed.
- MNTWAY: This field identifies the physical way on which maintenance operation is performed.

This operation accesses the snoop filter directly, independent of whether permutation-based interleaving has been enabled for set selection. Additionally, the DIRUSFMLR0[MNTWORD] and DIRUSFMLR1[MNTADDR] field values are set to 0.

For flush entry at address maintenance operation, the DIRUSFMLR0 and DIRUSFMLR1 registers corresponding to a DIRU represents a directory cacheline address within the coherent subsystem address window. Following is the procedure to form this address:

1. Mask off the physical address bits above the most significant bit of the coherent subsystem address window.
2. Shift the result right by the number of bits in the directory cacheline offset.

3. Write the lower order 32 bits of the result into the DIRUSFMLR0 register and the remaining higher order bits into the DIRUSFMLR1 register. There may not be any high order bits for certain system configurations, and in such cases, you do not write anything to the DIRUSFMLR1 register.

The table below summarizes the location specifications based on the type of maintenance operation performed.

Table 21. Location specification

DIRUSFMC[SFMNTOP]	Maintenance operation	DIRUSFMLR1[MNTADDR]	DIRUSFMLR0[MNTWORD]	DIRUSFMLR0[MNTWAY]	DIRUSFMLR0[MNTSET]
0h	Initialize all entries	0	0	0	0
1h-3h	Reserved	N/A	N/A	N/A	N/A
4h	Flush all entries	0	0	0	0
5h	Flush entry at set and way	0	0	Way	Set
6h	Flush entry at address	Directory cacheline address			
7h	Reserved	N/A	N/A	N/A	N/A
8h	Flush all victim buffer entries	0	0	0	0
9h-Fh	Reserved	N/A	N/A	N/A	N/A

Ncore indicates status (whether in progress or have completed) of a maintenance operation by writing to the DIRUSFMA[MNTOPACTV] field.

- A field value of 1 indicates that maintenance operation is ongoing.
- A field value of 0 indicates completion of the maintenance operation.

For the following maintenance operations, you must perform the operation in each DIRU to flush the entire snoop filter:

- Initialize all entries
- Flush all entries
- Flush all victim buffer entries

For the following maintenance operations you must perform the operation only in the appropriate DIRU:

- Flush entry at set and way
- Flush entry at address

7.9 Power management

You reduce power consumption of the system by transitioning component from an operational state to non-operational state.

Before transitioning the units in the component into the non-operational state, you must transition all structural units of a component to an offline state or to the inactive state for non-coherent transactions.

After transitioning components in the coherent subsystem from online state to offline state, preserve the following order to transition the components from the operational state to the non-operational state:

1. Any or all of the following, as required:
   - CAI
   - NCB

- CMI

2. Directory and any tag filters after all of the above are in the offline state

The following sections describes the sequences that you must perform for transitioning the various components to offline state.

7.9.1 CAI power management

Perform the following procedure for transitioning a CAI to offline state:

1. Ensure the following conditions are fulfilled:
   - For fully-coherent agent interface, all outstanding coherent transactions that may allocate cacheline in the agent's cache are complete.
   - For fully-coherent agent interface, entire cache of the agent is clean and invalidated.
   - All the remaining outstanding coherent transactions from the agent are complete.
   - No new coherent transaction is initiated by the agent.
2. For a fully-coherent agent interface, disable snoop messages to the CAI for each DIRU by writing 0 to the DIRUCASE0[CASNPEN*n*] field corresponding to the CAIUs within the interface.
3. For a fully-coherent agent interface, poll the DIRUCASA0[CASNPACTV*n*] field corresponding to the CAIUs within the interface for each DIRU until the field value equals 0.
4. For a CAI that supports ACE DVM transactions on the AC channel, disable DVM messages to the CAI for each CAIU within the interface by writing 0 to the corresponding CSADSE0[DVMSNPEN*n*] field.
5. For a CAI that supports ACE DVM transactions on the AC channel, poll the CSADSA[DVMSNPACTV*n*] field corresponding to the CAIUs within the interface until the field value equals 0.
6. Poll the CAIUTA[SNPACTV] field for each CAIU within the CAI until the field value equals 0.

   Additionaly, for non-coherent transactions, perform the following procedure for transitioning a CAI to inactive state:
7. Ensure that the CAI is in offine state.
8. Ensure the following conditions are fulfilled:
   - All outstanding coherent transactions from all the agent are complete.
   - No new coherent transaction is initiated by any agent.
9. Poll the CAIUTA[TRANSACTV] field for each CAIU within the CAI until the field value equals 0.

See CAI initialization for the procedure to transition the CAI back to online state.

7.9.1.1 Isolated power state

CAI supports an additional state for power management—isolated power state.

In this state:

- Coherent read and write transactions are translated into non-coherent transactions.
- Certain other coherent transactions are terminated within CAI.

In the isolated power state, CAI does not require any additional coherent subsystem resources to process native agent transactions, thus it enables you to transition all other components in the coherent subsystem into offline state. The components in the offline state may then be transitioned into the non-operational state, depending on the configuration of the clock and voltage domains in the system.

The system power state in which a single CAI is in the isolated state and the rest of components in the coherent subsystem are in the offline state is known as the single-agent low-power state.

Perform the following procedure for transitioning a CAI to isolated power state:

1. Ensure the following conditions are fulfilled:

- All other CAIs are in the offline state.
- All NCBs are in the offline state.

2. For a fully-coherent agent interface, disable snoop messages to the CAI for each DIRU by writing 0 to the DIRUCASE0[CASNPEN*n*] field corresponding to the CAIUs within the interface.
3. For a fully-coherent agent interface, poll the DIRUCASA0[CASNPACTV*n*] field corresponding to the CAIUs within the interface for each DIRU until the field value equals 0.
4. For a CAI that supports ACE DVM transactions on the AC channel, disable DVM messages to the CAI for each CAIU within the interface by writing 0 to the corresponding CSADSE0[DVMSNPEN*n*] field.
5. For a CAI that supports ACE DVM transactions on the AC channel, poll the CSADSA[DVMSNPACTV*n*] field corresponding to the CAIUs within the interface until the field value equals 0.
6. Poll the CAIUTA[SNPACTV] field for each CAIU within the CAI until the field value equals 0.
7. Enable isolated power state for each CAIU within the CAI by writing 1 to CAIUTC[ISOLEN].
8. Poll the CAIUTA[COHACTV] field for each CAIU within the CAI until the field value equals 0.

After the CAI enters the Isolated power state, perform the following procedure to enter the single-agent low-power state:

1. Ensure the following conditions are fulfilled:
   - All other CAIs are in the offline state.
   - All NCBs are in the offline state.
2. Transition the CMI and Directory to the offline state.

You can also transition the components in the offline state into the non-operational state.

Perform the following procedure to exit the single-agent low-power state:

1. Transition the CMI and Directory to the online state.
2. Transition the CAI in Isolated power state to the offline state.

   Ncore writes 0 to CAIUTC[ISOLEN] on performing this step.

You can transition the CAIs and NCBs to the online state after exiting the single-agent low-power state.

7.9.2 NCB power management

Perform the following procedure for transitioning an NCB to offline state:

1. Ensure that the following conditions are fulfilled:
   - All outstanding non-coherent transactions from all the agents are complete.
   - No new non-coherent transaction is initiated by any agent.
2. Poll the NCBUTA[TRANSACTV] field for each NCBU within the interface until the field value equals 0.

See NCB initialization for the procedure to transition the NCB back to online state.

7.9.3 CMI power management

Perform the following procedure for transitioning a CMI to offline state:

1. Ensure that the following conditions are fulfilled:
   - All outstanding coherent transactions from all the agents are complete.
   - No new coherent transaction is initiated by any agent.
2. Poll the CMIUTA[TRANSACTV] field for each CMIU within the interface until the field value equals 0.

See CMI initialization for the procedure to transition the CMI back to online state.

7.9.4 Directory power management

Perform the following steps for transitioning a tag filter in a directory to online state:

1. Disable snoop filtering for each DIRU by writing 0 to the DIRUSFE[SFENn] field corresponding to the snoop filter.

Perform the following procedure for transitioning a directory to offline state:

1. Ensure that the following conditions are fulfilled:
   - All outstanding coherent transactions from all the agents are complete.
   - No new coherent transaction is initiated by any agent.
2. Poll the DIRUTA[TRANSACTV] field for each DIRU within the interface until the field value equals 0.

See Directory initialization for the procedure to transition the directory back to online state.

7.10 Memory map and register definition

Only full–word (32-bit) accesses are allowed on all registers in the Ncore memory space.

7.10.1 Coherent agent interface unit (CAIU) register descriptions

7.10.1.1 CAIU memory map

CAIU0 base address: 5040_0000h

CAIU1 base address: 5040_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	CAIU Transaction Control (CAIUTC)	32	RW	0000_0001h
4h	CAIU Transaction Activity (CAIUTA)	32	RO	0000_0000h
100h	CAIU Correctable Error Control (CAIUCEC)	32	RW	0000_0000h
104h	CAIU Correctable Error Status (CAIUCES)	32	W1C	0000_0000h
108h	CAIU Correctable Error Location Register 0 (CAIUCELR0)	32	RW	0000_0000h
10Ch	CAIU Correctable Error Location Register 1 (CAIUCELR1)	32	RW	0000_0000h
124h	CAIU Correctable Error Status Alias (CAIUCESA)	32	RW	0000_0000h
140h	CAIU Uncorrectable Error Control (CAIUUEC)	32	RW	0000_0000h
144h	CAIU Uncorrectable Error Status (CAIUUES)	32	W1C	0000_0000h
148h	CAIU Uncorrectable Error Location Register 0 (CAIUUELR0)	32	RW	0000_0000h
14Ch	CAIU Uncorrectable Error Location Register 1 (CAIUUELR1)	32	RW	0000_0000h
164h	CAIU Uncorrectable Error Status Alias (CAIUUESA)	32	RW	0000_0000h
FFCh	CAIU Identification (CAIUID)	32	RO	See description

NXP CONFIDENTIAL PROPRIETARY Preliminary Draft for FDA only Beijing Hainachuan Automotive Technology Co., Ltd. c8c46a8-8248-b251-390d3ab624d1

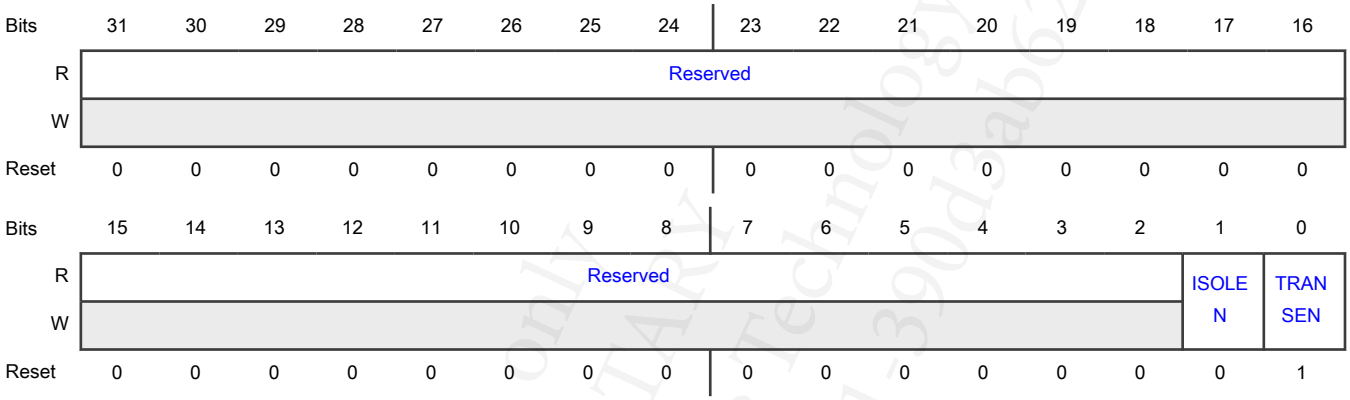
7.10.1.2 CAIU Transaction Control (CAIUTC)

Offset

Register	Offset
CAIUTC	0h

Function
Use this register to enable agent transactions in a CAIU.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 ISOLEN	Agent Isolation Enable Write a 1 to this field to enable isolation mode.
0 TRANSEN	Agent Transaction Enable Write a 1 to this field to enable agent transactions.

7.10.1.3 CAIU Transaction Activity (CAIUTA)

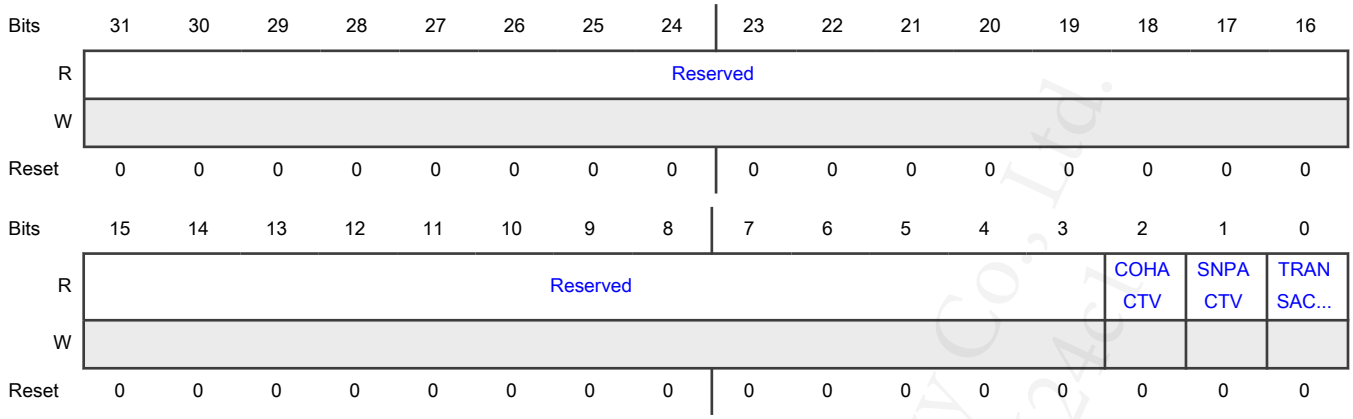
Offset

Register	Offset
CAIUTA	4h

Function
Use this register to learn whether certain protocol transactions are active in a CAIU.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHU Automotive Technology Co., Ltd.
c8462e5-4021-48a8-b251-390dab624c1

Diagram



Fields

Field	Function
31-3 —	Reserved
2 COHACTV	Coherent Transaction Active Ncore writes a 1 to this field when the unit is performing any activity related to the coherent transactions, and is cleared otherwise.
1 SNPACTV	Snoop Transaction Active Ncore writes a 1 to this field when the unit is performing any activity related to snoop messages, and is cleared otherwise.
0 TRANSACTV	Transaction Active Ncore writes a 1 to this field when the unit is performing any activity related to the native agent coherent transactions, and is cleared otherwise.

7.10.1.4 CAIU Correctable Error Control (CAIUCEC)

Offset

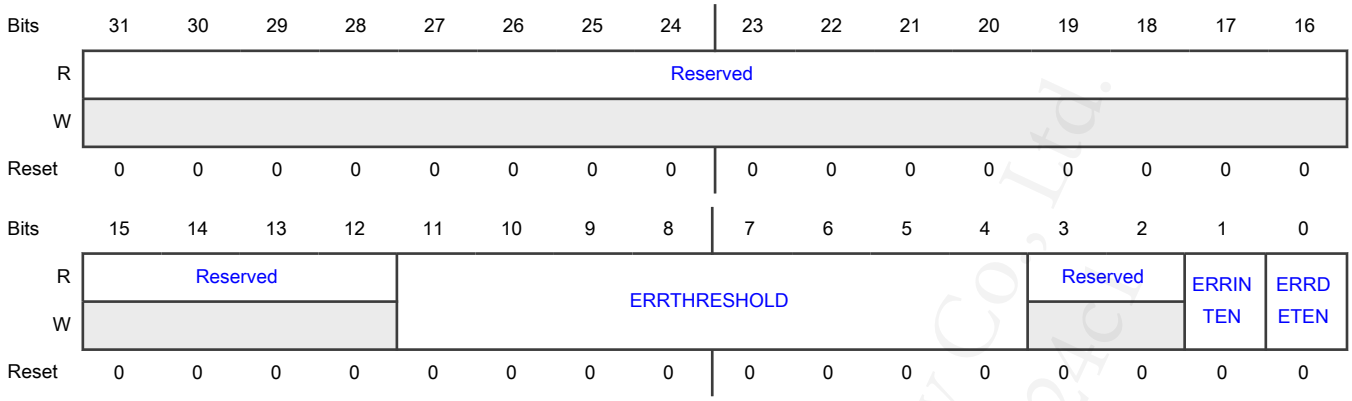
Register	Offset
CAIUCEC	100h

Function

Use this register to control the detection and signaling of correctable errors in a CAIU.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e5-4821-4848-1251-390a3ab624c1

Diagram



Fields

Field	Function
31-12 —	Reserved
11-4 ERRTHRESHOLD	Correctable Error Threshold Use this field to determine the threshold number of correctable errors that must be detected before Ncore asserts the correctable error interrupt signal.
3-2 —	Reserved
1 ERRINTEN	Correctable Error Interrupt Enable Use this field to control assertion of the correctable error interrupt signal on fulfilment of proper conditions. 0b - Disable assertion 1b - Enable assertion
0 ERRDETEN	Correctable Error Detection Enable Use this field to control the correctable error detection and logging logic. 0b - Disable correctable error detection and logging logic 1b - Enable correctable error detection and logging logic

7.10.1.5 CAIU Correctable Error Status (CAIUCES)

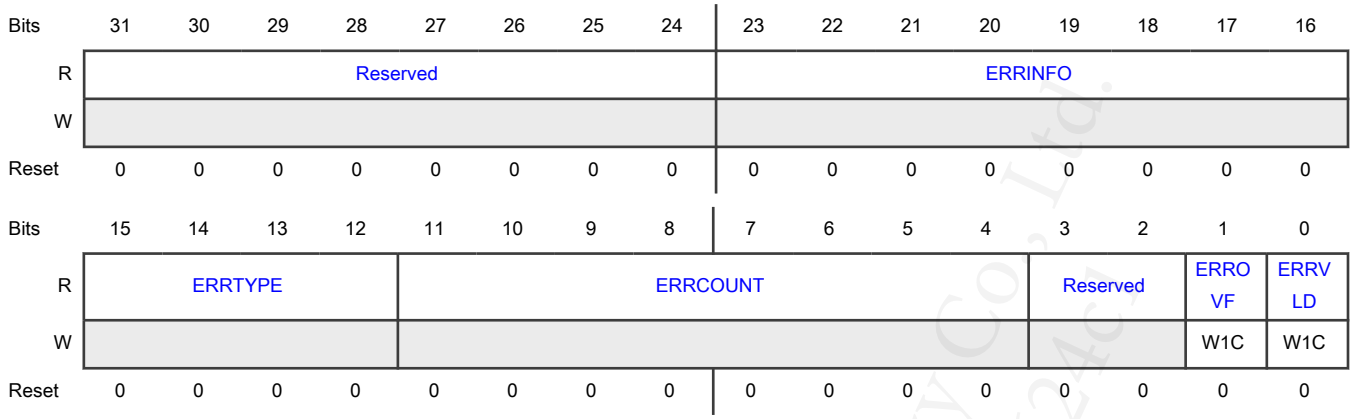
Offset

Register	Offset
CAIUCES	104h

Function

This register logs information about the correctable errors in a CAIU.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore writes appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when CAIUCES[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore writes appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when CAIUCES[ERRVLD] = 1; otherwise, the field is undefined. When Ncore writes 0 to this field, CAIUCES[ERRINFO] = 0 decodes to OTT data buffers
11-4 ERRCOUNT	Error Count This field indicates the number of correctable errors detected by the unit. The field value stops incrementing when you write 1 to CAIUCES[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of correctable errors detected by the unit overflowed the value indicated by CAIUCES[ERRCOUNT]. You write 1 to this field to clear the CAIUCES[ERRCOUNT] field.
0 ERRVLD	Error Valid Ncore writes a 1 to this field to indicate that the following registers have logged information about the correctable error; otherwise, certain fields in these registers are undefined.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• CAIU Correctable Error Status (CAIUCES) • CAIU Correctable Error Location Register 0 (CAIUCELR0) • CAIU Correctable Error Location Register 1 (CAIUCELR1)

7.10.1.6 CAIU Correctable Error Location Register 0 (CAIUCELR0)

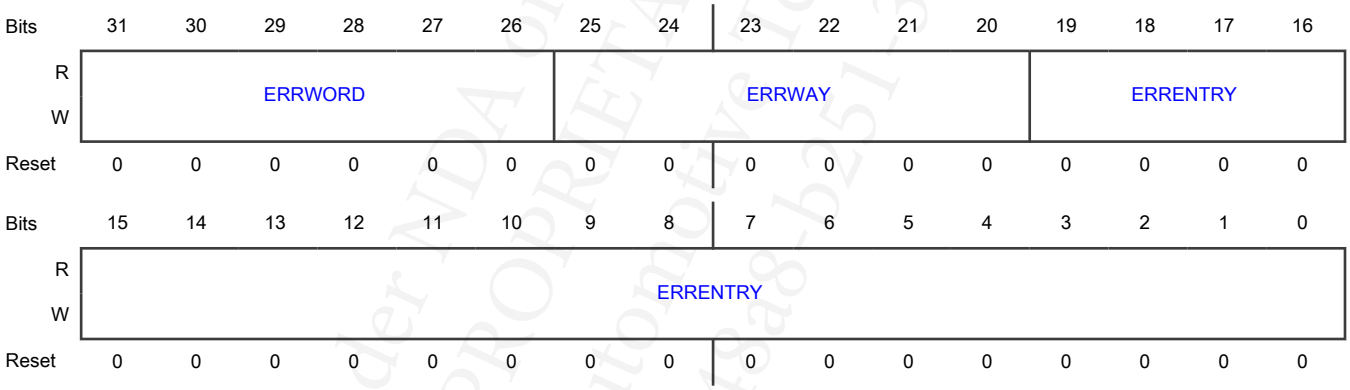
Offset

Register	Offset
CAIUCELR0	108h

Function

This register logs information about the correctable error locations in a CAIU.

Diagram



Fields

Field	Function
31-26 ERRWORD	Error Word When CAIUCES[ERRVLD] = 1, this field indicates the data word in which the logged error was detected; otherwise, the field is undefined.
25-20 ERRWAY	Error Way When CAIUCES[ERRVLD] = 1, this field indicates the way on which the logged error was detected; otherwise, the field is undefined.
19-0 ERRENTRY	Error Entry or Set When CAIUCES[ERRVLD] = 1, this field indicates the entry or the set index on which the logged error was detected; otherwise, the field is undefined.

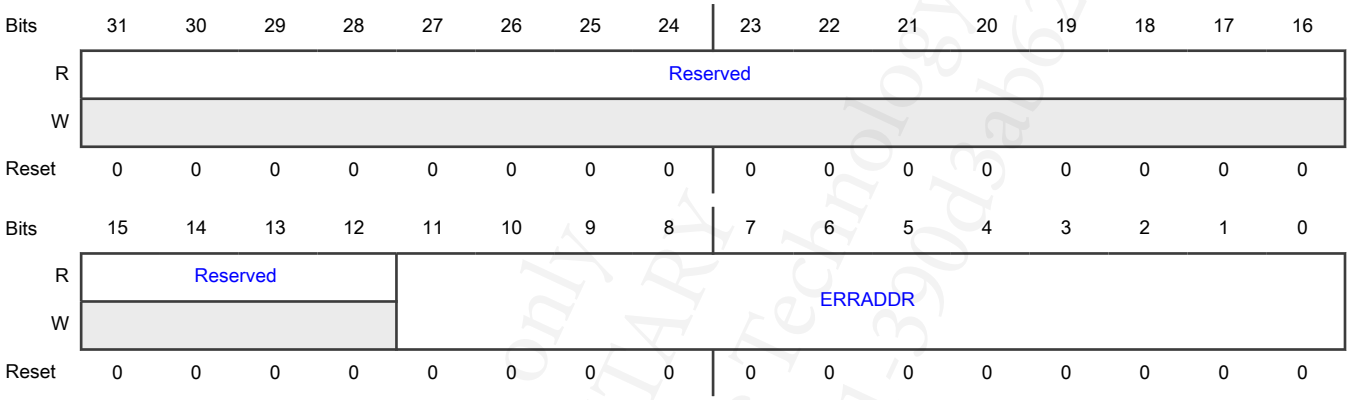
7.10.1.7 CAIU Correctable Error Location Register 1 (CAIUCELR1)

Offset

Register	Offset
CAIUCELR1	10Ch

Function
This register logs information about correctable error locations in a CAIU.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 ERRADDR	Error Address When CAIUCES[ERRVLD] = 1, this field indicates the high-order address bits of the transaction that encountered the error; otherwise, the field is undefined.

7.10.1.8 CAIU Correctable Error Status Alias (CAIUCESA)

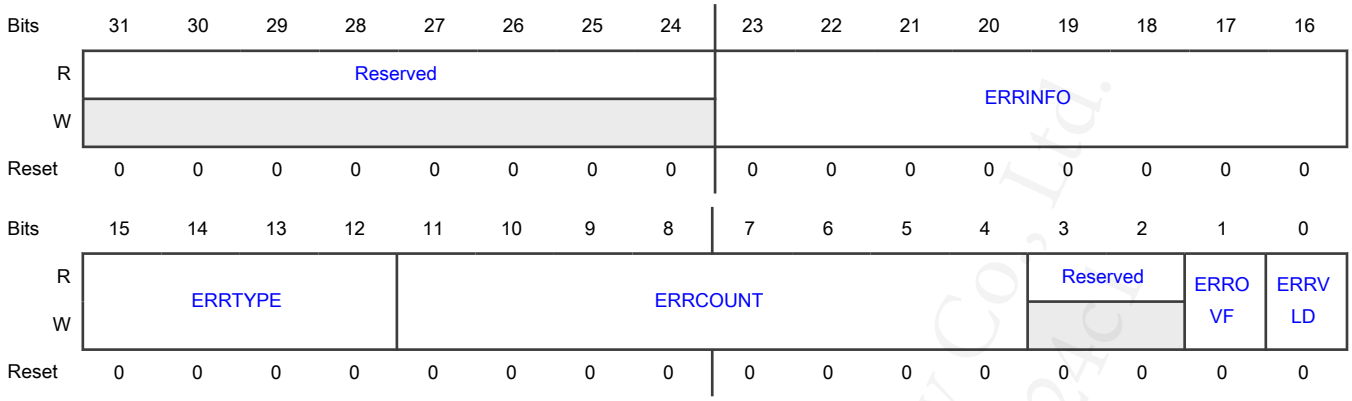
Offset

Register	Offset
CAIUCESA	124h

Function
Use this register to have a write access to the CAIU Correctable Error Status (CAIUCES) for diagnostic purposes.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co., Ltd.
ca462e5-4224-48a8-b251-390daab624c1

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore writes appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when CAIUCESA[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore writes appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when CAIUCESA[ERRVLD] = 1; otherwise, the field is undefined. When Ncore writes 0 to this field, CAIUCESA[ERRINFO] = 0 decodes to OTT data buffers.
11-4 ERRCOUNT	Error Count This field indicates the number of correctable errors detected by the unit. The field value stops incrementing when you write a 1 to CAIUCESA[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of correctable errors detected by the unit overflowed the value indicated by CAIUCESA[ERRCOUNT]. You write 1 to this field to clear the CAIUCESA[ERRCOUNT] field, and it also clears this field.
0 ERRVLD	Error Valid Ncore writes a 1 to this field to indicate that the following registers have logged information about a correctable error; otherwise, certain fields in these registers are undefined.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• CAIU Correctable Error Status (CAIUCES) • CAIU Correctable Error Location Register 0 (CAIUCELR0) • CAIU Correctable Error Location Register 1 (CAIUCELR1) You write 1 to this field to clear it.

7.10.1.9 CAIU Uncorrectable Error Control (CAIUUEC)

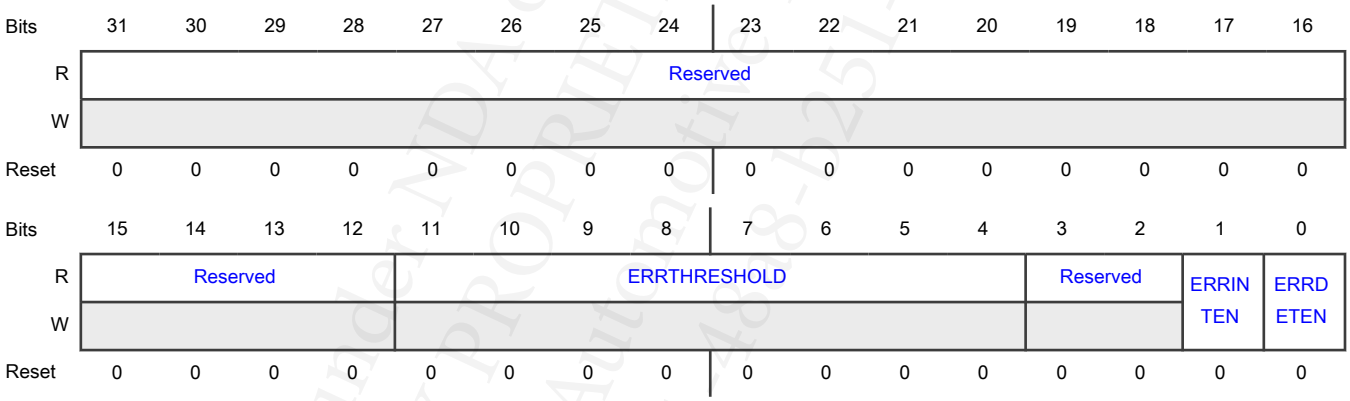
Offset

Register	Offset
CAIUUEC	140h

Function

Use this register to control the detection and signaling of uncorrectable errors in a CAIU.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-4 ERRTHRESHOLD	Uncorrectable Error Threshold Use this field to determine the threshold number of uncorrectable errors that must be detected before the Ncore asserts the uncorrectable error interrupt signal.
3-2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 ERRINTEN	Uncorrectable Error Interrupt Enable Use this field to control the assertion of uncorrectable error interrupt signal on fulfilment of proper conditions. 0b - Disable assertion 1b - Enable assertion
0 ERRDETEN	Uncorrectable Error Detection Enable Use this field to control uncorrectable error detection and logging logic. 0b - Disable uncorrectable error detection and logging logic 1b - Enable uncorrectable error detection and logging logic

7.10.1.10 CAIU Uncorrectable Error Status (CAIUUES)

Offset

Register	Offset
CAIUUES	144h

Function

This register logs information about the uncorrectable errors in a CAIU.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								ERRINFO							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRTYPE				ERRCOUNT								Reserved		ERROVF	ERRVLD
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c4625-4921-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
23-16 ERRINFO	Error Info Ncore writes appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when CAIUUES[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore writes appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when CAIUUES[ERRVLD] = 1; otherwise, the field is undefined. When Ncore writes 0 to this field, CAIUUES[ERRINFO] = 0 decodes to OTT data buffers
11-4 ERRCOUNT	Error Count This field indicates the number of uncorrectable errors detected by the unit. The field value stops incrementing when you write a 1 to CAIUUES[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of uncorrectable errors detected by the unit overflowed the value indicated by CAIUUES[ERRCOUNT]. You write a 1 to this field to clear the CAIUUES[ERRCOUNT] field, and it also clears this field.
0 ERRVLD	Error Valid Ncore writes a 1 to this field to indicate that the following registers have logged information about an uncorrectable error; otherwise, certain fields in these registers are undefined. • CAIU Uncorrectable Error Status (CAIUUES) • CAIU Uncorrectable Error Location Register 0 (CAIUUELR0) • CAIU Uncorrectable Error Location Register 1 (CAIUUELR1)

7.10.1.11 CAIU Uncorrectable Error Location Register 0 (CAIUUELR0)

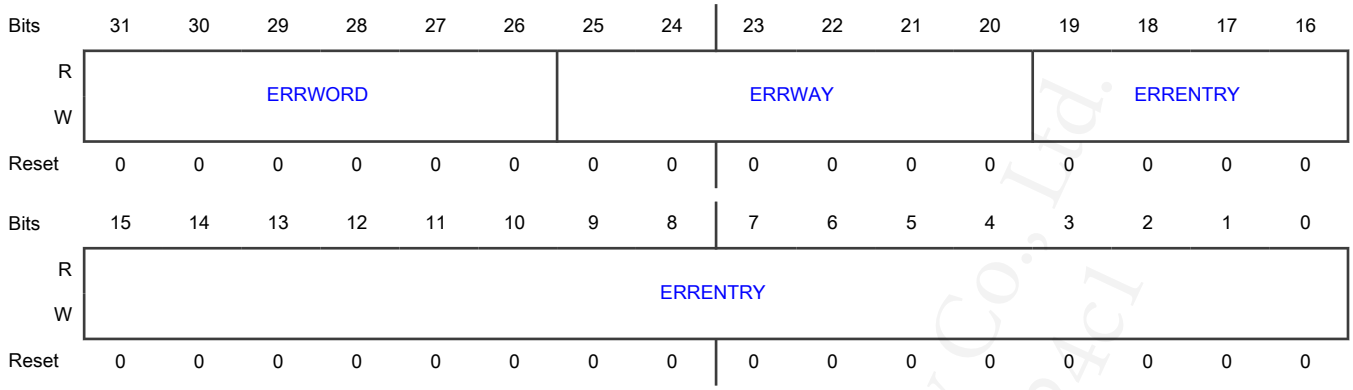
Offset

Register	Offset
CAIUUELR0	148h

Function

This register logs information about uncorrectable error locations in a CAIU.

Diagram



Fields

Field	Function
31-26 ERRWORD	Error Word When CAIUUES[ERRVLD] = 1, this field indicates the data word in which the logged error was detected; otherwise, the field is undefined.
25-20 ERRWAY	Error Way When CAIUUES[ERRVLD] = 1, this field indicates the way on which the logged error was detected; otherwise, the field is undefined.
19-0 ERRENTRY	Error Entry or Set When CAIUUES[ERRVLD] = 1, this field indicates the entry or the set index on which the logged error was detected; otherwise, the field is undefined.

7.10.1.12 CAIU Uncorrectable Error Location Register 1 (CAIUUELR1)

Offset

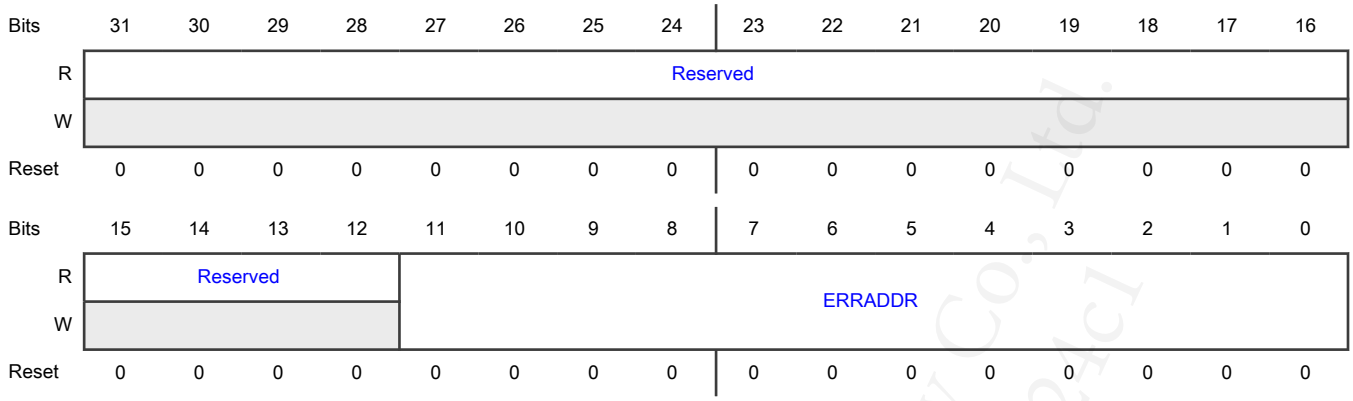
Register	Offset
CAIUUELR1	14Ch

Function

This register logs information about uncorrectable errors in a CAIU.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e3-4821-4858-1251-390d3ab624c1

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 ERRADDR	Error Address When CAIUUES[ERRVLD] = 1, this field indicates the high-order address bits of the transaction that encountered the error; otherwise, the field is undefined.

7.10.1.13 CAIU Uncorrectable Error Status Alias (CAIUUESA)

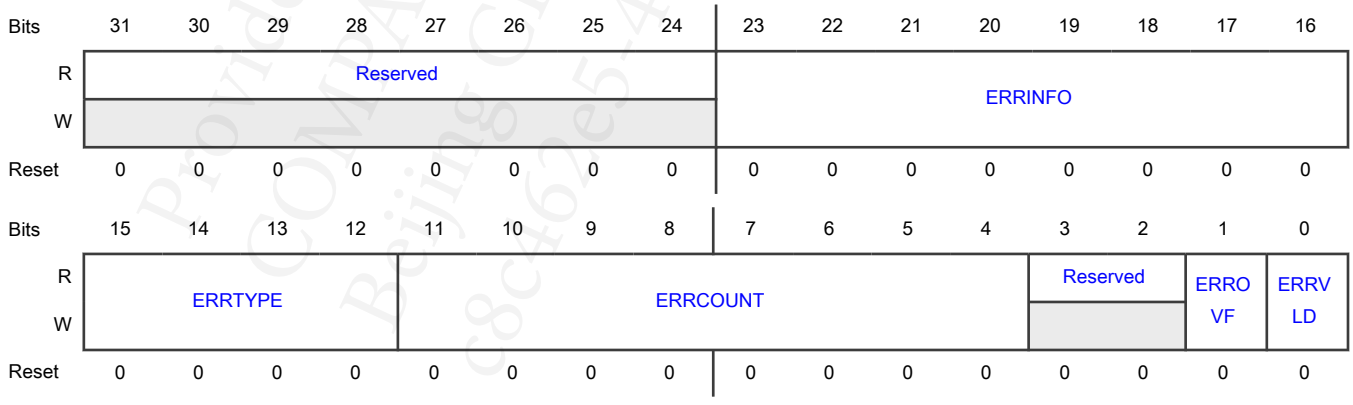
Offset

Register	Offset
CAIUUESA	164h

Function

Use this register to have write access to the CAIU Uncorrectable Error Status (CAIUUES) for diagnostic purposes.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore writes appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when CAIUUESA[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore writes appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when CAIUUESA[ERRVLD] = 1; otherwise, the field is undefined. When Ncore writes 0 to this field, CAIUUESA[ERRINFO] = 0 decodes to OTT data buffers
11-4 ERRCOUNT	Error Count This field indicates the number of uncorrectable errors detected by the unit. The field value stops incrementing when you write a 1 to CAIUUESA[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of uncorrectable errors detected by the unit overflowed the value indicated by CAIUUESA[ERRCOUNT]. You write a 1 to this field to clear the CAIUUESA[ERRCOUNT] field.
0 ERRVLD	Error Valid Ncore writes a 1 to this field to indicate that the following registers have logged information about an uncorrectable error; otherwise, certain fields in these registers are undefined. • CAIU Uncorrectable Error Status (CAIUUES) • CAIU Uncorrectable Error Location Register 0 (CAIUUELR0) • CAIU Uncorrectable Error Location Register 1 (CAIUUELR1) You write a 1 to this field to clear it.

7.10.1.14 CAIU Identification (CAIUID)

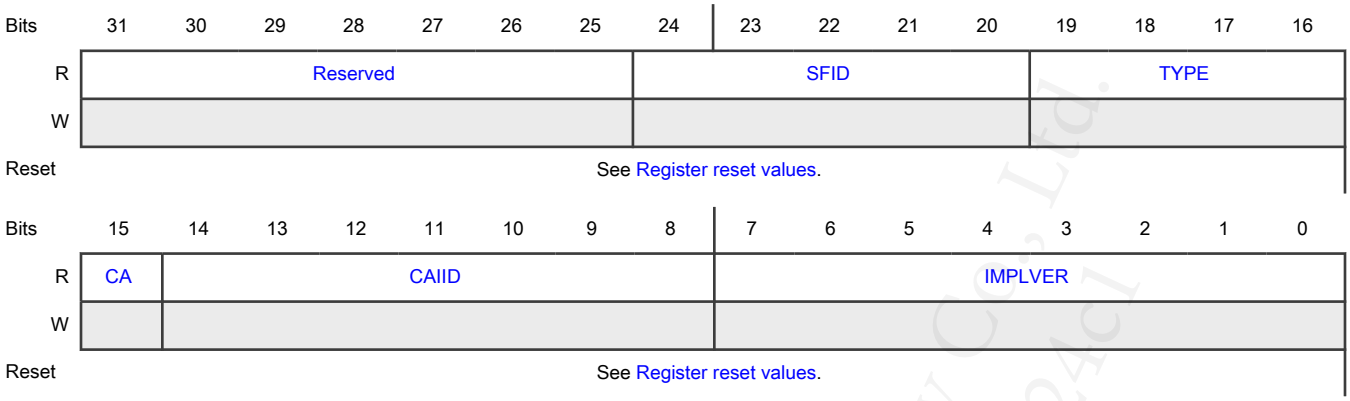
Offset

Register	Offset
CAIUID	FFCh

Function

Use this register to identify certain features and configuration information about a CAIU.

Diagram



Register reset values

Register	Reset value
CAIUID	CAIU0: 0000_8001h CAIU1: 0000_8101h

Fields

Field	Function
31-25 —	Reserved
24-20 SFID	Snoop Filter Identifier Ncore writes to this field to identify the snoop filter with which the CAI is associated.
19-16 TYPE	Type Ncore writes appropriate values to this field to identify the type of CAIU. 0000b - ACE CAI with DVM support 0001b - ACE-Lite CAI with DVM support 0010b - ACE CAI without DVM support 0011b - ACE-Lite CAI without DVM support
15 CA	Caching Agent Ncore writes to this field to identify whether the CAI is represented as a caching agent in a snoop filter. 0b - CAI is not represented as a caching agent 1b - CAI is represented as a caching agent
14-8 CAIID	CAI Identifier Ncore writes to this field to identify the CAI with which the CAIU is associated.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 IMPLVER	Implementation Version Ncore writes to this field to indicate the implementation version of the CAIU. 0b – Unimplemented CAIU 1b – Implemented CAIU

7.10.2 Non-coherent bridge unit (NCBU) register descriptions

7.10.2.1 NCBU memory map

NCBU0 base address: 5046_0000h

NCBU1 base address: 5046_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	NCBU Transaction Control (NCBUTC)	32	RW	0000_0001h
4h	NCBU Transaction Activity (NCBUTA)	32	RO	0000_0000h
100h	NCBU Correctable Error Control (NCBUCEC)	32	RW	0000_0000h
104h	NCBU Correctable Error Status (NCBUCES)	32	W1C	0000_0000h
108h	NCBU Correctable Error Location Register 0 (NCBUCELR0)	32	RW	0000_0000h
10Ch	NCBU Correctable Error Location Register 1 (NCBUCELR1)	32	RW	0000_0000h
124h	NCBU Correctable Error Status Alias (NCBUCESA)	32	RW	0000_0000h
140h	NCBU Uncorrectable Error Control (NCBUUEC)	32	RW	0000_0000h
144h	NCBU Uncorrectable Error Status (NCBUUES)	32	W1C	0000_0000h
148h	NCBU Uncorrectable Error Location Register 0 (NCBUUELR0)	32	RW	0000_0000h
14Ch	NCBU Uncorrectable Error Location Register 1 (NCBUUELR1)	32	RW	0000_0000h
164h	NCBU Uncorrectable Error Status Alias (NCBUUESA)	32	RW	0000_0000h
FFCh	NCBU Identification Register (NCBUID)	32	RO	See description

7.10.2.2 NCBU Transaction Control (NCBUTC)

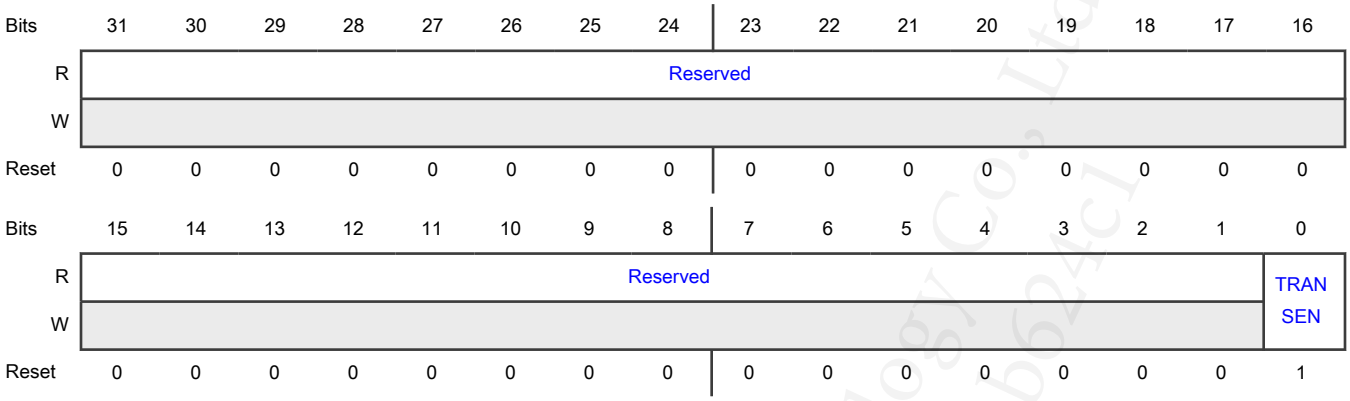
Offset

Register	Offset
NCBUTC	0h

Function

Use this register to enable agent transactions in a NCBU.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 TRANSEN	Agent Transaction Enable Write a 1 to this field to enable agent transactions.

7.10.2.3 NCBU Transaction Activity (NCBUTA)

Offset

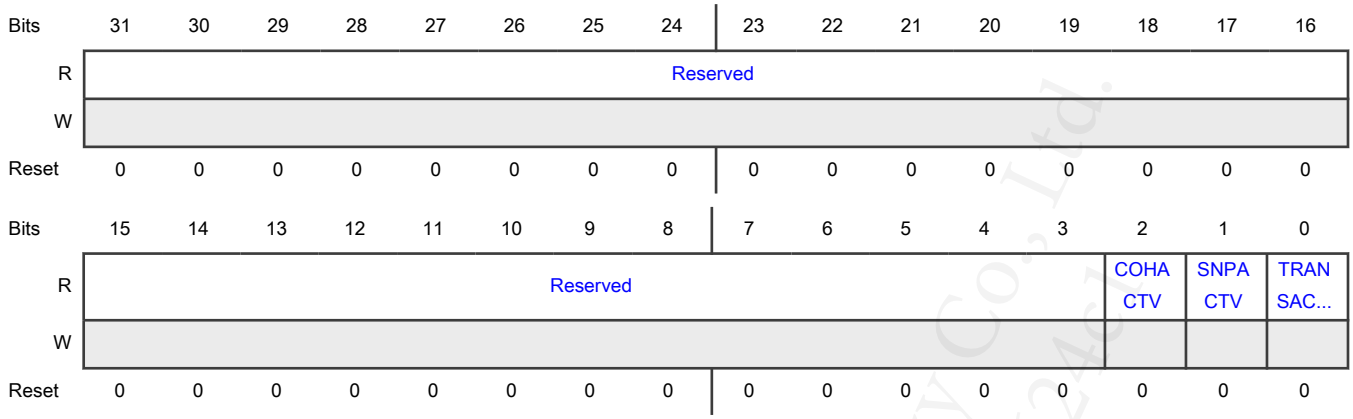
Register	Offset
NCBUTA	4h

Function

Use this register to learn whether certain protocol transactions are active in a NCBU.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4621-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-3 —	Reserved
2 COHACTV	Coherent Transaction Active Ncore writes a 1 to this field when the unit is performing any activity related to coherent transactions, and is cleared otherwise.
1 SNPACTV	Snoop Transaction Active Ncore writes a 1 to this field when the unit is performing any activity related to snoop messages, and is cleared otherwise.
0 TRANSACTV	Transaction Active Ncore writes a 1 to this field when the unit is performing any activity related to native agent coherent transactions, and is cleared otherwise.

7.10.2.4 NCBU Correctable Error Control (NCBUCEC)

Offset

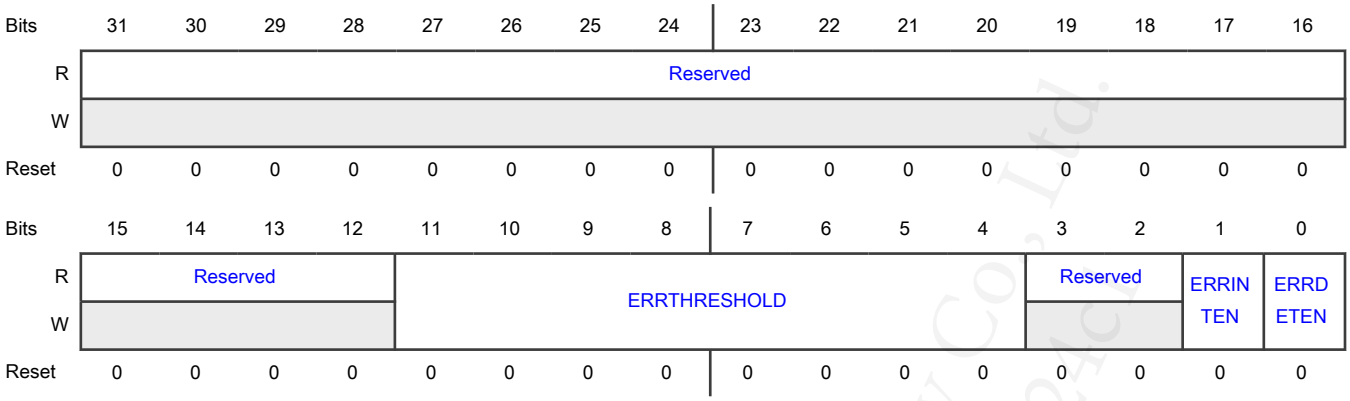
Register	Offset
NCBUCEC	100h

Function

Use this register to control the detection and signaling or correctable errors in a NCBU.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e5-4821-4848-1251-390a3ab624c1

Diagram



Fields

Field	Function
31-12 —	Reserved
11-4 ERRTHRESHOLD	Correctable Error Threshold Use this field to determine the threshold number of correctable errors that must be detected before the Ncore asserts the correctable error interrupt signal.
3-2 —	Reserved
1 ERRINTEN	Correctable Error Interrupt Enable Use this field to control assertion of correctable error interrupt signal on fulfilment of proper conditions. 0b - Disable assertion 1b - Enable assertion
0 ERRDETEN	Correctable Error Detection Enable Use this field to control correctable error detection and logging logic. 0b - Disable correctable error detection and logging logic 1b - Enable correctable error detection and logging logic

7.10.2.5 NCBU Correctable Error Status (NCBUCES)

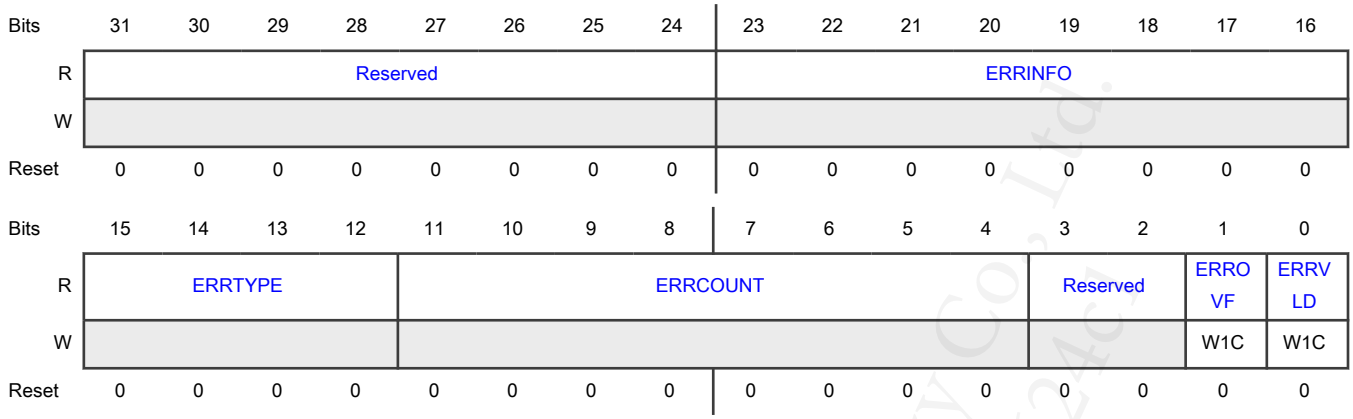
Offset

Register	Offset
NCBUCES	104h

Function

This register logs information about correctable errors in a NCBU.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore writes appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when NCBUCES[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore writes appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when NCBUCES[ERRVLD] = 1; otherwise, the field is undefined. • 0h – NCBU data correctable error. When NCBUCES[ERRTYPE] = 0, it decodes to OTT data buffers
11-4 ERRCOUNT	Error Count This field indicates the number of correctable errors detected by the unit. The field value stops incrementing when you write a 1 to NCBUCES[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of correctable errors detected by the unit overflowed the value indicated by NCBUCES[ERRCOUNT]. You write a 1 to this field to clear the NCBUCES[ERRCOUNT].
0 ERRVLD	Error Valid Ncore writes a 1 to this field to indicate that the following registers have logged information about a correctable error; otherwise, certain fields in these registers are undefined.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• NCBU Correctable Error Status (NCBUCES) • NCBU Correctable Error Location Register 0 (NCBUCELR0) • NCBU Correctable Error Location Register 1 (NCBUCELR1)

7.10.2.6 NCBU Correctable Error Location Register 0 (NCBUCELR0)

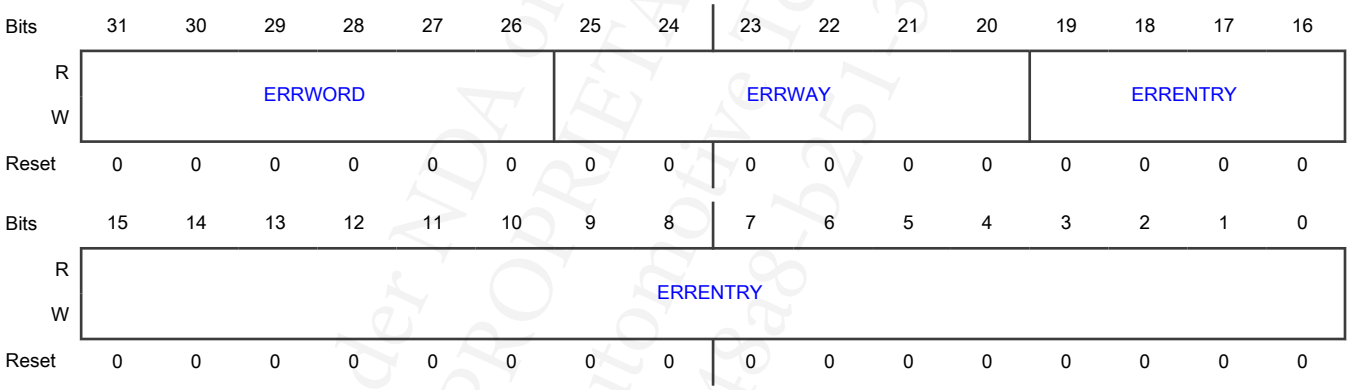
Offset

Register	Offset
NCBUCELR0	108h

Function

This register logs information about the correctable error locations in a NCBU.

Diagram



Fields

Field	Function
31-26 ERRWORD	Error Word When NCBUCES[ERRVLD] = 1, this field indicates the data word in which the logged error was detected; otherwise, the field is undefined.
25-20 ERRWAY	Error Way When NCBUCES[ERRVLD] = 1, this field indicates the way on which the logged error was detected; otherwise, the field is undefined.
19-0 ERRENTRY	Error Entry (or Set) When NCBUCES[ERRVLD] = 1, this field indicates the entry or the set index on which the logged error was detected; otherwise, the field is undefined.

7.10.2.7 NCBU Correctable Error Location Register 1 (NCBUCELR1)

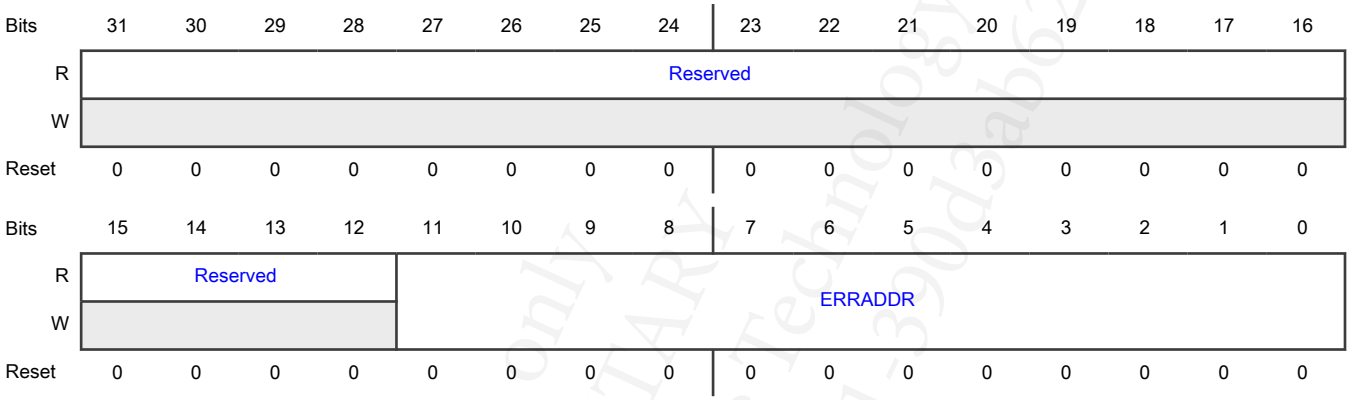
Offset

Register	Offset
NCBUCELR1	10Ch

Function

This register logs information about the correctable error locations in a NCBU.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 ERRADDR	Error Address When NCBUCES[ERRVLD] = 1, this field indicates the high-order address bits of the transaction that encountered the error; otherwise, the field is undefined.

7.10.2.8 NCBU Correctable Error Status Alias (NCBUCESA)

Offset

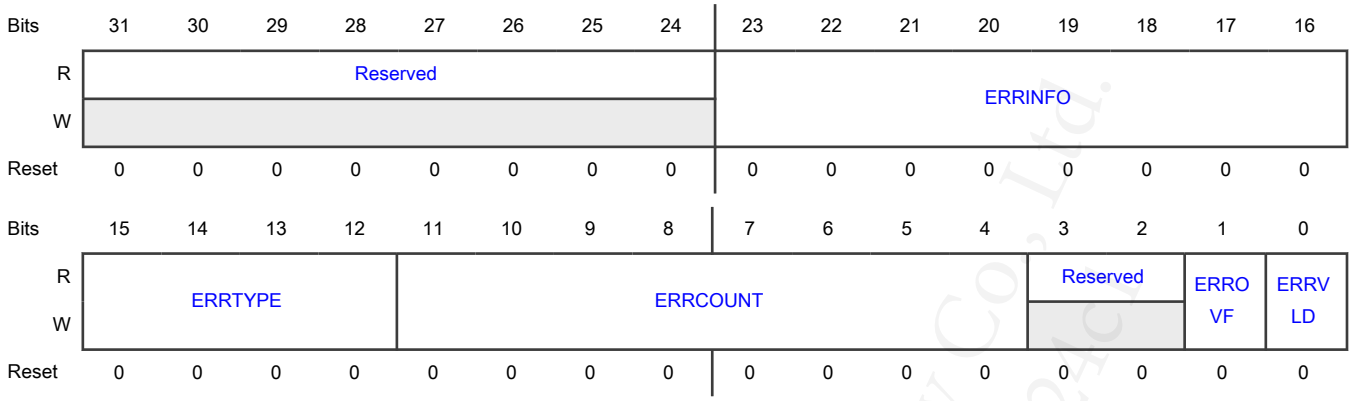
Register	Offset
NCBUCESA	124h

Function

Use this register to have write access to the NCBU Correctable Error Status (NCBUCES) for diagnostic purposes.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co., Ltd.
cc462e5-4224-48a8-b251-390daab624c1

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore writes appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when NCBUCESA[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore writes appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when NCBUCESA[ERRVLD] = 1; otherwise, the field is undefined. • 0h – NCBU data correctable error. When NCBUCESA[ERRINFO] = 0h, it decodes to OTT data buffers
11-4 ERRCOUNT	Error Count This field indicates the number of correctable errors detected by the unit. The field value stops incrementing when you write a 1 to NCBUCESA[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of correctable errors detected by the unit overflowed the value indicated by NCBUCESA[ERRCOUNT]. You write a 1 to this field to clear the NCBUCESA[ERRCOUNT].
0 ERRVLD	Error Valid Ncore writes a 1 to this field to indicate that the following registers have logged information about a correctable error; otherwise, certain fields in these registers are undefined.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• NCBU Correctable Error Status (NCBUCES) • NCBU Correctable Error Location Register 0 (NCBUCELR0) • NCBU Correctable Error Location Register 1 (NCBUCELR1)

7.10.2.9 NCBU Uncorrectable Error Control (NCBUUEC)

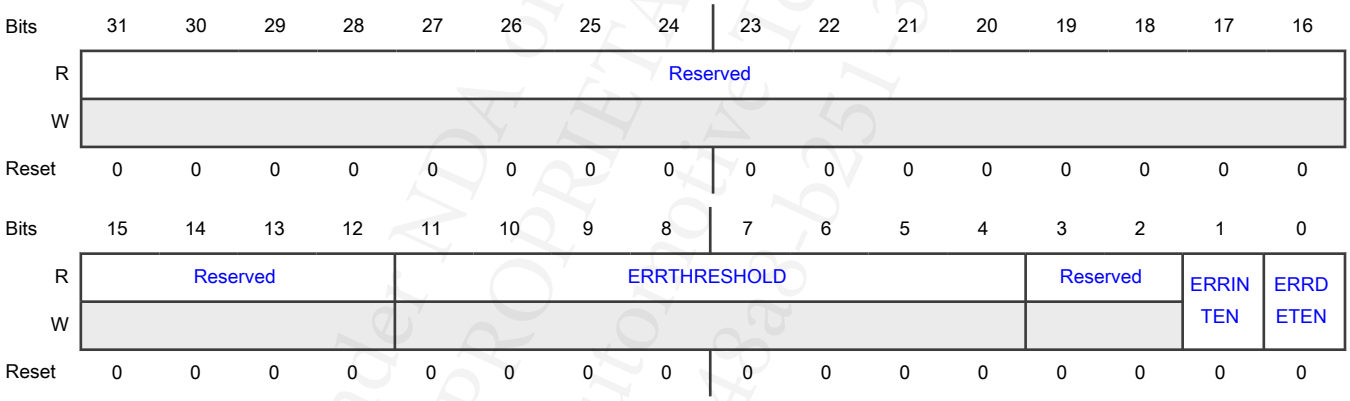
Offset

Register	Offset
NCBUUEC	140h

Function

Use this register to control the detection and signaling of uncorrectable errors in a NCBU.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-4 ERRTHRESHOLD	Uncorrectable Error Threshold Use this field to determine the threshold number of uncorrectable errors that must be detected before Ncore asserts the uncorrectable error interrupt signal.
3-2 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 ERRINTEN	Uncorrectable Error Interrupt Enable Use this field to control assertion of uncorrectable error interrupt signal on fulfilment of proper conditions. 0b - Disable assertion 1b - Enable assertion
0 ERRDETEN	Uncorrectable Error Detection Enable Use this field to control uncorrectable error detection and logging logic. 0b - Disable uncorrectable error detection and logging logic 1b - Enable uncorrectable error detection and logging logic

7.10.2.10 NCBU Uncorrectable Error Status (NCBUUES)

Offset

Register	Offset
NCBUUES	144h

Function
This register logs information about the uncorrectable errors in a NCBU.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								ERRINFO							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRTYPE				ERRCOUNT								Reserved		ERROVF	ERRVLD
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4e01-48a3-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
23-16 ERRINFO	Error Info Ncore writes appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when NCBUUES[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore writes appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when NCBUUES[ERRVLD] = 1; otherwise, the field is undefined. • 0h – NCBU data uncorrectable error, when NCBUUES[ERRINFO] = 0, it decodes to OTT data
11-4 ERRCOUNT	Error Count This field indicates the number of uncorrectable errors detected by the unit. The field value stops incrementing when you write a 1 to NCBUUES[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of uncorrectable errors detected by the unit overflowed the value indicated by NCBUUES[ERRCOUNT]. You write a 1 to this field to clear the NCBUUES[ERRCOUNT].
0 ERRVLD	Error Valid Ncore writes a 1 to this field to indicate that the following registers have logged information about an uncorrectable error; otherwise, certain fields in these registers are undefined. • NCBU Uncorrectable Error Status (NCBUUES) • NCBU Uncorrectable Error Location Register 0 (NCBUUELR0) • NCBU Uncorrectable Error Location Register 1 (NCBUUELR1)

7.10.2.11 NCBU Uncorrectable Error Location Register 0 (NCBUUELR0)

Offset

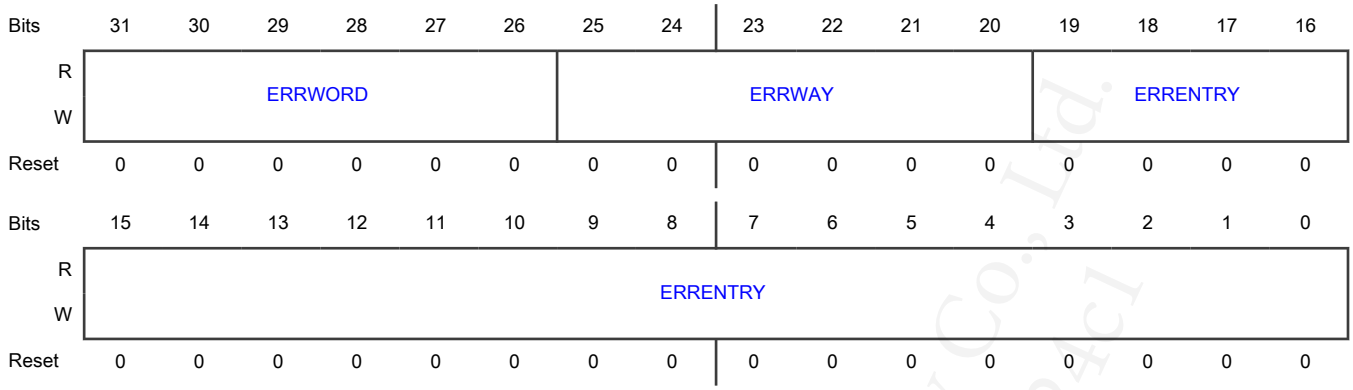
Register	Offset
NCBUUELR0	148h

Function

This register logs information about uncorrectable error locations in a NCBU.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHYA Automotive Technology Co., Ltd.
c8c462e5-4821-4ab2-b1 390a3ab0a1c1

Diagram



Fields

Field	Function
31-26 ERRWORD	Error Word When NCBUUES[ERRVLD] = 1, this field indicates the data word on which the logged error was detected; otherwise, the field is undefined.
25-20 ERRWAY	Error Way When NCBUUES[ERRVLD] = 1, this field indicates the way on which the logged error was detected; otherwise, the field is undefined.
19-0 ERRENTRY	Error Entry (or Set) When NCBUUES[ERRVLD] = 1, this field indicates the entry or the set index on which the logged error was detected; otherwise, the field is undefined.

7.10.2.12 NCBU Uncorrectable Error Location Register 1 (NCBUUELR1)

Offset

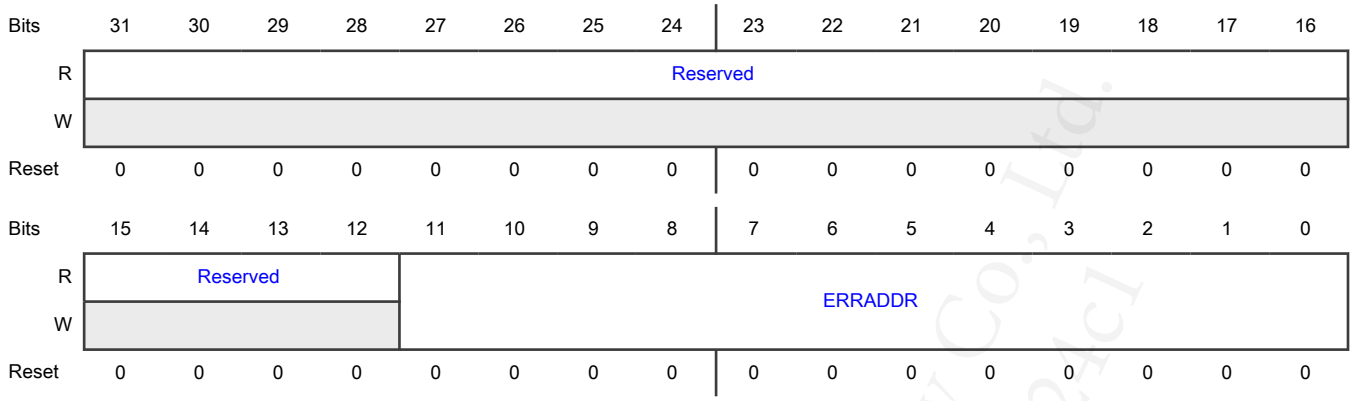
Register	Offset
NCBUUELR1	14Ch

Function

This register logs information about uncorrectable error locations in a NCBU.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4858-1251-390d3ab624c1

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 ERRADDR	Error Address When NCBUUES[ERRVLD] = 1, this field indicates the high-order address bits of the transaction that encountered the error; otherwise, the field is undefined.

7.10.2.13 NCBU Uncorrectable Error Status Alias (NCBUUESA)

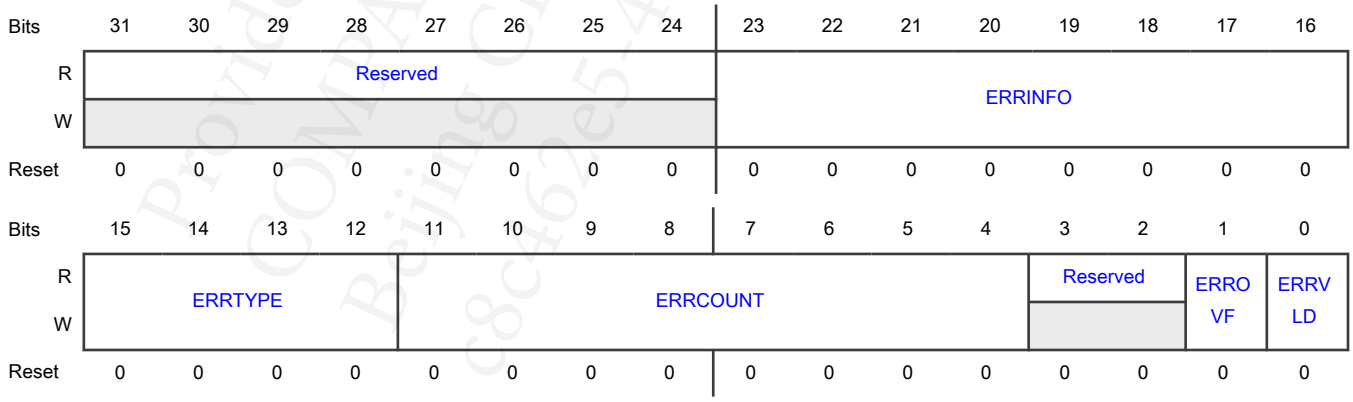
Offset

Register	Offset
NCBUUESA	164h

Function

Use this register to have write access to the NCBU Uncorrectable Error Status (NCBUUES) for diagnostic purposes.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c467e5-5482-48a8-b251-390d3ab624c1

Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore writes appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when NCBUUESA[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore writes appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when NCBUUESA[ERRVLD] = 1; otherwise, the field is undefined. • 0h – NCBU data uncorrectable error, when NCBUUESA[ERRINFO] = 0, it decodes to OTT data
11-4 ERRCOUNT	Error Count This field indicates the number of uncorrectable errors detected by the unit. The field stops incrementing when you write a 1 to NCBUUESA[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of uncorrectable errors detected by the unit overflowed the value indicated by NCBUUESA[ERRCOUNT]. You write a 1 to this field to clear the NCBUUESA[ERRCOUNT].
0 ERRVLD	Error Valid If this field is set, the NCBU Uncorrectable Error Status (NCBUUES), NCBU Uncorrectable Error Location Register 0 (NCBUUELR0) and NCBU Uncorrectable Error Location Register 1 (NCBUUELR1) have logged information about an uncorrectable error; otherwise, certain fields in these registers are undefined. Writing a 1 to this field clears it.

7.10.2.14 NCBU Identification Register (NCBUID)

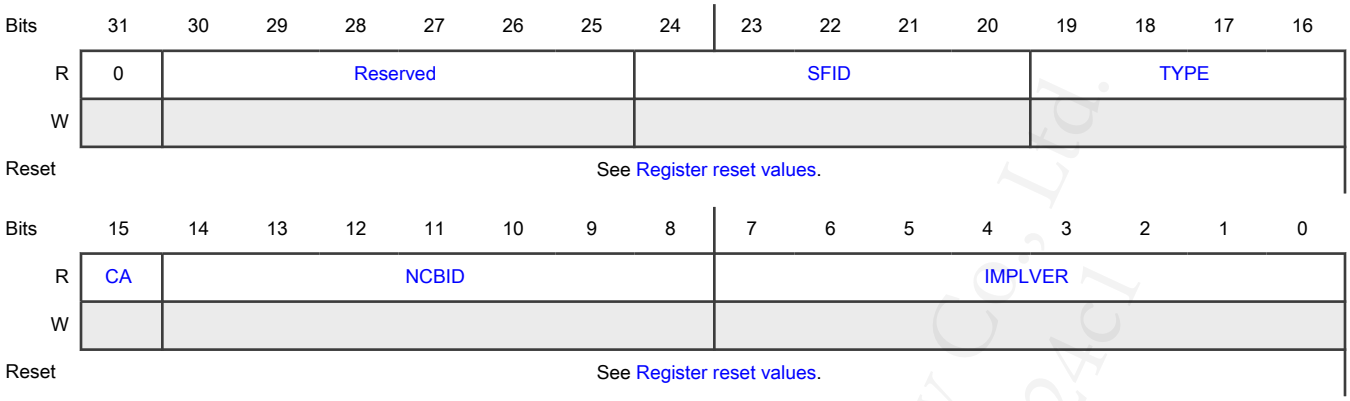
Offset

Register	Offset
NCBUID	FFCh

Function

Use this register to identify certain features and configuration information about a NCBU.

Diagram



Register reset values

Register	Reset value
NCBUID	NCBU0: 0003_0001h NCBU1: 0003_0101h

Fields

Field	Function
31 —	Reserved
30-25 —	Reserved
24-20 SFID	Snoop Filter Identifier Ncore writes to this field to identify the snoop filter with which the NCB is associated.
19-16 TYPE	Type Ncore writes appropriate values to this field to identify the type of NCBU. Ncore always writes 11b to this field.
15 CA	Caching Agent Ncore writes to this field to identify whether the NCB is represented as a caching agent in a snoop filter.
14-8 NCBID	Non-Coherent Bridge Identifier Ncore writes to this field to identify the NCB with which the NCBU is associated.
7-0 IMPLVER	Implementation Version Ncore writes to this field to indicate the implementation version of the NCBU. 0b – Unimplemented NCBU 1b – Implemented NCBU

Provided under NDA only
COMPANY PROPRIETARY
Renesas Electronics Corporation
Renesas Automotive Technology Co., Ltd.
8c4c02e3-4821-48a8-b251-390d3ab624c1

7.10.3 Directory unit (DIRU) register descriptions

7.10.3.1 DIRU memory map

DIRU base address: 5048_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	DIRU Transaction Activity (DIRUTA)	32	RO	0000_0000h
10h	DIRU Snoop Filtering Enable (DIRUSFE)	32	RW	0000_0000h
40h	DIRU Caching Agent Snoop Enable (DIRUCASE0)	32	RW	0000_0000h
50h	DIRU Caching Agent Snoop Activity (DIRUCASA0)	32	RO	0000_0000h
80h	DIRU Snoop Filter Maintenance Control (DIRUSFMC)	32	RW	0000_0000h
84h	DIRU Snoop Filter Maintenance Activity (DIRUSFMA)	32	RO	0000_0000h
88h	DIRU Snoop Filter Maintenance Location Register 0 (DIRUSFMLR0)	32	RW	0000_0000h
8Ch	DIRU Snoop Filter Maintenance Location Register 1 (DIRUSFMLR1)	32	RW	0000_0000h
100h	DIRU Correctable Error Control (DIRUCEC)	32	RW	0000_0000h
104h	DIRU Correctable Error Status (DIRUCES)	32	W1C	0000_0000h
108h	DIRU Correctable Error Location Register 0 (DIRUCELR0)	32	RW	0000_0000h
10Ch	DIRU Correctable Error Location Register 1 (DIRUCELR1)	32	RW	0000_0000h
124h	DIRU Correctable Error Status Alias (DIRUCESA)	32	RW	0000_0000h
140h	DIRU Uncorrectable Error Control (DIRUUEC)	32	RW	0000_0000h
144h	DIRU Uncorrectable Error Status (DIRUUES)	32	W1C	0000_0000h
148h	DIRU Uncorrectable Error Location Register 0 (DIRUUELR0)	32	RW	0000_0000h
14Ch	DIRU Uncorrectable Error Location Register 1 (DIRUUELR1)	32	RW	0000_0000h
164h	DIRU Uncorrectable Error Status Alias (DIRUUESA)	32	RW	0000_0000h
FFCh	DIRU Identification Register (DIRUID)	32	RO	0000_0001h

7.10.3.2 DIRU Transaction Activity (DIRUTA)

Offset

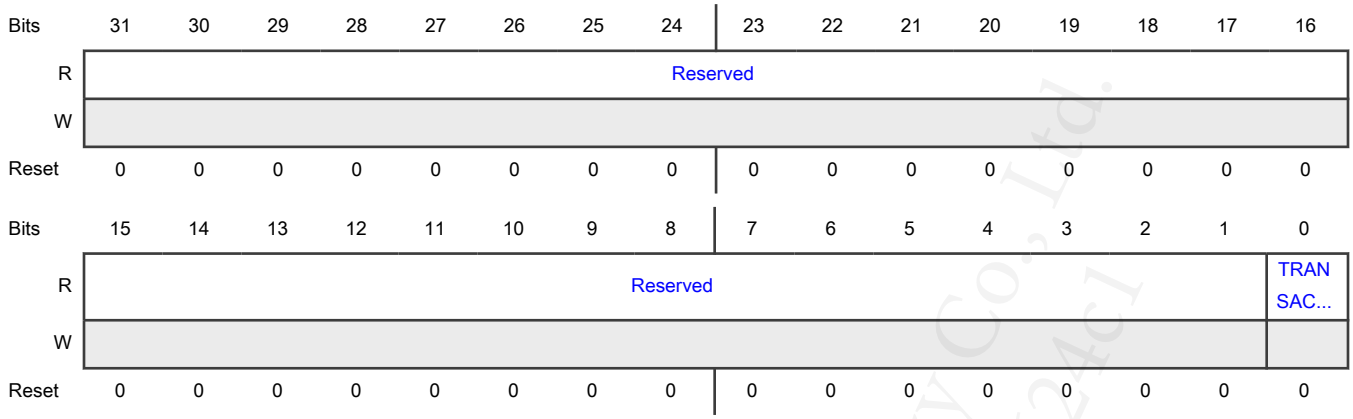
Register	Offset
DIRUTA	4h

Function

Use this register to learn whether certain protocol transactions are active in a DIRU.

Provided by IHS Markit
COMPANY CONFIDENTIAL
Beijing CEC Huada Electronic Design Co.,Ltd.
ca462e5-4821-4c8-b251-29000a62d

Diagram



Fields

Field	Function
31-1 —	Reserved
0 TRANSACTV	Transaction Active Ncore writes a 1 to this field when the unit is performing any activity related to coherent transactions, and is clear otherwise.

7.10.3.3 DIRU Snoop Filtering Enable (DIRUSFE)

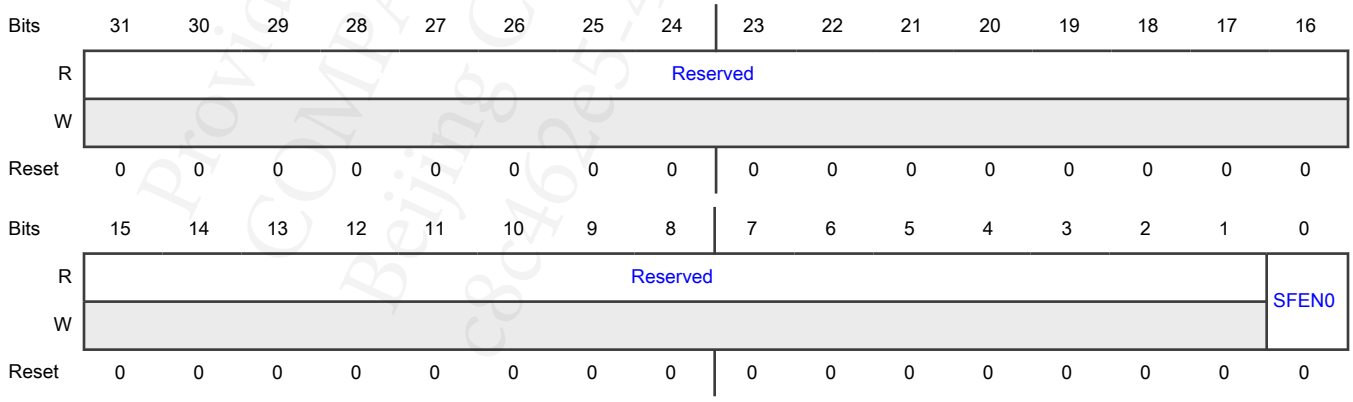
Offset

Register	Offset
DIRUSFE	10h

Function

Use this register to control snoop filtering for a given snoop filter in a DIRU.

Diagram



Fields

Field	Function
31-1 —	Reserved
0-0 SFENn	Snoop Filter Enable Write a 1 to this field to enable snoop filtering for the corresponding snoop filter.

7.10.3.4 DIRU Caching Agent Snoop Enable (DIRUCASE0)

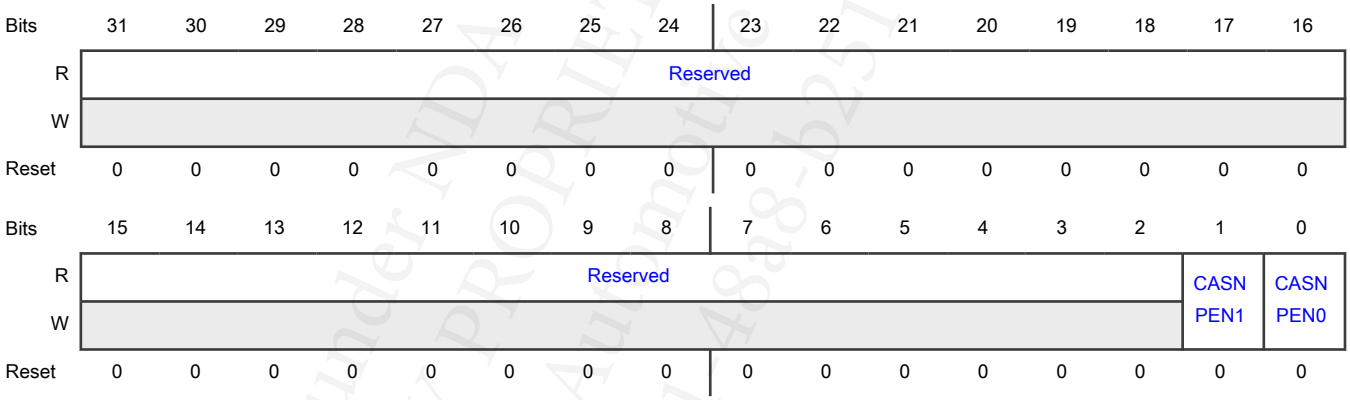
Offset

Register	Offset
DIRUCASE0	40h

Function

Use this register to control snoop messages to the specified CAIU associated with a caching agent.

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 CASNPENn	Caching Agent Snoop Enable Write a 1 to this field to enable snoop messages to the CAIU.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c62e54821484b8-b251-390d3ab624c1

7.10.3.5 DIRU Caching Agent Snoop Activity (DIRUCASA0)

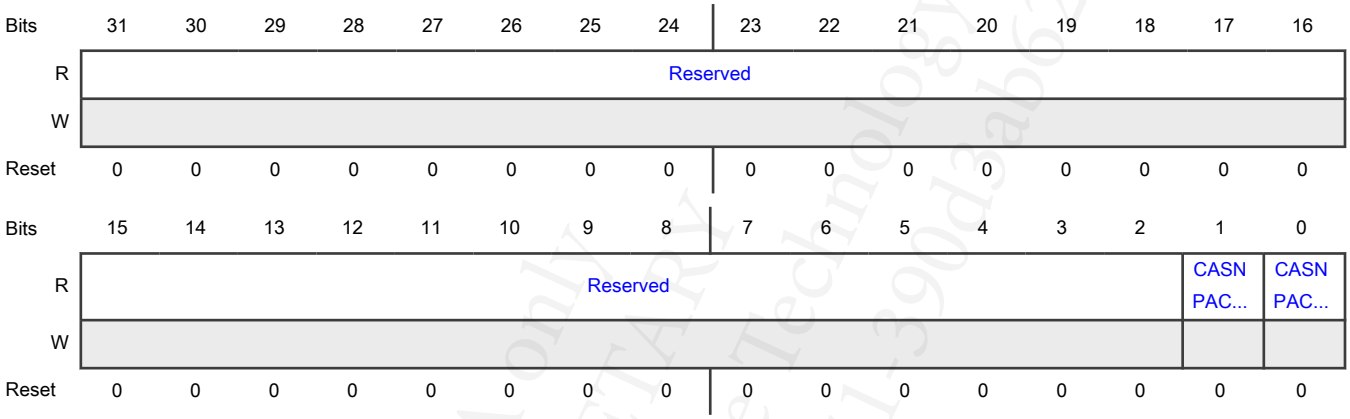
Offset

Register	Offset
DIRUCASA0	50h

Function

Use this register to learn whether the snoop messages are active to the specified CAIU associated with a caching agent.

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 CASNPACTVn	Caching Agent Snoop Activity Ncore writes a 1 to this field to indicate that snoop messages are active to the CAIU; otherwise, snoop messages are not active.

7.10.3.6 DIRU Snoop Filter Maintenance Control (DIRUSFMC)

Offset

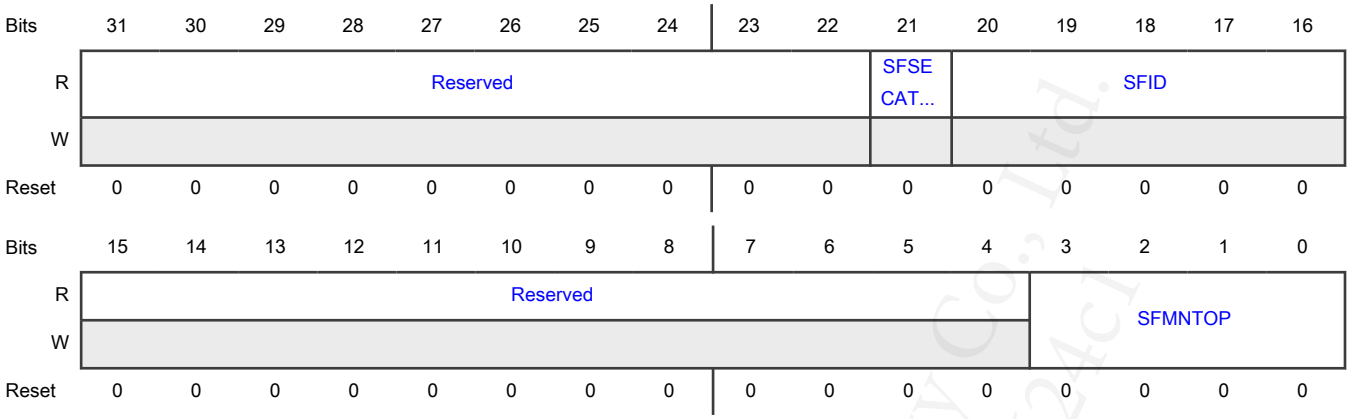
Register	Offset
DIRUSFMC	80h

Function

Use this register to control maintenance operations on a snoop filter for a given DIRU.

When you write to this register, the unit performs the snoop filter maintenance operation specified by the value written to the Snoop Filter Maintenance Operation field on the snoop filter specified by the value written to the Snoop Filter Identifier field.

Diagram



Fields

Field	Function
31-22 —	Reserved
21 SFSECATTR	Snoop Filter Security Attribute Ncore writes to this field to identify the security attribute of the address on which the maintenance operation is performed. 0b - System is not configured with security attribute. 1b - System is configured with security attribute.
20-16 SFID	Snoop Filter Identifier Ncore writes to this field to identify the snoop filter on which the maintenance operation is performed.
15-4 —	Reserved
3-0 SFMNTOP	Snoop Filter Maintenance Operation Write appropriate values to this field to specify the type of maintenance operation to be performed on a snoop filter. 0000b - Initialize all entries 0001—0011b - Reserved 0100b - Flush all entries 0101b - Flush entry at set and way 0110b - Flush entry at address 0111b - Reserved 1000b - Flush all victim buffer entries 1001b—1111b - Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing ChiHy Automotive Technology Co., Ltd.
8540e25-4821-48a8-b251-390a3ab624c1

7.10.3.7 DIRU Snoop Filter Maintenance Activity (DIRUSFMA)

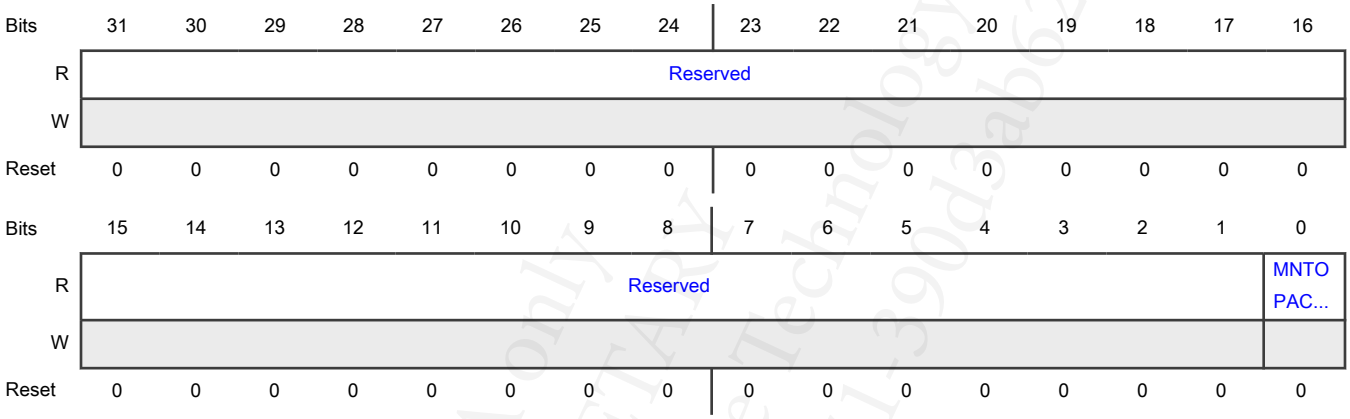
Offset

Register	Offset
DIRUSFMA	84h

Function

Use this register to learn about maintenance operation activities on a snoop filter for a given DIRU.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 MNTOPACTV	Maintenance Operation Active Ncore writes a 1 to this field when the unit is performing any maintenance operation activity on the snoop filter, and is clear otherwise.

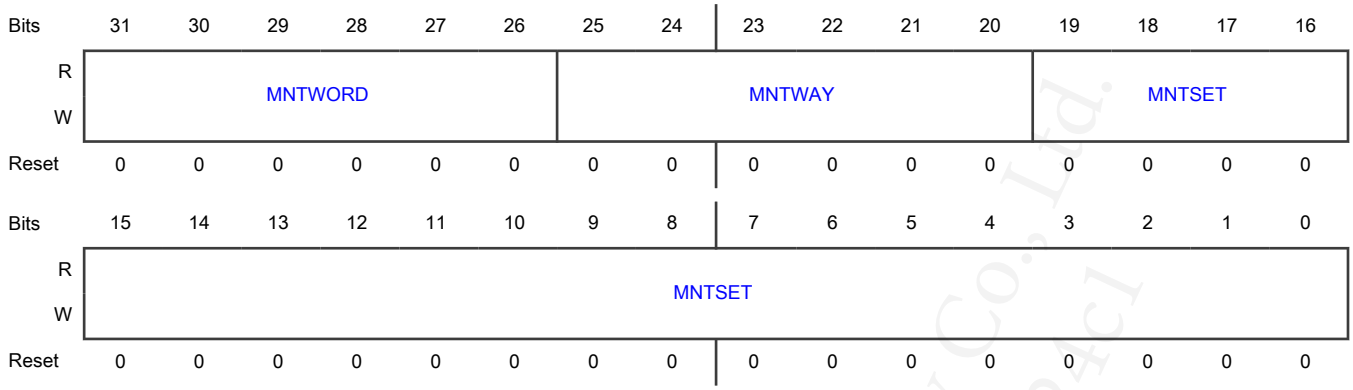
7.10.3.8 DIRU Snoop Filter Maintenance Location Register 0 (DIRUSFMLR0)

Offset

Register	Offset
DIRUSFMLR0	88h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390da3ab624c1

Diagram



Fields

Field	Function
31-26 MNTWORD	Maintenance Word Function of this field depends on the type of maintenance operation being performed. See Snoop filter maintenance for details.
25-20 MNTWAY	Maintenance Way Function of this field depends on the type of maintenance operation being performed. See Snoop filter maintenance for details.
19-0 MNTSET	Maintenance Set Function of this field depends on the type of maintenance operation being performed. See Snoop filter maintenance for details.

7.10.3.9 DIRU Snoop Filter Maintenance Location Register 1 (DIRUSFMLR1)

Offset

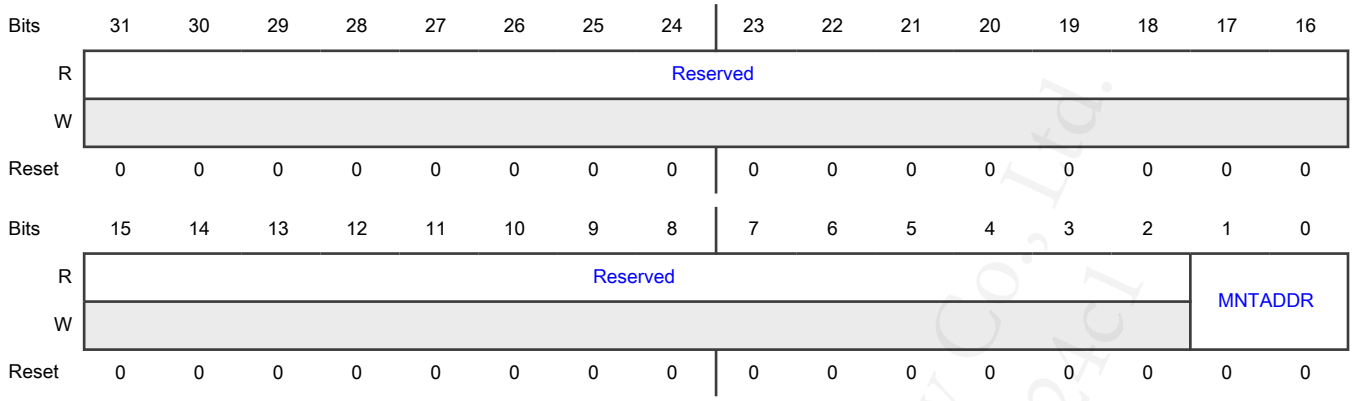
Register	Offset
DIRUSFMLR1	8Ch

Function

This register indicates the location in a snoop filter or the address on which the snoop filter maintenance operation is performed in a DIRU.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4888-1251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 MNTADDR	Maintenance Address Function of this field depends on the type of maintenance operation being performed. See Snoop filter maintenance for details.

7.10.3.10 DIRU Correctable Error Control (DIRUCEC)

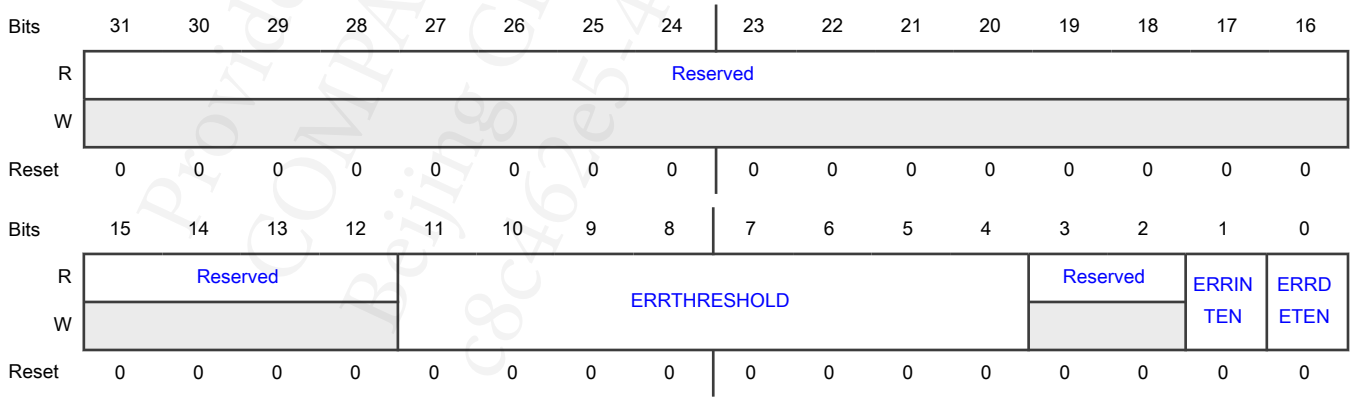
Offset

Register	Offset
DIRUCEC	100h

Function

Use this field to control correctable error detection and logging logic.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c467e5-5482-48a8-b251-390d3ab624c1

Fields

Field	Function
31-12 —	Reserved
11-4 ERRTHRESHOLD	Correctable Error Threshold Use this field to determine the threshold number of correctable errors that must be detected before Ncore asserts the correctable error interrupt signal.
3-2 —	Reserved
1 ERRINTEN	Correctable Error Interrupt Enable Use this field to control assertion of correctable error interrupt signal on fulfilment of proper conditions. 0b - Disable assertion 1b - Enable assertion
0 ERRDETEN	Correctable Error Detection Enable Use this field to control correctable error detection and logging logic. 0b - Disable correctable error detection and logging logic 1b - Enable correctable error detection and logging logic

7.10.3.11 DIRU Correctable Error Status (DIRUCES)

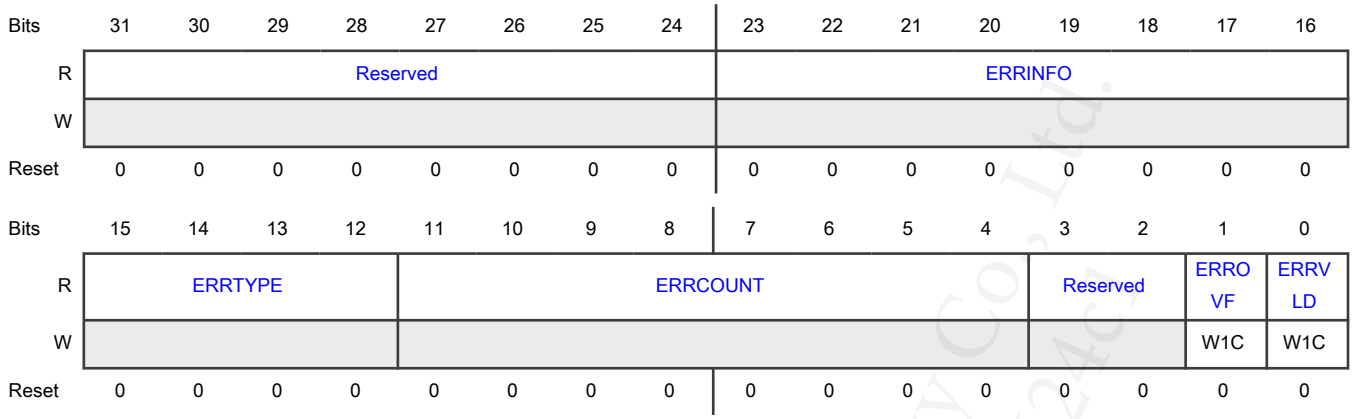
Offset

Register	Offset
DIRUCES	104h

Function
This register logs information about correctable errors in a DIRU.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-90d3ab624c1

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore write appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when DIRUCES[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore write appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when DIRUCES[ERRVLD] = 1; otherwise, the field is undefined. • 4h – Snoop filter entry correctable error. When: — DIRUCES[ERRINFO] = 0, it decodes to reserved — DIRUCES[ERRINFO] = any other value, it decodes to snoop filter identifier
11-4 ERRCOUNT	Error Count This field indicates the number of correctable errors detected by the unit. The field value stops incrementing when you write 1 to DIRUCES[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of correctable errors detected by the unit overflowed the value indicated by DIRUCES[ERRCOUNT]. You write a 1 to this field to clear the DIRUCES[ERRCOUNT] field.
0	Error Valid

Table continues on the next page...

Table continued from the previous page...

Field	Function
ERRVLD	Ncore writes a 1 to this field to indicate that the following registers have logged information about a correctable error; otherwise, certain fields in these registers are undefined. • DIRU Correctable Error Status (DIRUCES) • DIRU Correctable Error Location Register 0 (DIRUCELR0) • DIRU Correctable Error Location Register 1 (DIRUCELR1)

7.10.3.12 DIRU Correctable Error Location Register 0 (DIRUCELR0)

Offset

Register	Offset
DIRUCELR0	108h

Function

This register logs information about the correctable error locations in a DIRU.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ERRWORD						ERRWAY						ERRENTRY			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRENTRY															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 ERRWORD	Error Word When DIRUCES[ERRVLD] = 1, this field indicates the data word on which the logged error was detected; otherwise, the field is undefined.
25-20 ERRWAY	Error Way When DIRUCES[ERRVLD] = 1, this field indicates the way on which the logged error was detected; otherwise, the field is undefined.

Table continues on the next page...

Table continued from the previous page...

Field	Function
19-0 ERRENTRY	Error Entry (or Set) When DIRUCES[ERRVLD] = 1, this field indicates the entry or the set index on which the logged error was detected; otherwise, the field is undefined.

7.10.3.13 DIRU Correctable Error Location Register 1 (DIRUCELR1)

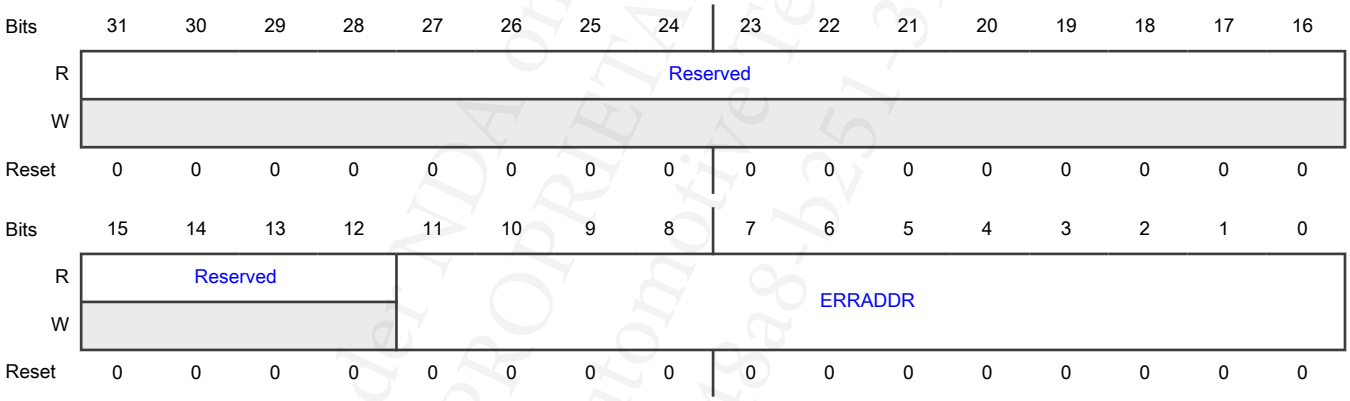
Offset

Register	Offset
DIRUCELR1	10Ch

Function

This register logs information about the correctable error locations in a DIRU.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 ERRADDR	Error Address When DIRUCES[ERRVLD] = 1, this field indicates the high-order address bits of the transaction that encountered the error; otherwise, the field is undefined.

Printed under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c40e5-4821-43a8-b251-390d3ab624c1

7.10.3.14 DIRU Correctable Error Status Alias (DIRUCESA)

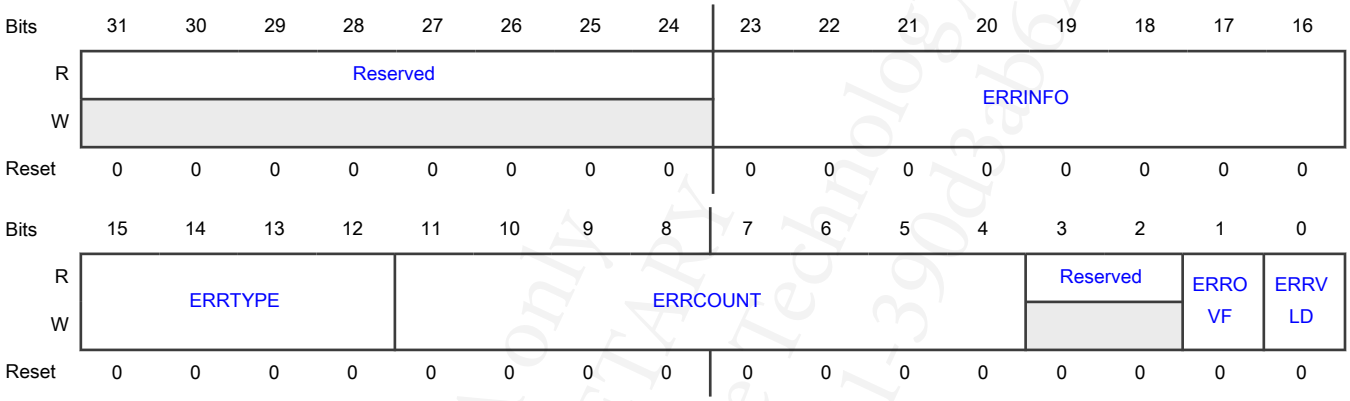
Offset

Register	Offset
DIRUCESA	124h

Function

Use this register to have write access to the DIRU Correctable Error Status (DIRUCES) for diagnostic purposes.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore write appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when DIRUCESA[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore write appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when DIRUCESA[ERRVLD] = 1; otherwise, the field is undefined. • 4h – Snoop filter entry correctable error. When: — DIRUCESA[ERRINFO] = 0, it decodes to reserved — DIRUCESA[ERRINFO] = any other value, it decodes to snoop filter identifier
11-4 ERRCOUNT	Error Count This field indicates the number of correctable errors detected by the unit. The field value stops incrementing when you write a 1 to DIRUCESA[ERROVF].

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of correctable errors detected by the unit overflowed the value indicated by DIRUCESA[ERRCOUNT]. You write a 1 to this field to clear the DIRUCESA[ERRCOUNT] field.
0 ERRVLD	Error Valid Ncore writes a 1 to this field to indicate that the following registers have logged information about a correctable error; otherwise, certain fields in these registers are undefined. • DIRU Correctable Error Status (DIRUCES) • DIRU Correctable Error Location Register 0 (DIRUCELR0) • DIRU Correctable Error Location Register 1 (DIRUCELR1) You write a 1 to this field to clear it.

7.10.3.15 DIRU Uncorrectable Error Control (DIRUUEC)

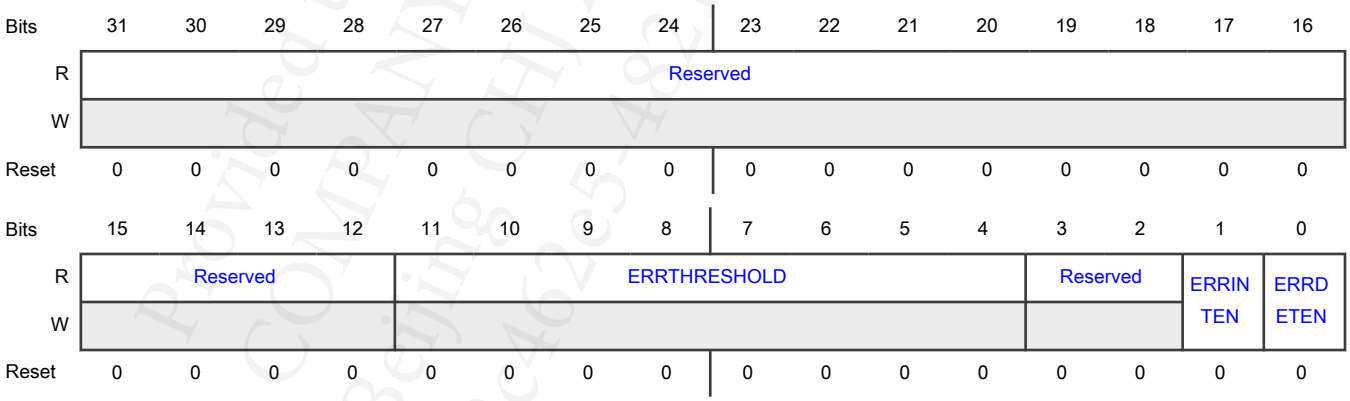
Offset

Register	Offset
DIRUUEC	140h

Function

Use this register to control the detection and signaling of uncorrectable errors in a DIRU.

Diagram



Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-48c1-48a8-b251-390d3ab624c1

Fields

Field	Function
31-12 —	Reserved
11-4 ERRTHRESHOLD	Uncorrectable Error Threshold Use this field to determine the threshold number of uncorrectable errors that must be detected before Ncore asserts the uncorrectable error interrupt signal.
3-2 —	Reserved
1 ERRINTEN	Uncorrectable Error Interrupt Enable Use this field to control assertion of uncorrectable error interrupt signal on fulfilment of proper conditions. 0b - Disable assertion 1b - Enable assertion
0 ERRDETEN	Uncorrectable Error Detection Enable Use this field to control uncorrectable error detection and logging logic. 0b - Disable uncorrectable error detection and logging logic 1b - Enable uncorrectable error detection and logging logic

7.10.3.16 DIRU Uncorrectable Error Status (DIRUUES)

Offset

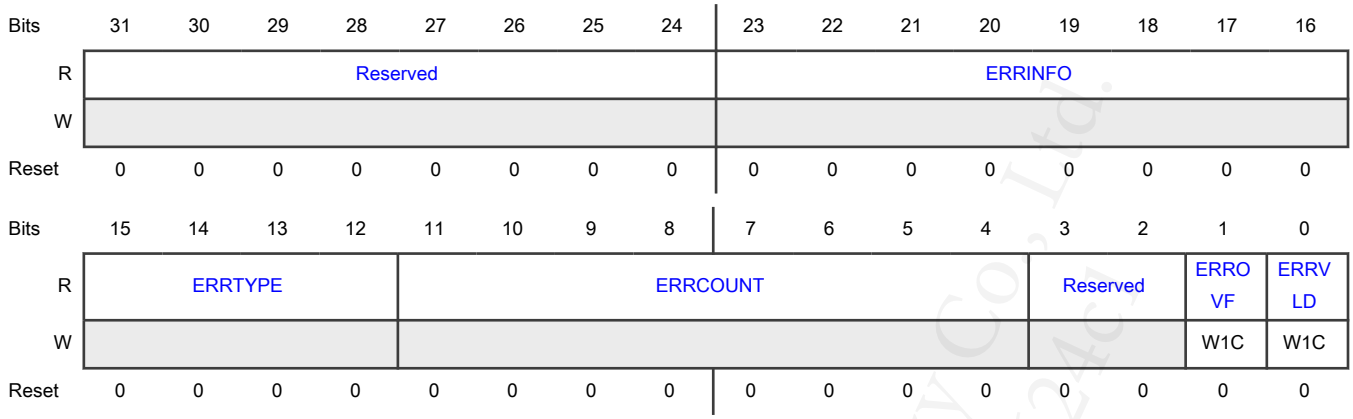
Register	Offset
DIRUUES	144h

Function

This register logs information about the uncorrectable errors in a DIRU.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4891-48a8-b251-90d3ab624c1

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore write appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when DIRUUES[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore write appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when DIRUUES[ERRVLD] = 1; otherwise, the field is undefined. • 4h – Snoop filter entry uncorrectable error. When: — DIRUUES[ERRINFO] = 0, it decodes to reserved — DIRUUES[ERRINFO] = any other value, it decodes to snoop filter identifier • 5h – Recall transaction error.

DIRUUES[ERRINFO] bits	Value	Description
23:22	0b	Reserved
21	0b	Secure address
	1b	Non-secure address
20:18	0b	Reserved
17:16	0b	Reserved
	1b	Transport error
	10b	Address corruption error
	11b	Data corruption error

Table continued from the previous page...

Field	Function
11-4 ERRCOUNT	Error Count This field indicates the number of uncorrectable errors detected by the unit. The field value stops incrementing when you write a 1 to DIRUUES[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of uncorrectable errors detected by the unit overflowed the value indicated by DIRUUES[ERRCOUNT]. You write a 1 to this field to clear the DIRUUES[ERRCOUNT] field.
0 ERRVLD	Error Valid Ncore writes a 1 to this field to indicate that the following registers have logged information about an uncorrectable error; otherwise, certain fields in these registers are undefined. • DIRU Uncorrectable Error Status (DIRUUES) • DIRU Uncorrectable Error Location Register 0 (DIRUUELR0) • DIRU Uncorrectable Error Location Register 1 (DIRUUELR1)

7.10.3.17 DIRU Uncorrectable Error Location Register 0 (DIRUUELR0)

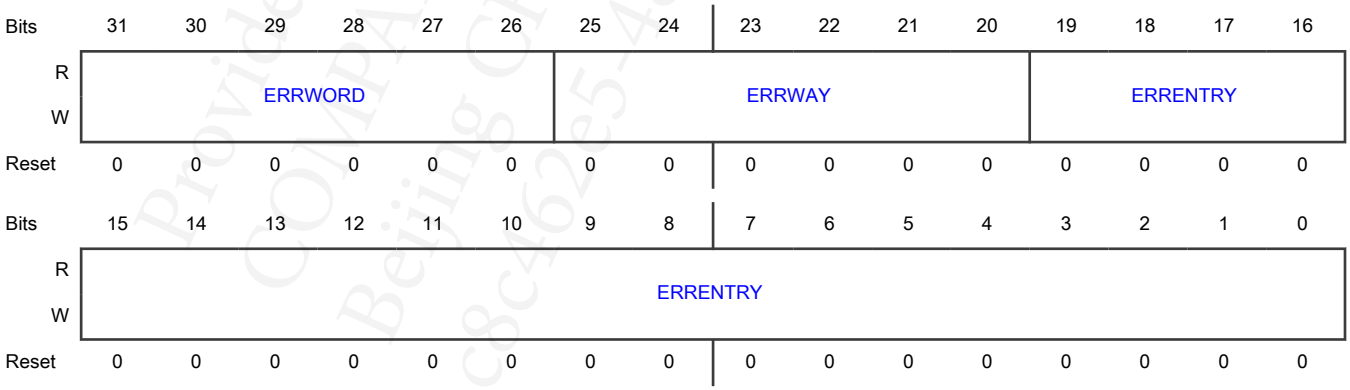
Offset

Register	Offset
DIRUUELR0	148h

Function

This register logs information about uncorrectable errors in a DIRU.

Diagram



Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c4625-482-48a8-b251-90f3ab624c1

Fields

Field	Function
31-26 ERRWORD	Error Word When DIRUUES[ERRVLD] = 1, this field indicates the data word on which the logged error was detected; otherwise, the field is undefined.
25-20 ERRWAY	Error Way When DIRUUES[ERRVLD] = 1, this field indicates the way on which the logged error was detected; otherwise, the field is undefined.
19-0 ERRENTRY	Error Entry (or Set) When DIRUUES[ERRVLD] = 1, this field indicates the entry or the set index on which the logged error was detected; otherwise, the field is undefined.

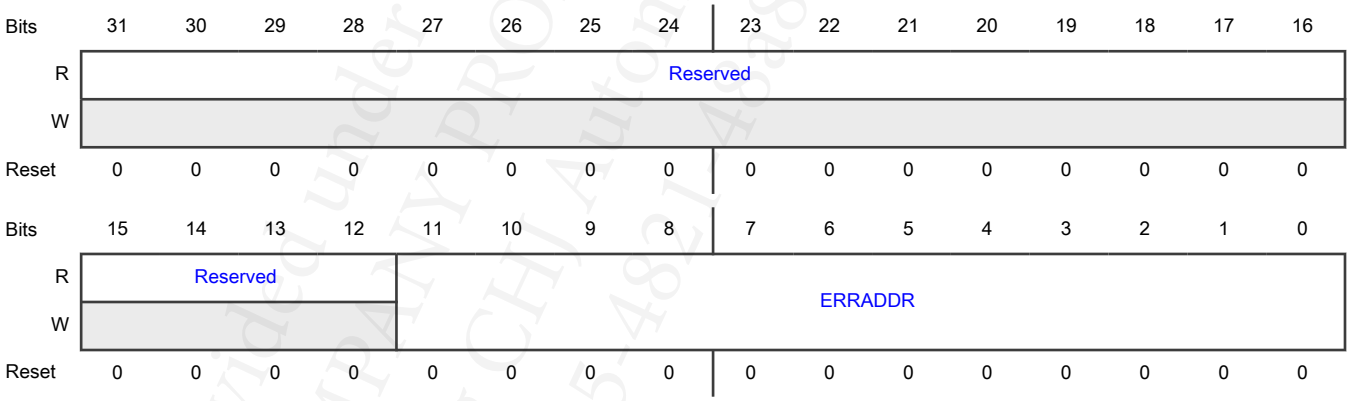
7.10.3.18 DIRU Uncorrectable Error Location Register 1 (DIRUUELR1)

Offset

Register	Offset
DIRUUELR1	14Ch

Function
This register logs information about the uncorrectable errors in a DIRU.

Diagram



Fields

Field	Function
31-12 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624d1

Table continued from the previous page...

Field	Function
11-0 ERRADDR	Error Address When DIRUUES[ERRVLD] = 1, this field indicates the high-order address bits of the transaction that encountered the error; otherwise, the field is undefined.

7.10.3.19 DIRU Uncorrectable Error Status Alias (DIRUUESA)

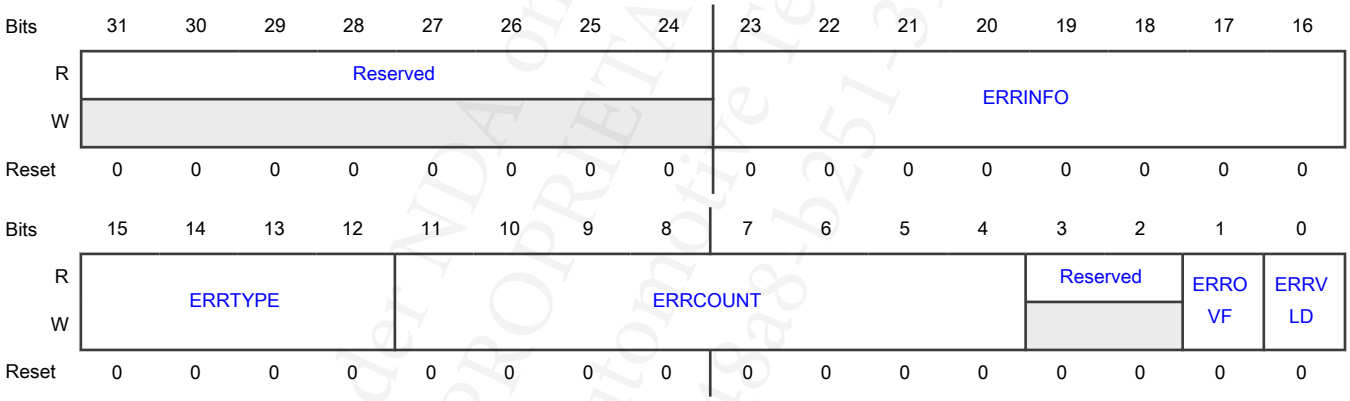
Offset

Register	Offset
DIRUUESA	164h

Function

Use this register to have write access to the DIRU Uncorrectable Error Status (DIRUUES) for diagnostic purposes.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore write appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when DIRUUESA[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore write appropriate values to this field to indicate the type of logged error.

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>Ncore writes to this field only when DIRUUESA[ERRVLD] = 1; otherwise, the field is undefined.
4h – Snoop filter entry uncorrectable error. When:
— DIRUUESA[ERRINFO] = 0, it decodes to reserved
— DIRUUESA[ERRINFO] = any other value, it decodes to snoop filter identifier
5h – Recall transaction error.
<table>
<tr><th>DIRUUESA[ERRINFO] bits</th><th>Value</th><th>Description</th></tr>
<tr><td>23:22</td><td>0b</td><td>Reserved</td></tr>
<tr><td>21</td><td>0b</td><td>Secure address</td></tr>
<tr><td></td><td>1b</td><td>Non-secure address</td></tr>
<tr><td>20:18</td><td>0b</td><td>Reserved</td></tr>
<tr><td>17:16</td><td>0b</td><td>Reserved</td></tr>
<tr><td></td><td>1b</td><td>Transport error</td></tr>
<tr><td></td><td>10b</td><td>Address corruption error</td></tr>
<tr><td></td><td>11b</td><td>Data corruption error</td></tr>
</table></td></tr>
<tr><td>11-4
ERRCOUNT</td><td>Error Count
This field indicates the number of uncorrectable errors detected by the unit. The field stops incrementing when you write 1 to DIRUUESA[ERROvf].</td></tr>
<tr><td>3-2
—</td><td>Reserved</td></tr>
<tr><td>1
ERROVF</td><td>Error Overflow
Ncore writes a 1 to this field to indicate that the number of uncorrectable errors detected by the unit overflowed the value indicated by DIRUUESA[ERRCOUNT].
You write a 1 to this field to clear the DIRUUESA[ERRCOUNT] field, and it also clears this field.</td></tr>
<tr><td>0
ERRVLD</td><td>Error Valid
Ncore writes a 1 to this field to indicate that the following registers have logged information about an uncorrectable error; otherwise, certain fields in these registers are undefined.
DIRU Uncorrectable Error Status (DIRUUES)
DIRU Uncorrectable Error Location Register 0 (DIRUUELR0)
DIRU Uncorrectable Error Location Register 1 (DIRUUELR1)
You write 1 to this field to clear it.</td></tr>
</table>

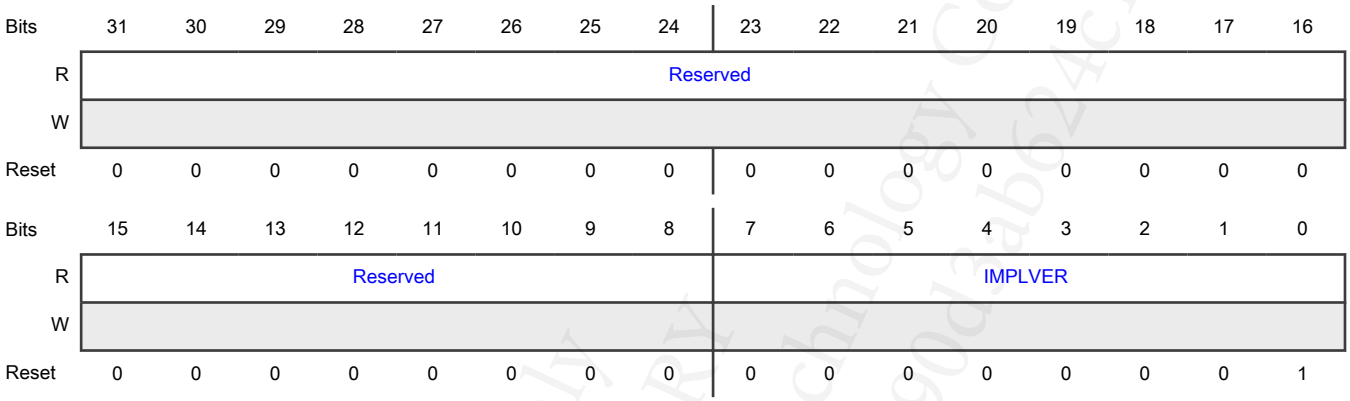
Preliminary NDA only
COMPANY PROPRIETARY
Bosch CC-Automotive Technology Co., Ltd.
c84c025-4851-48a8-b251-390d3ab624c1

7.10.3.20 DIRU Identification Register (DIRUID)

Offset

Register	Offset
DIRUID	FFCh

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 IMPLVER	Implementation Version Ncore writes to this field to indicate the implementation version of the DIRU. 0b – Unimplemented DIRU 1b – Implemented DIRU

7.10.4 Coherent memory interface unit (CMIU) register descriptions

7.10.4.1 CMIU memory map

CMIU base address: 504C_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	CMIU Transaction Activity (CMIUTA)	32	RO	0000_0000h
100h	CMIU Correctable Error Control (CMIUCEC)	32	RW	0000_0000h
104h	CMIU Correctable Error Status (CMIUCES)	32	W1C	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
108h	CMIU Correctable Error Location Register 0 (CMIUCELR0)	32	RW	0000_0000h
10Ch	CMIU Correctable Error Location Register 1 (CMIUCELR1)	32	RW	0000_0000h
124h	CMIU Correctable Error Status Alias (CMIUCESA)	32	RW	0000_0000h
140h	CMIU Uncorrectable Error Control (CMIUUEC)	32	RW	0000_0000h
144h	CMIU Uncorrectable Error Status (CMIUUES)	32	W1C	0000_0000h
148h	CMIU Uncorrectable Error Location Register 0 (CMIUUELR0)	32	RW	0000_0000h
14Ch	CMIU Uncorrectable Error Location Register 1 (CMIUUELR1)	32	RW	0000_0000h
164h	CMIU Uncorrectable Error Status Alias (CMIUUESA)	32	RW	0000_0000h
FFCh	CMIU Identification Register (CMIUID)	32	RO	0000_0001h

7.10.4.2 CMIU Transaction Activity (CMIUTA)

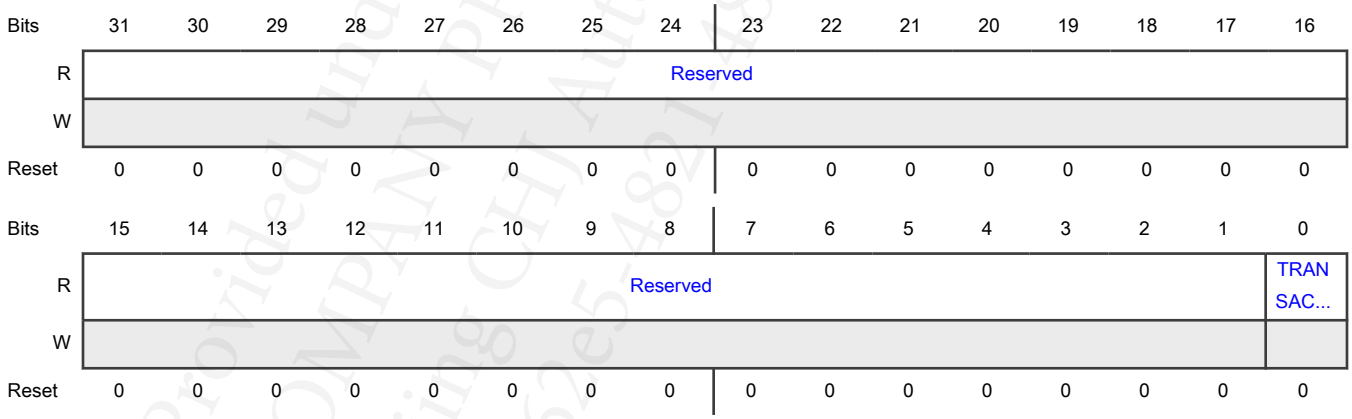
Offset

Register	Offset
CMIUTA	4h

Function

Use this register to learn whether, certain protocol transactions are active in a CMIU.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c4675e5482148a8b251-390d4b6a1.1

Fields

Field	Function
31-1 —	Reserved
0 TRANSACTV	Transaction Active Ncore writes a 1 to this field when the unit is performing any activity related to the coherent transactions, and is clear otherwise.

7.10.4.3 CMIU Correctable Error Control (CMIUCEC)

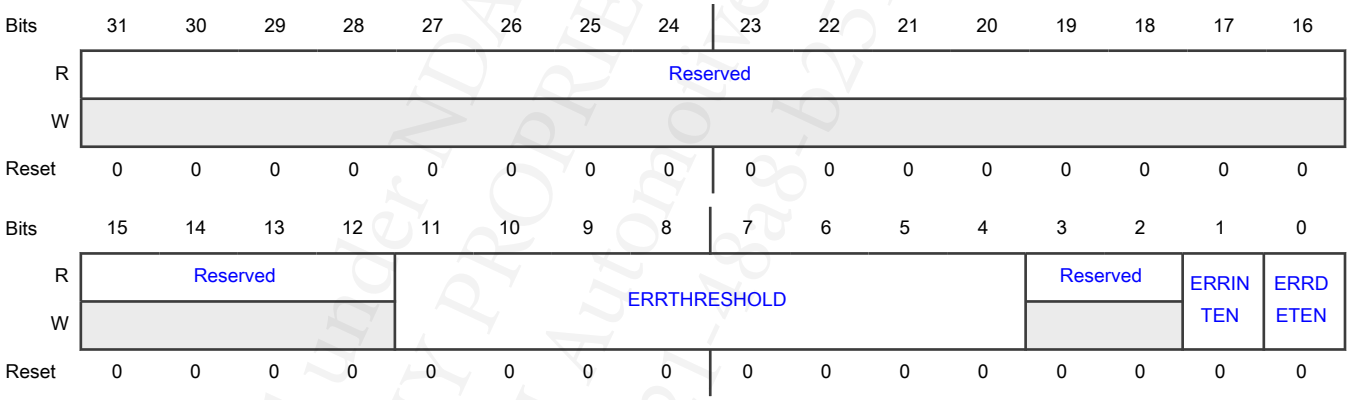
Offset

Register	Offset
CMIUCEC	100h

Function

Use this register to control the detection and signaling of correctable errors in a CMIU.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-4 ERRTHRESHOLD	Correctable Error Threshold Use this field to determine the threshold number of correctable errors that must be detected before Ncore asserts the correctable error interrupt signal.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-2 —	Reserved
1 ERRINTEN	Correctable Error Interrupt Enable Use this field to control assertion of correctable error interrupt signal on fulfilment of proper conditions. 0b - Disable assertion 1b - Enable assertion
0 ERRDETEN	Correctable Error Detection Enable Use this field to control correctable error detection and logging logic. 0b - Disable correctable error detection and logging logic 1b - Enable correctable error detection and logging logic

7.10.4.4 CMIU Correctable Error Status (CMIUCES)

Offset

Register	Offset
CMIUCES	104h

Function

This register logs information about the correctable errors in a CMIU.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								ERRINFO							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRTYPE				ERRCOUNT								Reserved		ERROVF	ERRVLD
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c461e5-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore write appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when CMIUCES[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore write appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when CMIUCES[ERRVLD] = 1; otherwise, the field is undefined. • 6h – CMIU data correctable error. When CMIUCES[ERRINFO] = 1h, it decodes to RTT data buffers
11-4 ERRCOUNT	Error Count This field indicates the number of correctable errors detected by the unit. The field value stops incrementing when you write 1 to CMIUCES[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of correctable errors detected by the unit overflowed the value indicated by CMIUCES[ERRCOUNT]. You write a 1 to this field to clear the CMIUCES[ERRCOUNT] field, and it also clears this field.
0 ERRVLD	Error Valid Ncore writes a 1 to this field to indicate that the following registers have logged information about a correctable error; otherwise, certain fields in these registers are undefined. • CMIU Correctable Error Status (CMIUCES) • CMIU Correctable Error Location Register 0 (CMIUCELR0) • CMIU Correctable Error Location Register 1 (CMIUCELR1)

7.10.4.5 CMIU Correctable Error Location Register 0 (CMIUCELR0)

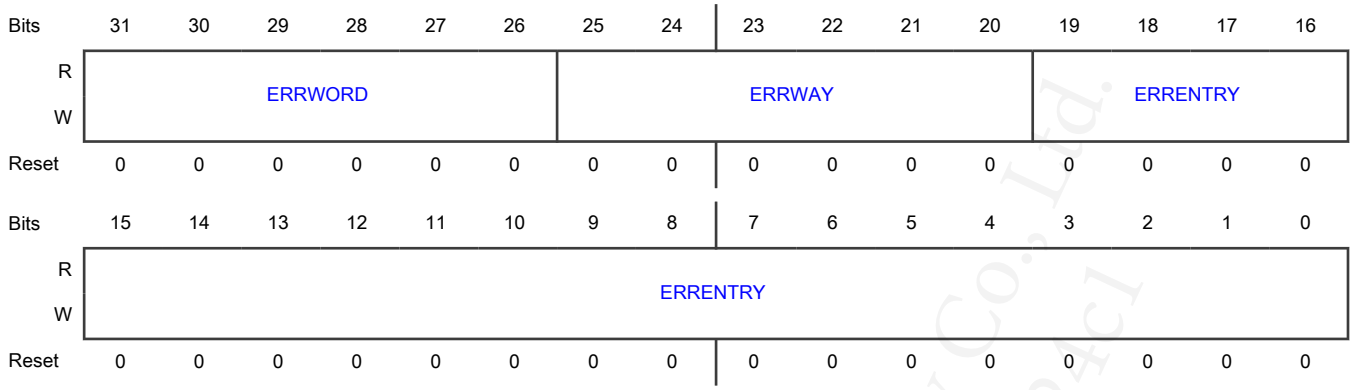
Offset

Register	Offset
CMIUCELR0	108h

Function

This register logs information about the correctable error locations in a CMIU.

Diagram



Fields

Field	Function
31-26 ERRWORD	Error Word When CMIUCES[ERRVLD] = 1, this field indicates the data word in which the logged error was detected; otherwise, the field is undefined.
25-20 ERRWAY	Error Way When CMIUCES[ERRVLD] = 1, this field indicates the way on which the logged error was detected; otherwise, the field is undefined.
19-0 ERRENTRY	Error Entry or Set When CMIUCES[ERRVLD] = 1, this field indicates the entry or the set index on which the logged error was detected; otherwise, the field is undefined.

7.10.4.6 CMIU Correctable Error Location Register 1 (CMIUCELR1)

Offset

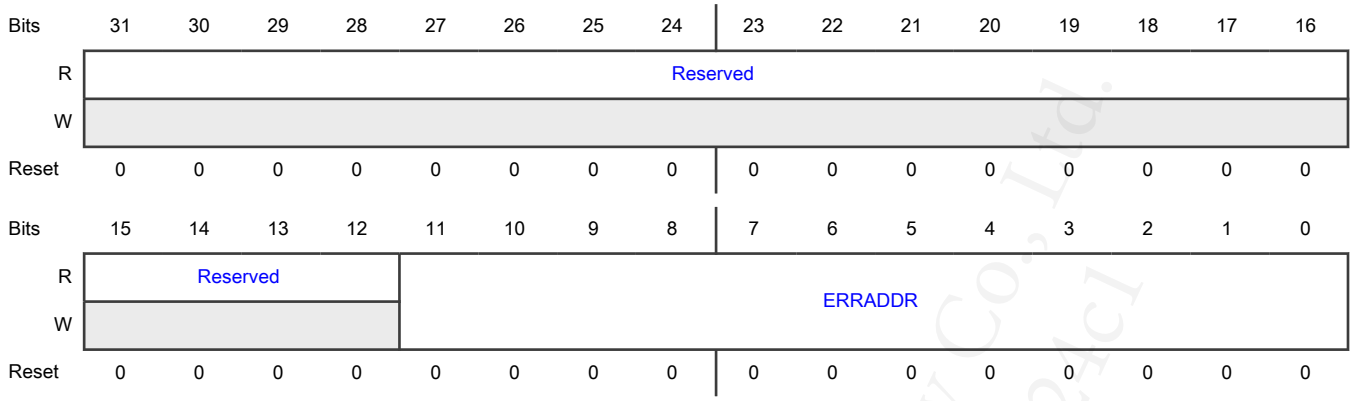
Register	Offset
CMIUCELR1	10Ch

Function

This register logs information about the correctable error locations in a CMIU.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4888-1251-390d3ab624c1

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 ERRADDR	Error Address When CMIUCES[ERRVLD] = 1, this field indicates the high-order address bits of the transaction that encountered the error; otherwise, the field is undefined.

7.10.4.7 CMIU Correctable Error Status Alias (CMIUCESA)

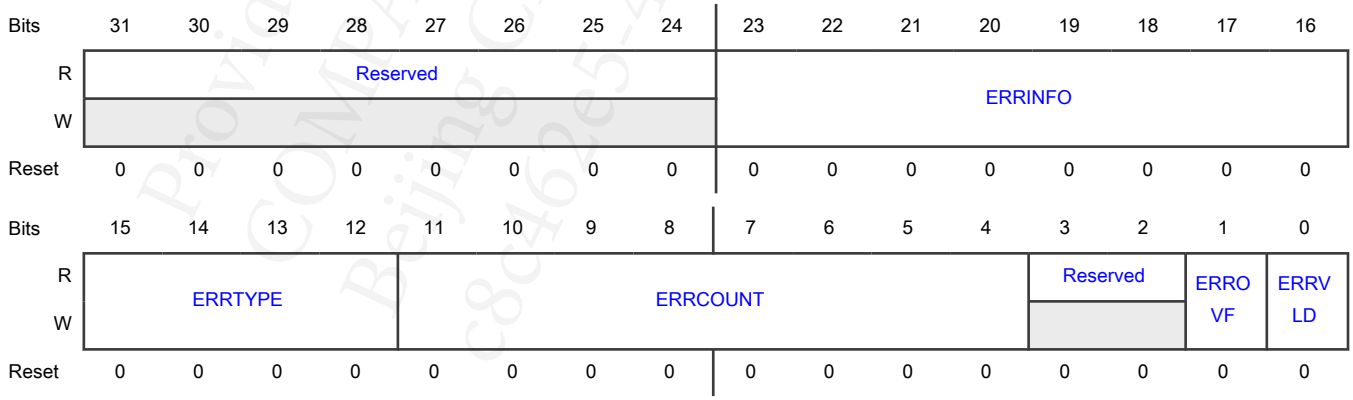
Offset

Register	Offset
CMIUCESA	124h

Function

You use this register to have write access to the CMIU Correctable Error Status (CMIUCES) for diagnostic purposes.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore writes appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when CMIUCESA[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore writes appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when CMIUCESA[ERRVLD] = 1; otherwise, the field is undefined. • 6h – CMIU data correctable error. When CMIUCESA[ERRINFO] = 1, it decodes to RTT data buffers
11-4 ERRCOUNT	Error Count This field indicates the number of correctable errors detected by the unit. The field value stops incrementing when you write a 1 to CMIUCESA[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of correctable errors detected by the unit overflowed the value indicated by CMIUCESA[ERRCOUNT]. You write a 1 to this field to clear the CMIUCESA[ERRCOUNT] field.
0 ERRVLD	Error Valid Ncore writes a 1 to this field to indicate that the following registers have logged information about a correctable error; otherwise, certain fields in these registers are undefined. • CMIU Correctable Error Status (CMIUCES) • CMIU Correctable Error Location Register 0 (CMIUCELR0) • CMIU Correctable Error Location Register 1 (CMIUCELR1)

7.10.4.8 CMIU Uncorrectable Error Control (CMIUUEC)

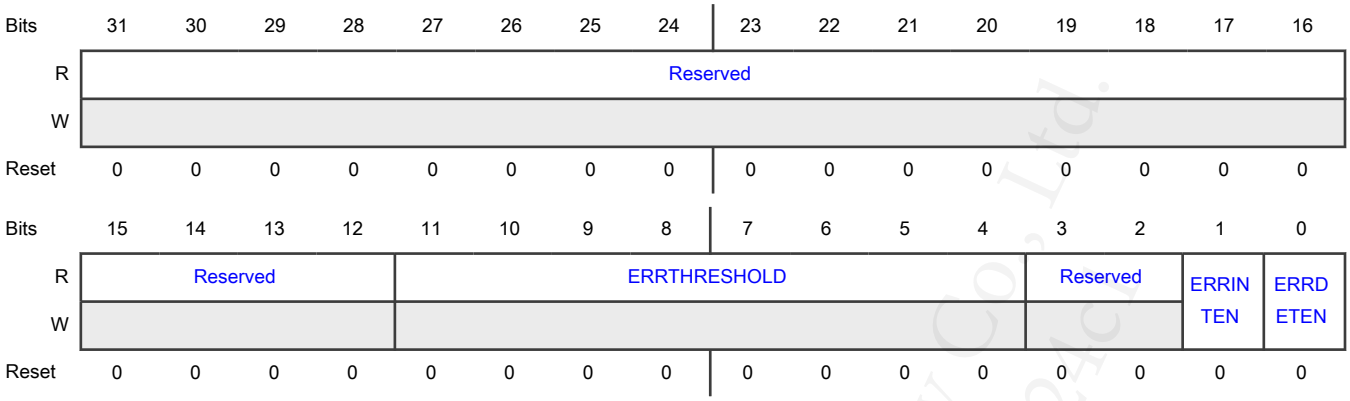
Offset

Register	Offset
CMIUUEC	140h

Function

Use this register to control the detection and signaling of uncorrectable errors in a CMIU.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-4 ERRTHRESHOLD	Uncorrectable Error Threshold Use this field to determine the threshold number of uncorrectable errors that must be detected before Ncore asserts the uncorrectable error interrupt signal.
3-2 —	Reserved
1 ERRINTEN	Uncorrectable Error Interrupt Enable Use this field to control the assertion of uncorrectable error interrupt signal on fulfilment of proper conditions. 0b - Disable assertion 1b - Enable assertion
0 ERRDETEN	Uncorrectable Error Detection Enable Use this field to control uncorrectable error detection and logging logic. 0b - Disable uncorrectable error detection and logging logic 1b - Enable uncorrectable error detection and logging logic

7.10.4.9 CMIU Uncorrectable Error Status (CMIUUES)

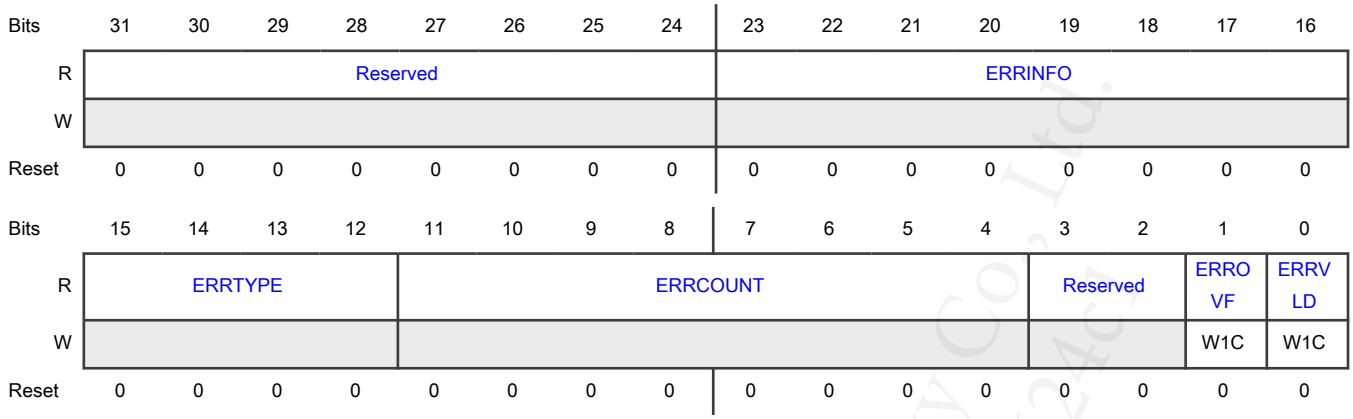
Offset

Register	Offset
CMIUUES	144h

Function

This register logs information about the uncorrectable errors in a CMIU.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore writes appropriate values to this field to indicate additional information about logged errors. Ncore writes to this field only when CMIUUES[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore writes appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when CMIUUES[ERRVLD] = 1; otherwise, the field is undefined. • 6h – CMIU data uncorrectable error. When CMIUUES[ERRINFO] = 1h, it decodes to HTT data buffers
11-4 ERRCOUNT	Error Count This field indicates the number of uncorrectable errors detected by the unit. The field value stops incrementing when you write a 1 to CMIUUES[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of uncorrectable errors detected by the unit overflowed the value indicated by CMIUUES[ERRCOUNT]. You write a 1 to this field to clear the CMIUUES[ERRCOUNT] field.
0 ERRVLD	Error Valid Ncore writes a 1 to this field to indicate that the following registers have logged information about an uncorrectable error; otherwise, certain fields in these registers are undefined.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• CMIU Uncorrectable Error Status (CMIUUES) • CMIU Uncorrectable Error Location Register 0 (CMIUUELR0) • CMIU Uncorrectable Error Location Register 1 (CMIUUELR1)

7.10.4.10 CMIU Uncorrectable Error Location Register 0 (CMIUUELR0)

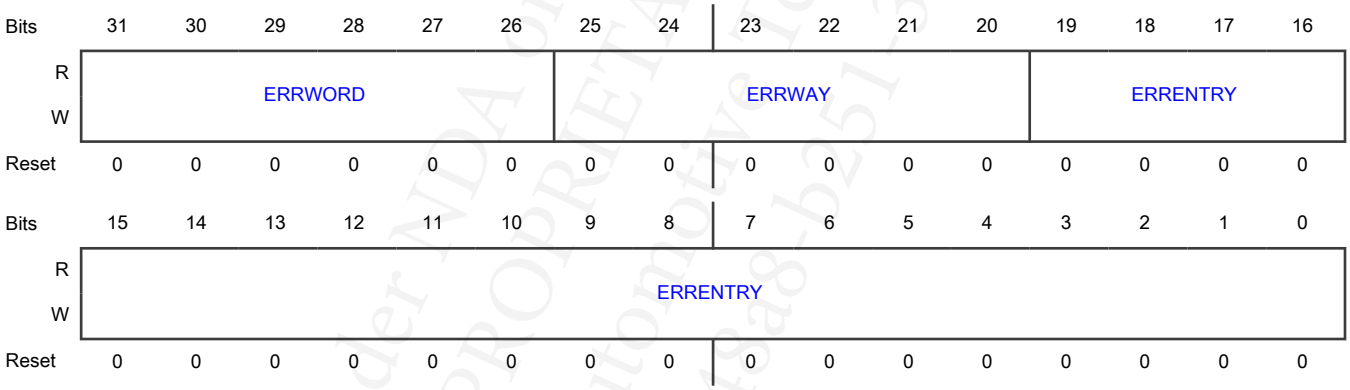
Offset

Register	Offset
CMIUUELR0	148h

Function

This register logs information about uncorrectable error locations in a CMIU.

Diagram



Fields

Field	Function
31-26 ERRWORD	Error Word When CMIUUES[ERRVLD] = 1, this field indicates the data word on which the logged error was detected; otherwise, the field is undefined.
25-20 ERRWAY	Error Way When CMIUUES[ERRVLD] = 1, this field indicates the way on which the logged error was detected; otherwise, the field is undefined.
19-0 ERRENTRY	Error Entry (or Set) When CMIUUES[ERRVLD] = 1, this field indicates the entry or the set index on which the logged error was detected; otherwise, the field is undefined.

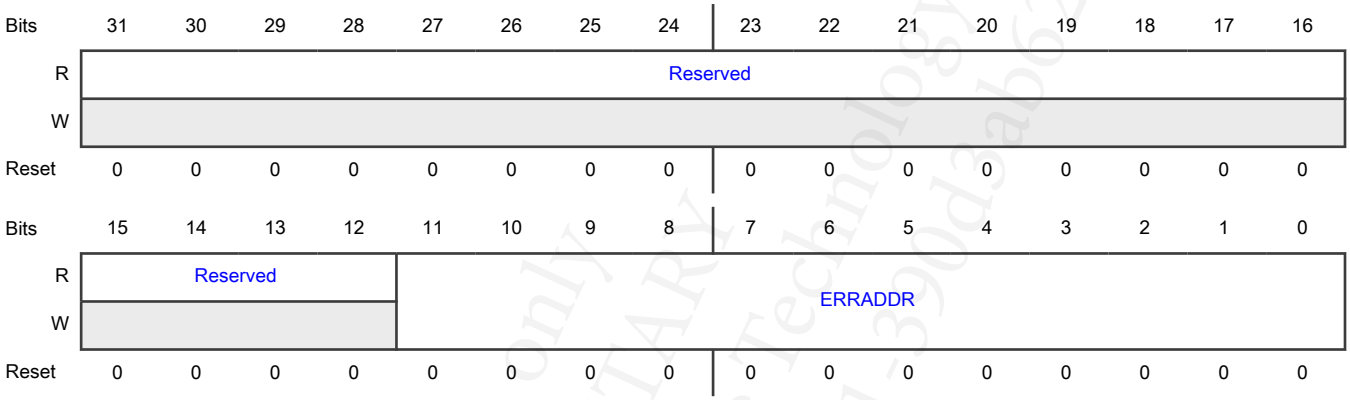
7.10.4.11 CMIU Uncorrectable Error Location Register 1 (CMIUUELR1)

Offset

Register	Offset
CMIUUELR1	14Ch

Function
This register logs information about the uncorrectable error locations in a CMIU.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 ERRADDR	Error Address When CMIUUES[ERRVLD] = 1, this field indicates the high-order address bits of the transaction that encountered the error; otherwise, the field is undefined.

7.10.4.12 CMIU Uncorrectable Error Status Alias (CMIUUESA)

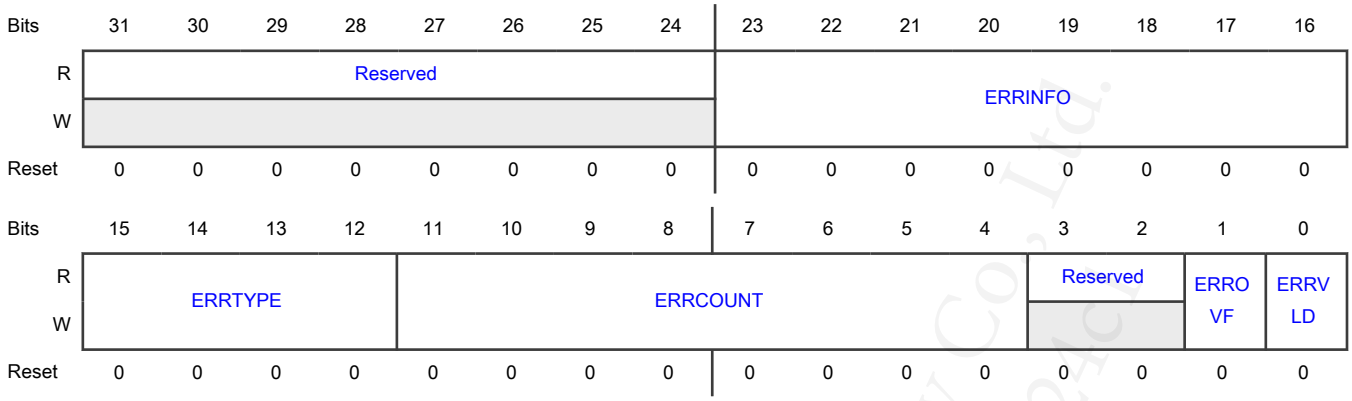
Offset

Register	Offset
CMIUUESA	164h

Function
Use this register to have write access to the CMIU Correctable Error Status (CMIUCES) for diagnostic purposes.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca462e5-4224-48a8-b251-390daab624c1

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 ERRINFO	Error Info Ncore writes appropriate values to this field to indicate the additional information about logged errors. Ncore writes to this field only when CMIUUESA[ERRVLD] = 1; otherwise, the field is undefined.
15-12 ERRTYPE	Error Type Ncore write appropriate values to this field to indicate the type of logged error. Ncore writes to this field only when CMIUUESA[ERRVLD] = 1; otherwise, the field is undefined. • 6h – CMIU data uncorrectable error. When CMIUUES[ERRINFO] = 1h, it decodes to HTT data buffers.
11-4 ERRCOUNT	Error Count This field indicates the number of uncorrectable errors detected by the unit. The field stops incrementing when you write a 1 to CMIUUESA[ERROVF].
3-2 —	Reserved
1 ERROVF	Error Overflow Ncore writes a 1 to this field to indicate that the number of uncorrectable errors detected by the unit overflowed the value indicated by CMIUUESA[ERRCOUNT]. You write a 1 to this field to clear the CMIUUESA[ERRCOUNT] field.
0 ERRVLD	Error Valid Ncore writes a 1 to this field to indicate that the following registers have logged information about an uncorrectable error; otherwise, certain fields in these registers are undefined.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• CMIU Uncorrectable Error Status (CMIUUES) • CMIU Uncorrectable Error Location Register 0 (CMIUUELR0) • CMIU Uncorrectable Error Location Register 1 (CMIUUELR1)

7.10.4.13 CMIU Identification Register (CMIUID)

Offset

Register	Offset
CMIUID	FFCh

Function

Use this register to identify certain features and configuration information about a CMIU.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	Reserved														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		Reserved	CMIID					IMPLVER							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31 —	Reserved
30-14 —	Reserved
13 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
12-8 CMIID	Coherent Memory Interface Identifier Ncore writes to this field to identify the CMI with which the CMIU is associated.
7-0 IMPLVER	Implementation Version Ncore writes to this field to indicate the implementation version of the CMIU. 0b – Unimplemented CMIU 1b – Implemented CMIU

7.10.5 Coherent subsystem (CSR) register descriptions

7.10.5.1 CSR memory map

CSR base address: 504F_F000h

Offset	Register	Width (In bits)	Access	Reset value
40h	Coherent Subsystem ACE DVM Snoop Enable (CSADSE0)	32	RW	0000_0000h
50h	Coherent Subsystem ACE DVM Snoop Activity (CSADSA)	32	RO	0000_0000h
100h	Coherent Subsystem Correctable Error Interrupt Status (CSCEIS0)	32	RO	0000_0000h
10Ch	Coherent Subsystem Correctable Error Interrupt Status (CSCEIS3)	32	RO	0000_0000h
110h	Coherent Subsystem Correctable Error Interrupt Status (CSCEIS4)	32	RO	0000_0000h
118h	Coherent Subsystem Correctable Error Interrupt Status (CSCEIS6)	32	RO	0000_0000h
140h	Coherent Subsystem Uncorrectable Error Interrupt Status (CSUEIS0)	32	RO	0000_0000h
14Ch	Coherent Subsystem Uncorrectable Error Interrupt Status (CSUEIS3)	32	RO	0000_0000h
150h	Coherent Subsystem Uncorrectable Error Interrupt Status (CSUEIS4)	32	RO	0000_0000h
158h	Coherent Subsystem Uncorrectable Error Interrupt Status (CSUEIS6)	32	RO	0000_0000h
F00h	Coherent Subsystem Snoop Filter Identification (CSSFIDR0)	32	RO	0CB0_07FFh
F04h - F7Ch	Coherent Subsystem Snoop Filter Identification (CSSFIDR1 - CSSFIDR31)	32	RO	0000_0000h
FF8h	Coherent Subsystem Unit Identification (CSUID)	32	RO	0101_0202h
FFCh	Coherent Subsystem Identification (CSID)	32	RO	0000_0109h

7.10.5.2 Coherent Subsystem ACE DVM Snoop Enable (CSADSE0)

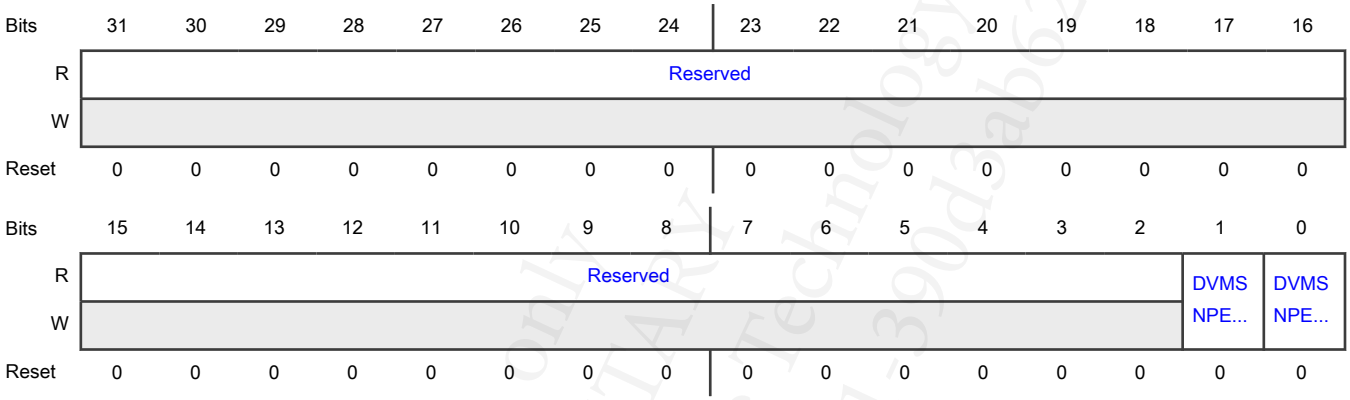
Offset

Register	Offset
CSADSE0	40h

Function

Use this register to enable DVM-related snoop messages to the CAIU associated with a snooped DVM agent.

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 DVMSNPENn	ACE DVM Snoop Enable n Use this field to control DVM-related snoop messages for the respective CAIU. 0b - Disable 1b - Enable

7.10.5.3 Coherent Subsystem ACE DVM Snoop Activity (CSADSA)

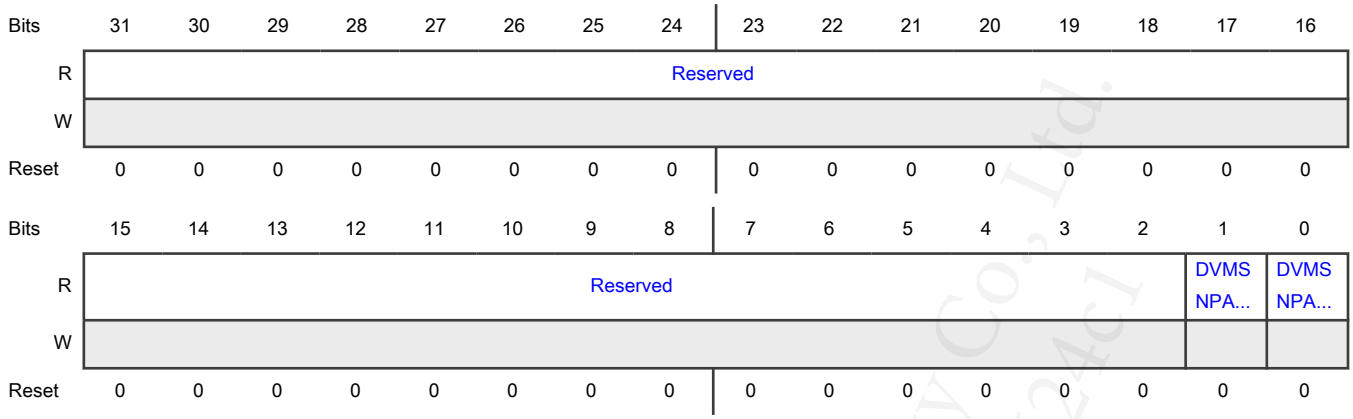
Offset

Register	Offset
CSADSA	50h

Function

Use this register to learn whether the DVM-related snoop messages to the CAIU that is associated with a snooped DVM agent is active or not.

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 DVMSNPACTVn	ACE DVM Snoop Active Ncore writes a 1 to this field if DVM-related snoop messages are active for the respective CAIU.

7.10.5.4 Coherent Subsystem Correctable Error Interrupt Status (CSCEIS0 - CSCEIS3)

Offset

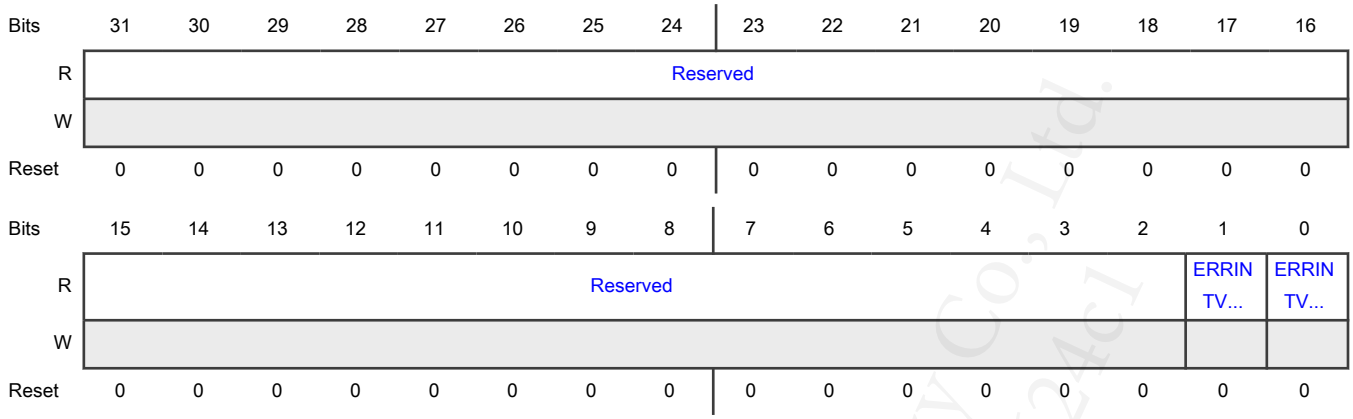
Register	Offset
CSCEIS0	100h
CSCEIS3	10Ch

Function

Use these registers to learn whether a unit has asserted a correctable error interrupt.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4a21-48a8-b351-390d3ab624c1

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 ERRINTVLDn	Error Interrupt Valid Ncore writes a 1 to this field to indicate that a correctable error interrupt is asserted by the respective CAIU.

7.10.5.5 Coherent Subsystem Correctable Error Interrupt Status (CSCEIS4 - CSCEIS6)

Offset

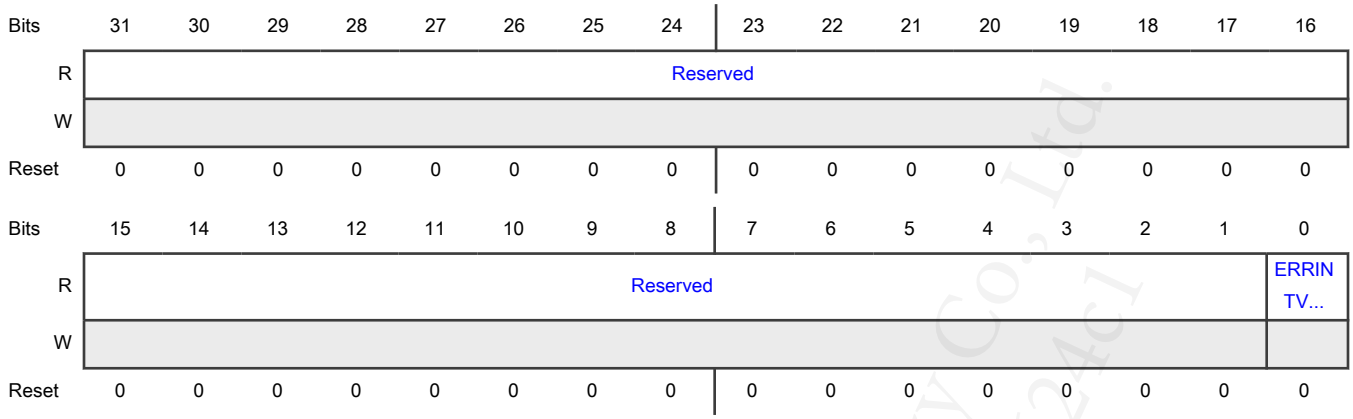
Register	Offset
CSCEIS4	110h
CSCEIS6	118h

Function

Use these registers to learn whether a unit has asserted a correctable error interrupt.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved
0-0 ERRINTVLDn	Error Interrupt Valid Ncore writes a 1 to this field to indicate that a correctable error interrupt is asserted by the respective DIRU.

7.10.5.6 Coherent Subsystem Uncorrectable Error Interrupt Status (CSUEIS0 - CSUEIS3)

Offset

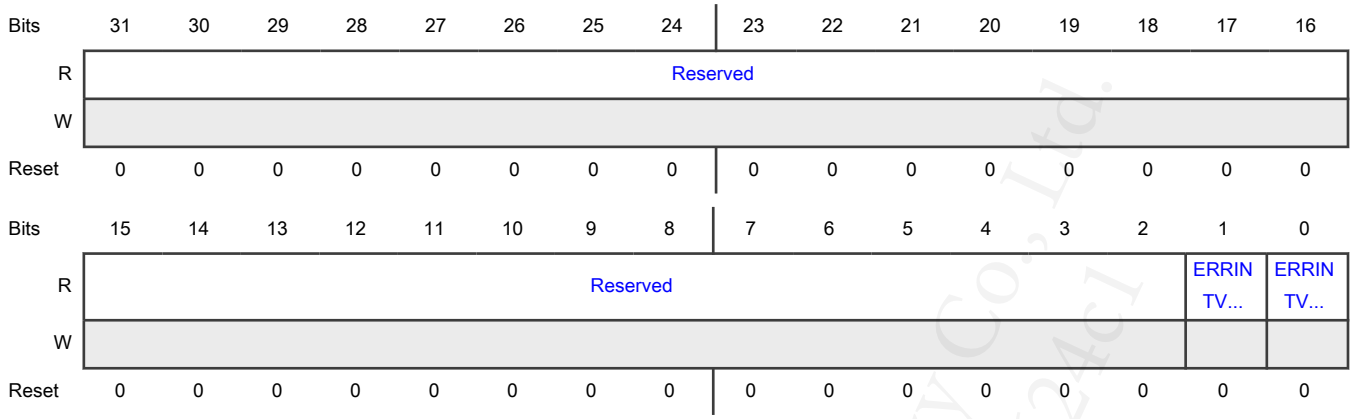
Register	Offset
CSUEIS0	140h
CSUEIS3	14Ch

Function

Use these registers to learn whether a unit has asserted an uncorrectable error interrupt.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 ERRINTVLDn	Error Interrupt Valid Ncore writes a 1 to this field to indicate that an uncorrectable error interrupt is asserted by the respective CAIU.

7.10.5.7 Coherent Subsystem Uncorrectable Error Interrupt Status (CSUEIS4 - CSUEIS6)

Offset

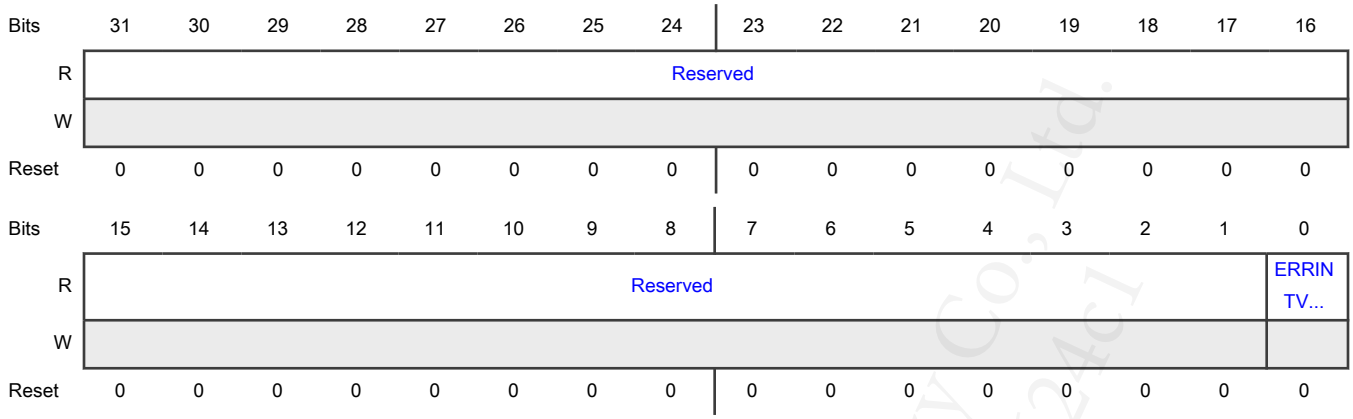
Register	Offset
CSUEIS4	150h
CSUEIS6	158h

Function

Use these registers to learn whether a unit has asserted an uncorrectable error interrupt.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4621-48a8-b513-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved
0-0 ERRINTVLDn	Error Interrupt Valid Ncore writes a 1 to this field to indicate that an uncorrectable error interrupt is asserted by the respective DIRU.

7.10.5.8 Coherent Subsystem Snoop Filter Identification (CSSFIDR0)

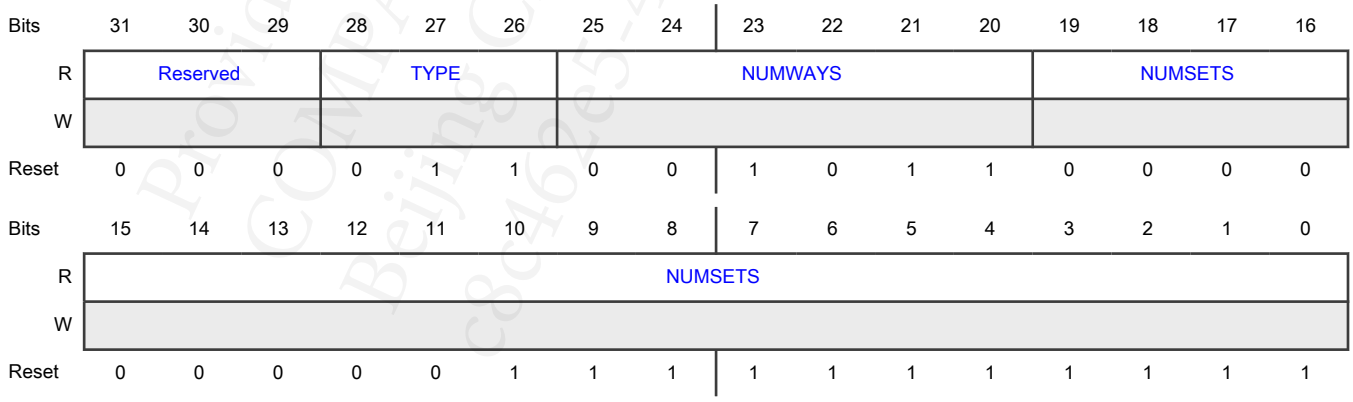
Offset

Register	Offset
CSSFIDR0	F00h

Function

Use these registers to learn the configuration information of the snoop filters in the coherent subsystem.

Diagram



Fields

Field	Function
31-29 —	Reserved
28-26 TYPE	Snoop Filter Type Ncore writes appropriate values to this field to identify the type of snoop filter. 000b - Unimplemented filter 001b - Null filter 010b - Tag filter - presence vector 011b - Tag filter - owner-sharer vector
25-20 NUMWAYS	Number of Snoop Filter Ways Ncore writes a value to this field that equals to 'number of ways (that is, the associativity) in the snoop filter minus one'.
19-0 NUMSETS	Number of Snoop Filter Sets Ncore writes a value to this field that equals to 'number of sets in the snoop filter minus one' in each DIRU. To determine the total number of sets in the snoop filter, increment this value by one and multiply with the number of DIRUs.

7.10.5.9 Coherent Subsystem Snoop Filter Identification (CSSFIDR1 - CSSFIDR31)

Offset

For a = 1 to 31:

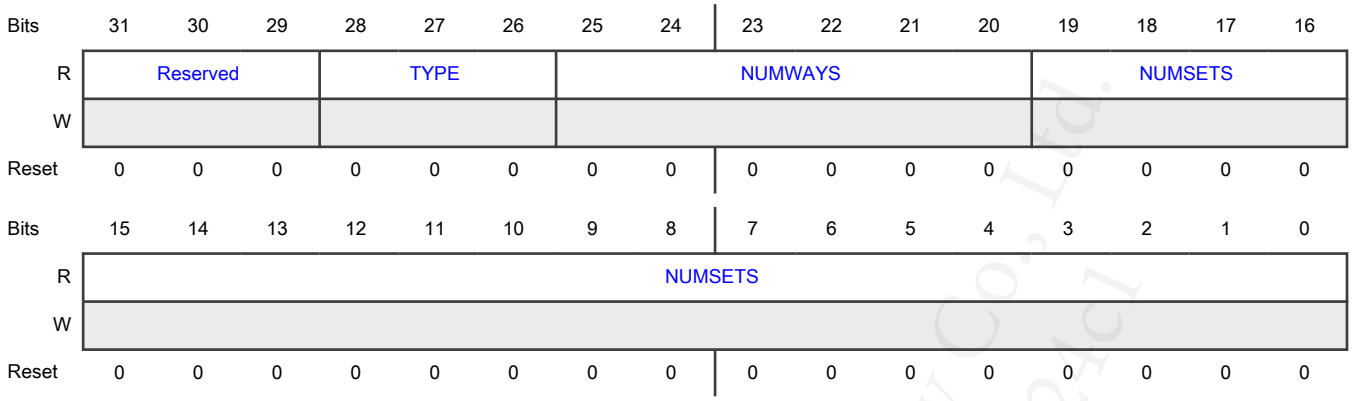
Register	Offset
CSSFIDRa	F00h + (a × 4h)

Function

Use these registers to learn the configuration information of the snoop filters in the coherent subsystem.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4891-48a8-b251-690d3ab624c1

Diagram



Fields

Field	Function
31-29 —	Reserved
28-26 TYPE	Snoop Filter Type Ncore writes appropriate values to this field to identify the type of snoop filter. 000b - Unimplemented filter 001b - Null filter 010b - Tag filter - presence vector 011b - Tag filter - owner-sharer vector
25-20 NUMWAYS	Number of Snoop Filter Ways Ncore writes a value to this field that equals to 'number of ways (that is, the associativity) in the snoop filter minus one'.
19-0 NUMSETS	Number of Snoop Filter Sets Ncore writes a value to this field that equals to 'number of sets in the snoop filter minus one' in each DIRU. To determine the total number of sets in the snoop filter, increment this value by one and multiply with the number of DIRUs.

7.10.5.10 Coherent Subsystem Unit Identification (CSUID)

Offset

Register	Offset
CSUID	FF8h

Function

Use this register to learn the number of units in the coherent subsystem.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		NUMCMIUS						Reserved		NUMDIRUS					
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		NUMNCBUS						Reserved	NUMCAIUS						
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-30 —	Reserved
29-24 NUMCMIUS	Number of CMIUs Ncore writes to this field to indicate the number of CMIUs in the coherent subsystem.
23-22 —	Reserved
21-16 NUMDIRUS	Number of DIRUs Ncore writes to this field to indicate the number of DIRUs in the coherent subsystem.
15-14 —	Reserved
13-8 NUMNCBUS	Number of NCBUs Ncore writes to this field to indicate the number of NCBUs in the coherent subsystem.
7 —	Reserved
6-0 NUMCAIUS	Number of CAIUs Ncore writes to this field to indicate the number of CAIUs in the coherent subsystem.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a6-b251-390d3ab624d1

7.10.5.11 Coherent Subsystem Identification (CSID)

Offset

Register	Offset
CSID	FFCh

Function

Use this register to get the identification information about the coherent subsystem.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								NUMSFS						Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					DIRCLOFFSET			RELVER							
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1

Fields

Field	Function
31-23 —	Reserved
22-18 NUMSFS	Number of Snoop Filters (-1) Ncore writes a value to this field that equals to 'the number of snoop filters minus one'.
17-11 —	Reserved
10-8 DIRCLOFFSET	Directory Cache Line Offset (-5) Ncore writes to this field to indicate the width in bits of the directory cacheline offset. The number of bits in the directory cacheline offset equals the value of this field +5. For example, for a 6-bit offset the value of this field equals one. The size in bytes of the directory cacheline is equal to two to the power of the offset.
7-0 RELVER	Release Version Ncore writes to this field to indicate the release version.

7.10.6 CCTI fault controller register descriptions

7.10.6.1 CCTI_fault_ctrl memory map

CCTI_fault_ctrl base address: 5050_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Core ID (COREID)	32	RO	7073_B716h
4h	Revision ID (REVISIONID)	32	RO	C67F_8700h
8h	Mission Fault 0 (MISSION_FAULT0)	32	RO	0000_0000h
Ch	Mission Fault 1 (MISSION_FAULT1)	32	RO	0000_0000h
10h	Mission Fault 2 (MISSION_FAULT2)	32	RO	0000_0000h
14h	Mission Fault 3 (MISSION_FAULT3)	32	RO	0000_0000h
18h	Latent Fault 0 (LATENT_FAULT0)	32	RO	0000_0000h
1Ch	Latent Fault 1 (LATENT_FAULT1)	32	RO	0000_0000h
20h	Latent Fault 2 (LATENT_FAULT2)	32	RO	0000_0000h
24h	Latent Fault 3 (LATENT_FAULT3)	32	RO	0000_0000h
28h	Faults (FAULTS)	32	RO	0000_0000h
2Ch	Interrupt Enable (INTEN)	32	RW	0000_0000h
30h	Interrupt Clear (INTCLR)	32	W1C	0000_0000h
34h	BIST Control (BISTCTL)	32	W1C	0000_0000h
38h	BIST Done (BIST_DONE)	32	RO	0000_0002h
3Ch	BIST Timeout 1 (BIST_TO1)	32	RW	0000_FFFFh
40h	BIST Timeout 2 (BIST_TO2)	32	RW	0000_00FFh

7.10.6.2 Core ID (COREID)

Offset

Register	Offset
COREID	0h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39016ab6a1a1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CORECHECKSUM															
W																
Reset	0	1	1	1	0	0	0	0	0	1	1	1	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CORECHECKSUM								CORETYPEID							
W																
Reset	1	0	1	1	0	1	1	1	0	0	0	1	0	1	1	0

Fields

Field	Function
31-8 CORECHECKSUM	Core Checksum This field contains a checksum of the parameters of the module.
7-0 CORETYPEID	Core Type ID This field identifies the type of core.

7.10.6.3 Revision ID (REVISIONID)

Offset

Register	Offset
REVISIONID	4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NOCID															
W																
Reset	1	1	0	0	0	1	1	0	0	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NOCID								USERID							
W																
Reset	1	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-4861-48a8-b251-390d3ab624c1

Fields

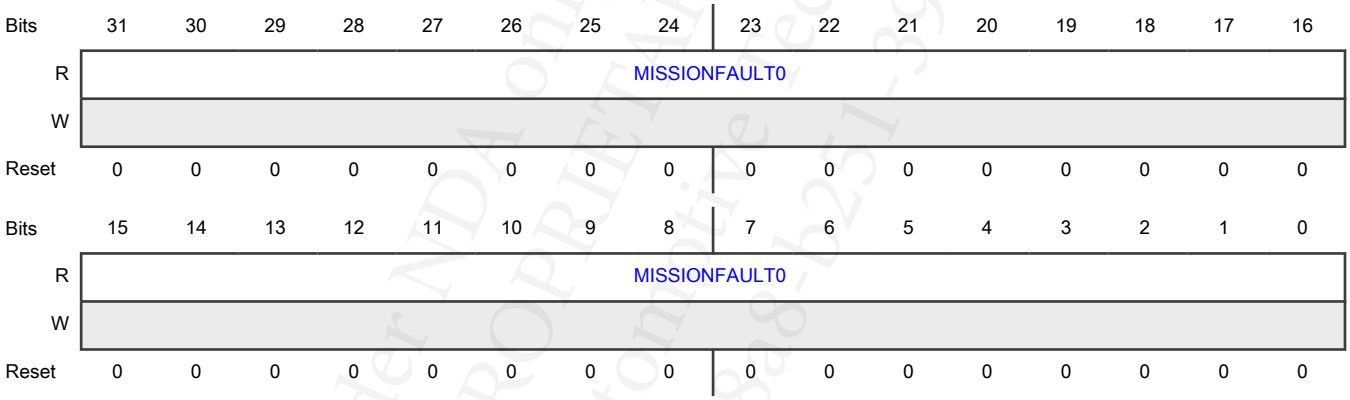
Field	Function
31-8 NOCID	NoC ID This field contains the build revision of the software that generates the module's HDL code.
7-0 USERID	User ID This field contains a user defined value. The field is not used anywhere inside the module.

7.10.6.4 Mission Fault 0 (MISSION_FAULT0)

Offset

Register	Offset
MISSION_FAULT0	8h

Diagram



Fields

Field	Function
31-0 MISSIONFAULT0	Mission Fault This field reports the mission faults for the Ncore components. The following table provides a mapping between the bit position and corresponding Ncore unit for which faults are reported.

Bit position	Ncore component
0	FSC
1	FSC
2	NCBU1
3	NCBU1
4	NCBU0

Field	Function

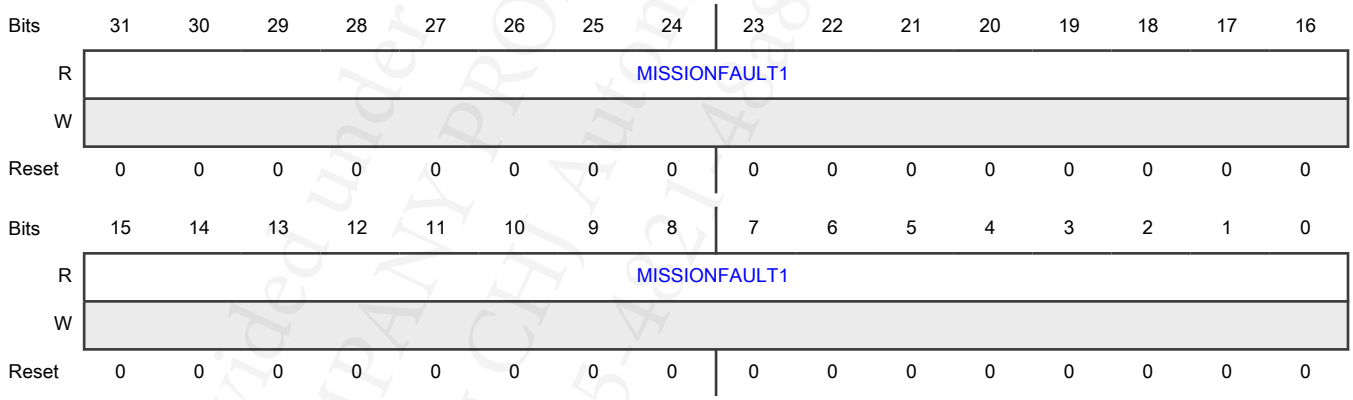
Bit position	Ncore component
5	NCBU0
6	CMIU
7	CMIU
8	DIRU
9	DIRU
10	CAIU1
11	CAIU1
12	CAIU0
13	CAIU0

7.10.6.5 Mission Fault 1 (MISSION_FAULT1)

Offset

Register	Offset
MISSION_FAULT1	Ch

Diagram



Fields

Field	Function
31-0 MISSIONFAULT1	Mission Fault This field does not report mission faults for any of the Ncore components.

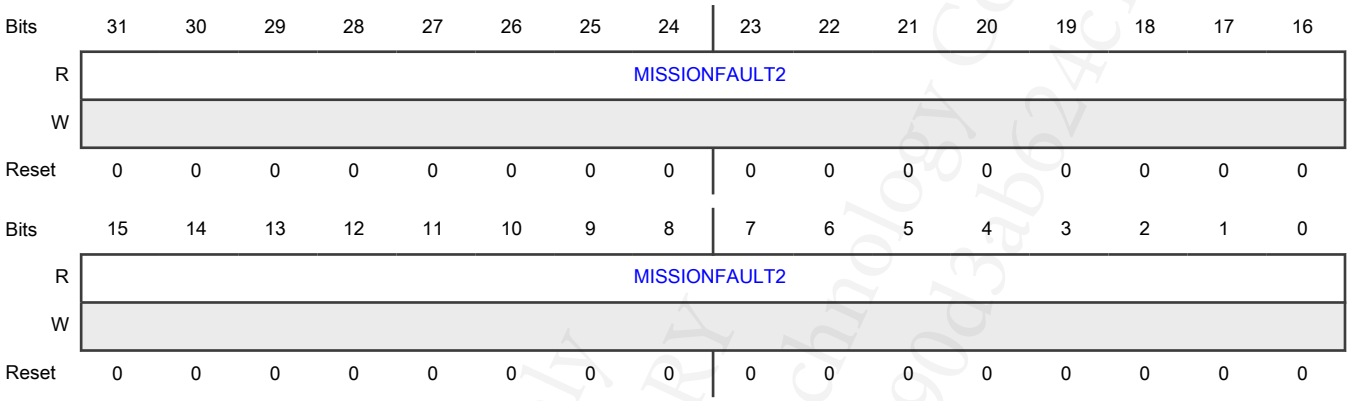
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4c1f-4e8-b251-390a3ab624c1

7.10.6.6 Mission Fault 2 (MISSION_FAULT2)

Offset

Register	Offset
MISSION_FAULT2	10h

Diagram



Fields

Field	Function
31-0 MISSIONFAULT2	Mission Fault This field does not report mission faults for any of the Ncore components.

7.10.6.7 Mission Fault 3 (MISSION_FAULT3)

Offset

Register	Offset
MISSION_FAULT3	14h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-9251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MISSIONFAULT3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MISSIONFAULT3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MISSIONFAULT3	Mission Fault This field does not report mission faults for any of the Ncore components.

7.10.6.8 Latent Fault 0 (LATENT_FAULT0)

Offset

Register	Offset
LATENT_FAULT0	18h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LATENTFAULT0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LATENTFAULT0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e54821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-0 LATENTFAULT0	Latent Fault This field reports the latent faults for the Ncore components. The following table provides a mapping between the bit position and corresponding Ncore unit for which faults are reported.

Bit position	Ncore component
0	FSC
1	FSC
2	NCBU1
3	NCBU1
4	NCBU0
5	NCBU0
6	CMIU
7	CMIU
8	DIRU
9	DIRU
10	CAIU1
11	CAIU1
12	CAIU0
13	CAIU0

7.10.6.9 Latent Fault 1 (LATENT_FAULT1)

Offset

Register	Offset
LATENT_FAULT1	1Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a6-8251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LATENTFAULT1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LATENTFAULT1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 LATENTFAULT1	Latent Fault This field does not report latent faults for any of the Ncore components.

7.10.6.10 Latent Fault 2 (LATENT_FAULT2)

Offset

Register	Offset
LATENT_FAULT2	20h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LATENTFAULT2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LATENTFAULT2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Fields

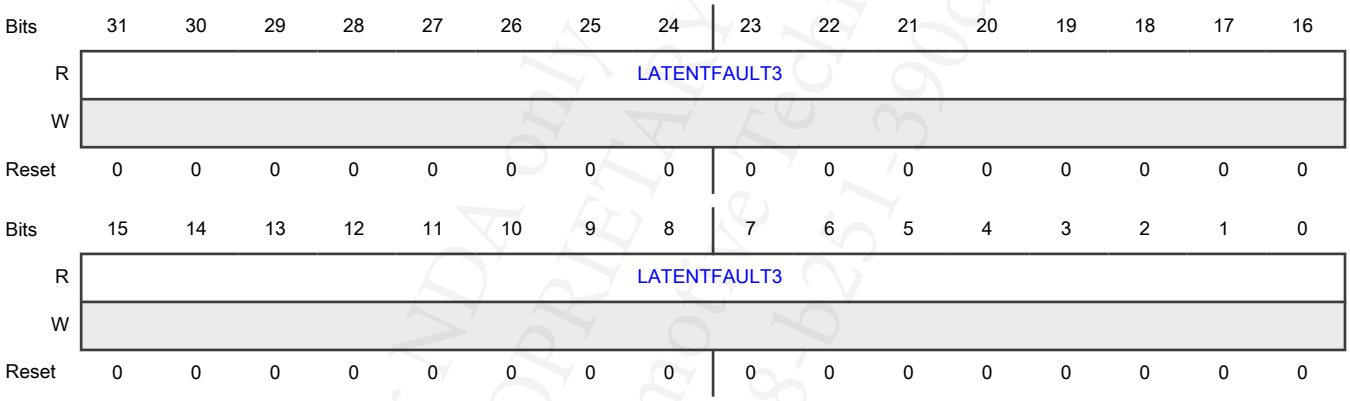
Field	Function
31-0 LATENTFAULT2	Latent Fault This field does not report latent faults for any of the Ncore components.

7.10.6.11 Latent Fault 3 (LATENT_FAULT3)

Offset

Register	Offset
LATENT_FAULT3	24h

Diagram



Fields

Field	Function
31-0 LATENTFAULT3	Latent Fault This field does not report latent faults for any of the Ncore components.

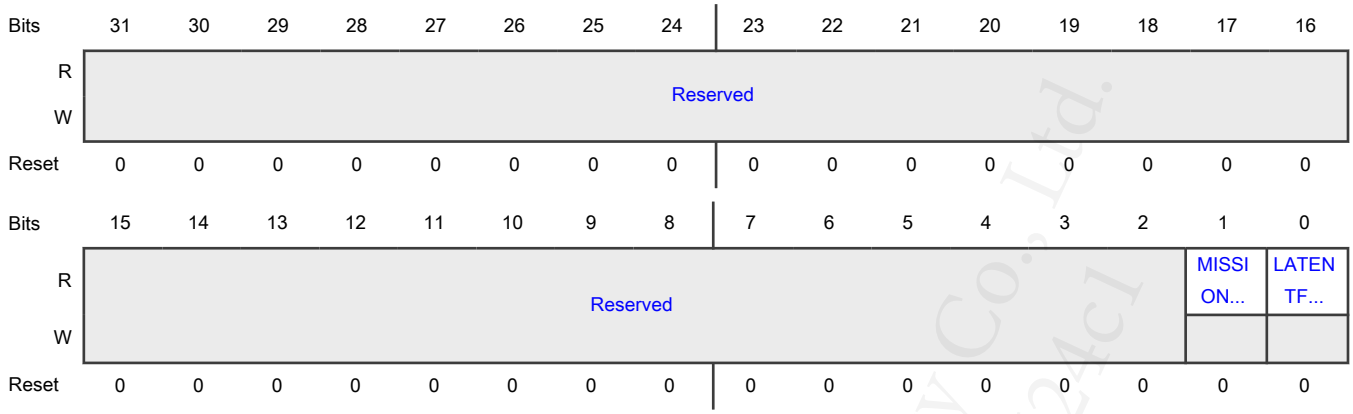
7.10.6.12 Faults (FAULTS)

Offset

Register	Offset
FAULTS	28h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
ca462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 —	Reserved
1 MISSIONFAULT	Mission Fault
0 LATENTFAULT	Latent Fault

7.10.6.13 Interrupt Enable (INTEN)

Offset

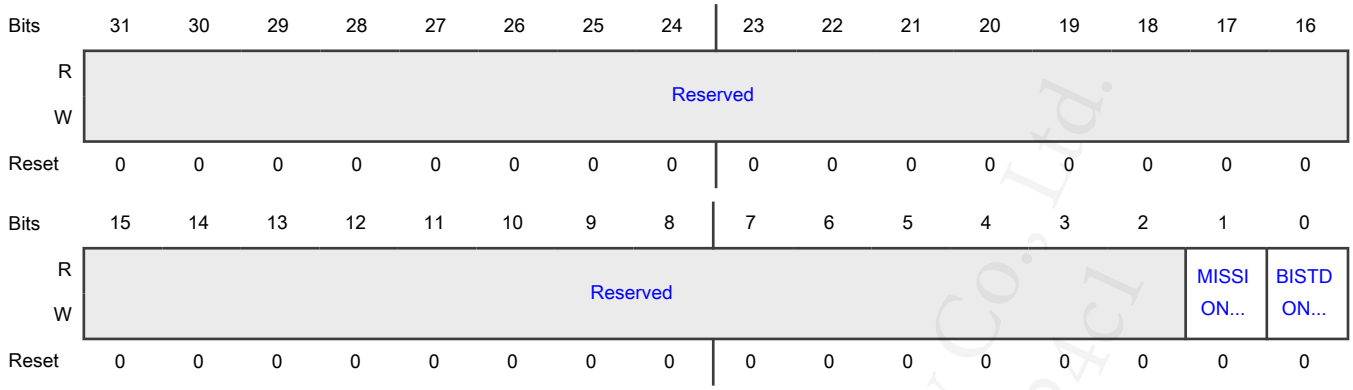
Register	Offset
INTEN	2Ch

Function

Use this register to enable generation of interrupt on different events.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 —	Reserved
1 MISSIONFAULTEN	Mission Fault Interrupt Enable Write a 1 to this field to enable generation of interrupt on a mission fault.
0 BISTDONEEN	BIST Done Interrupt Enable Write a 1 to this field to enable generation of interrupt on BIST done.

7.10.6.14 Interrupt Clear (INTCLR)

Offset

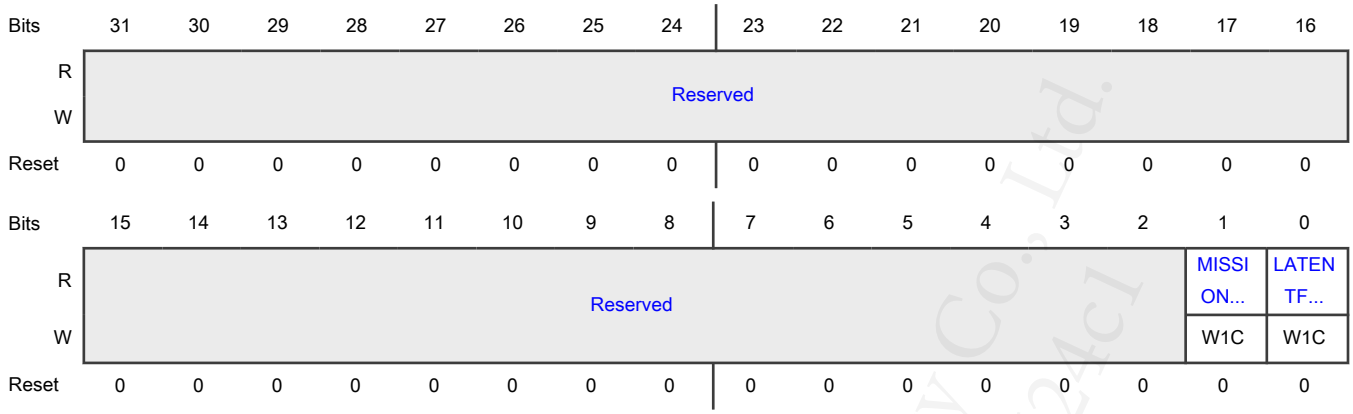
Register	Offset
INTCLR	30h

Function

Use this field to clear the interrupts generated on faults.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 —	Reserved
1 MISSIONFAULTCLR	Clear Mission Fault Write a 1 to this field to clear the interrupt on mission faults.
0 LATENTFAULTCLR	Clear Latent Fault Write a 1 to this field to clear the interrupt on latent faults.

7.10.6.15 BIST Control (BISTCTL)

Offset

Register	Offset
BISTCTL	34h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														BISTDONE...	BISTSTA...
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	Reserved
1 BISTDONECLR	Clear BIST Done Write a 1 to this field to clear the BIST_DONE[BISTDONE] field.
0 BISTSTART	Start BIST Sequence Write a 1 to this field to command the BIST FSM to start the BIST sequence.

7.10.6.16 BIST Done (BIST_DONE)

Offset

Register	Offset
BIST_DONE	38h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														MISSION...	BISTDONE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Fields

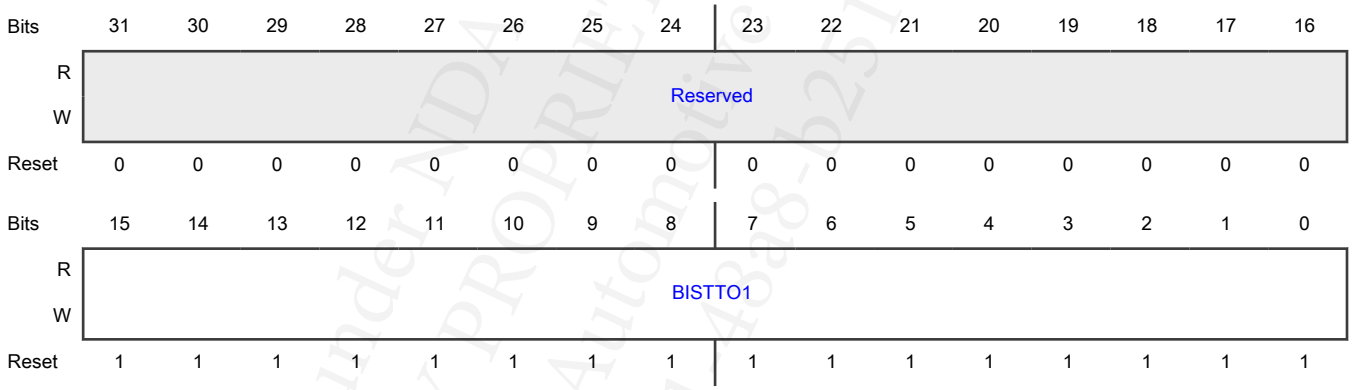
Field	Function
31-2 —	Reserved
1 MISSIONMODE	Mission Mode Status
0 BISTDONE	BIST Done Status Ncore writes 1 to this field to indicate completion of the BIST sequence.

7.10.6.17 BIST Timeout 1 (BIST_TO1)

Offset

Register	Offset
BIST_TO1	3Ch

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 BISTTO1	BIST Timeout

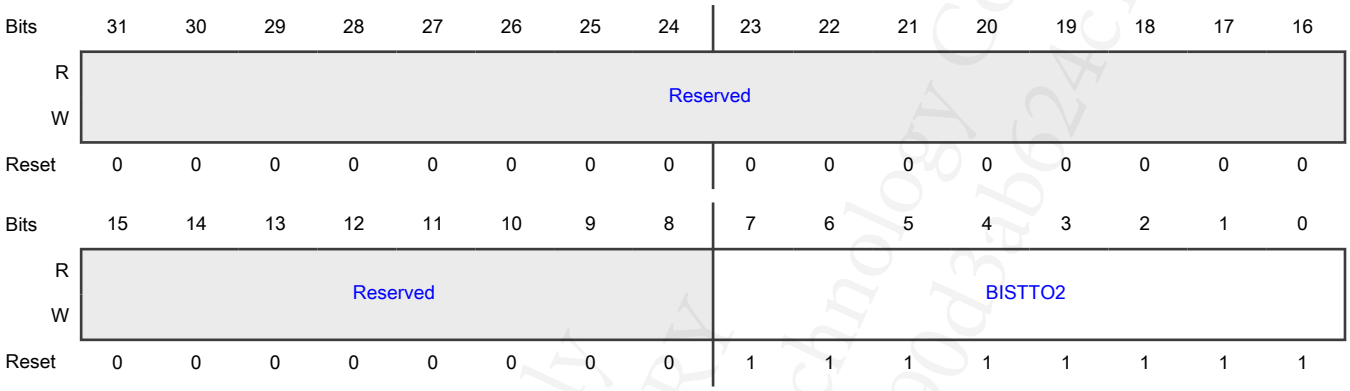
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

7.10.6.18 BIST Timeout 2 (BIST_TO2)

Offset

Register	Offset
BIST_TO2	40h

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 BISTTO2	BIST Timeout

7.10.7 Functional safety controller (FSC) register descriptions

7.10.7.1 FSC memory map

FSC base address: 5060_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	FSC BIST Control (SCBISTC)	32	WORZ	0000_0000h
4h	FSC BIST Activity (SCBISTA)	32	RO	0000_0000h
8h	FSC Correctable Errors Threshold (SCCETH)	32	RW	0000_0001h
10h	FSC Latent Fault (SCLF0)	32	RO	0000_0000h
20h	FSC Mission Fault (SCMF0)	32	RO	0000_0000h
30h	FSC Correctable Error Threshold Fault (SCCETHF0)	32	RO	0000_0000h

7.10.7.2 FSC BIST Control (SCBISTC)

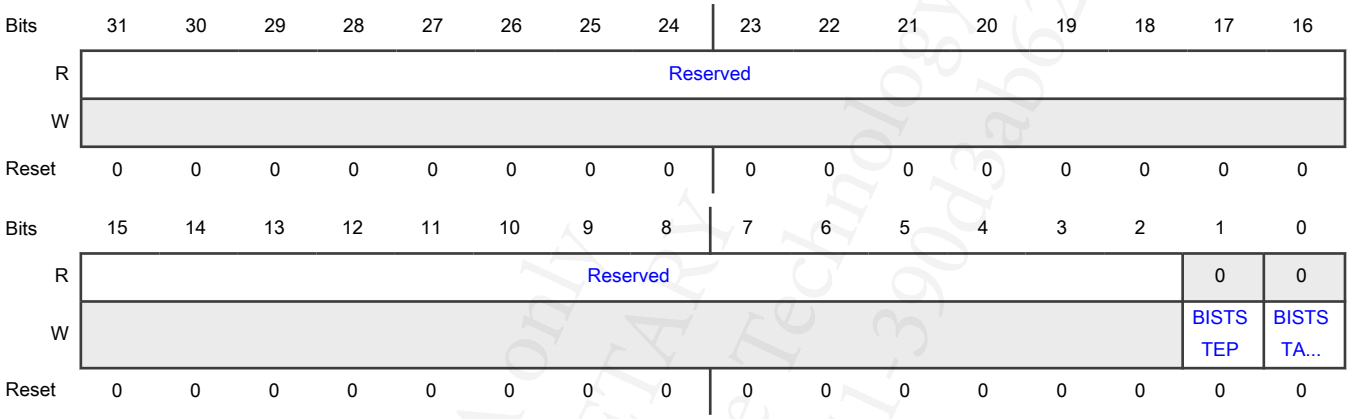
Offset

Register	Offset
SCBISTC	0h

Function

Use this register to control the BIST FSM for executing the BIST sequence. (See the BIST sequence).

Diagram



Fields

Field	Function
31-2 —	Reserved
1 BISTSTEP	BIST Step Write a 1 to this field to command the BIST FSM to move to the next step in the BIST sequence.
0 BISTSTART	BIST Start Write a 1 to this field to command the BIST FSM to start the BIST sequence.

7.10.7.3 FSC BIST Activity (SCBISTA)

Offset

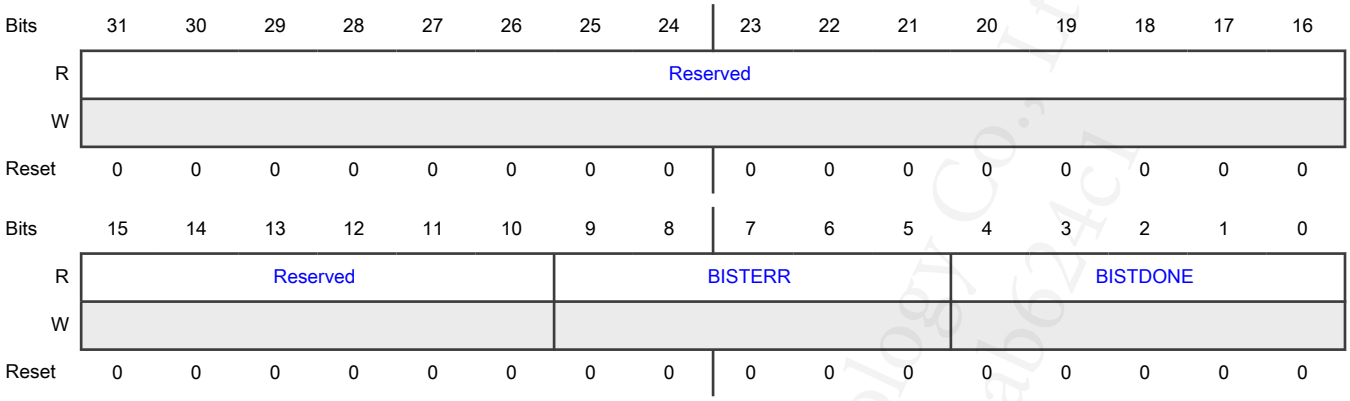
Register	Offset
SCBISTA	4h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390da8b624c1

Function

Use this register to learn about any BIST activity, that is to learn wehether any error has been reported or whether the BIST sequence has completed. (See the BIST sequence)

Diagram



Fields

Field	Function
31-10 —	Reserved
9-5 BISTERR	BIST Error Each bit position in this field represents one step of the BIST sequence. As each step of the BIST sequence completes and if the sequence has completed with an error, Ncore writes 1 to the respective bit position of the field. • Bit position 5 corresponds to step 1 of the BIST sequence • Bit position 6 corresponds to step 2 of the BIST sequence • Bit position 7 corresponds to step 3 of the BIST sequence • Bit position 8 corresponds to step 4 of the BIST sequence • Bit position 9 corresponds to step 5 of the BIST sequence This field resets on starting a BIST sequence (that is, when you write 1 to SCBISTC[BISTSTART]).
4-0 BISTDONE	BIST Done Each bit position in this field represents one step of the BIST sequence. As each step of the BIST sequence completes, Ncore writes a 1 to the respective bit position of the field. • Bit position 0 corresponds to step 1 of the BIST sequence • Bit position 1 corresponds to step 2 of the BIST sequence • Bit position 2 corresponds to step 3 of the BIST sequence • Bit position 3 corresponds to step 4 of the BIST sequence • Bit position 4 corresponds to step 5 of the BIST sequence This field resets on starting a BIST sequence (that is, when you write 1 to SCBISTC[BISTSTART]).

7.10.7.4 FSC Correctable Errors Threshold (SCCETH)

Offset

Register	Offset
SCCETH	8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								CERRTHRES							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-8 —	Reserved
7-0 CERRTHRES	Correctable Errors Threshold Use this field to determine the threshold number of correctable errors that must be detected before Ncore asserts the correctable error beyond programmed threshold signal.

7.10.7.5 FSC Latent Fault (SCLF0)

Offset

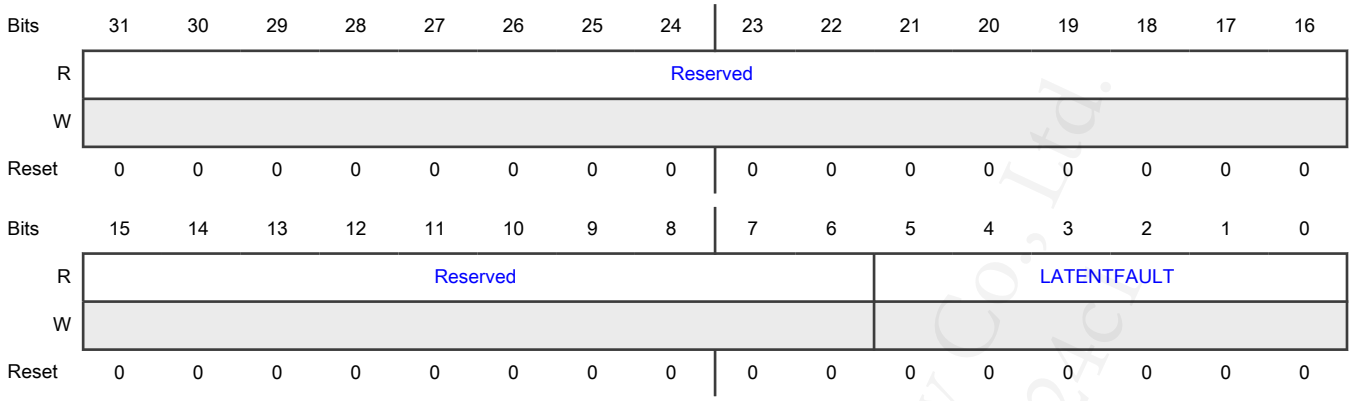
Register	Offset
SCLF0	10h

Function

Use this register to learn the status of latent faults for different Ncore components. See "Latent fault" in Component logic duplication to learn what causes the safety checker to report this fault.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e5-4821-4008-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 LATENTFAULT	Latent Fault Ncore reports the status of the latent faults corresponding to each of the Ncore components in this field. The table below shows the bit positions of this field that reports the status of the faults corresponding to each of the Ncore components:

Bit position	Ncore component
0	CAIU0
1	CAIU1
2	NCBU0
3	NCBU1
4	DIRU
5	CMIU

7.10.7.6 FSC Mission Fault (SCMF0)

Offset

Register	Offset
SCMF0	20h

Function

Use this register to learn the status of mission faults for different Ncore components. See "Mission fault" in Component logic duplication to learn what causes the safety checker to report this fault.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4821-4825-390d8ab624c

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										MISSIONFAULT					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5-0 MISSIONFAULT	Mission Faults Ncore reports the status of the mission faults corresponding to each of the Ncore components in this field. The table below shows the bit positions of this field that reports the status of the faults corresponding to each of the Ncore components:

Bit position	Ncore component
0	CAIU0
1	CAIU1
2	NCBU0
3	NCBU1
4	DIRU
5	CMIU

7.10.7.7 FSC Correctable Error Threshold Fault (SCCETHF0)

Offset

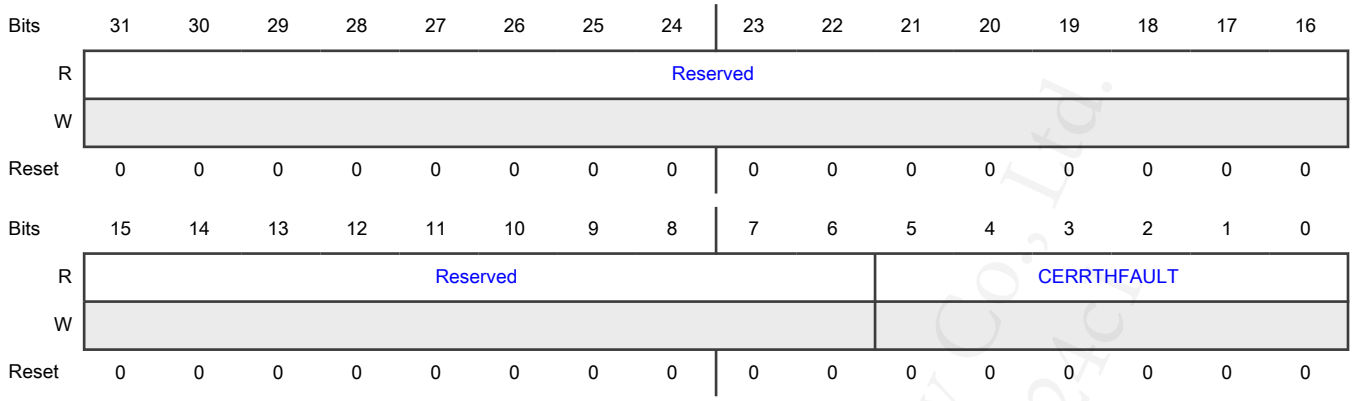
Register	Offset
SCCETHF0	30h

Function

Use this register to learn the status of correctble errors above threshold fault for different Ncore components.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4821-480-9025-390d8ab624c

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 CERRTHFAULT	Correctable Error Threshold Fault Ncore reports the status of correctable errors above threshold fault in this field. The table below shows the bit positions of this field that report the faults corresponding to each of the Ncore components:

Bit position	Ncore component
0	CAIU0
1	CAIU1
2	NCBU0
3	NCBU1
4	DIRU
5	CMIU

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4825-390d8ab624c

Chapter 8
Miscellaneous System Control Module (MSCM)

8.1 Introduction

MSCM contains CPU configuration registers and on-chip memory controller registers.

8.1.1 Features

MSCM provides:

- Software-accessible processor core configuration information
- Interrupt router control registers
- Message-based interrupt configuration and management registers

8.2 Memory map and register definition

This section discusses the following topics:

- Core configuration registers
- Message-signaled interrupt (MSI) routing
  - Core-to-core MSIs
  - PCIe-to-core message-signaled interrupts
- Shared peripheral interrupt routing

8.2.1 Core configuration registers

These read-only registers contain data that defines the core setup for this chip. You can access the registers using 32-bit read references; other access sizes terminate with an error. Attempted write accesses to the read-only core configuration registers also terminate with an error.

The core configuration portion of the MSCM programming model map is organized based on the logical core number, and not on any type of physical port number. The following table shows how the configuration is partitioned.

Table 22. MSCM core configuration partitioning

Offset address range	Purpose
0h - 18h	Defines the generic core x configuration information. Only the Cortex-A53 and Cortex-M7 cores on this chip can access this region in either user or privileged mode; reads by non-core[1] bus masters (including the debugger) are treated as read as zero (RAZ) accesses. Write attempts are not permitted and terminate with an error.
20h - 38h	Defines the configuration information for core 0 (CP0). A bus master can access this region in either user or privileged mode. Write attempts are not permitted and terminate with an error.
40h - 58h	Defines the configuration information for core 1 (CP1). A bus master can access this region in either user or privileged mode. Write attempts are not permitted and terminate with an error.
60h - 78h	Defines the configuration information for core 2 (CP2). A bus master can access this region in either user or privileged mode. Write attempts are not permitted and terminate with an error.
80h - 98h	Defines the configuration information for core 3 (CP3). A bus master can access this region in either user or privileged mode. Write attempts are not permitted and terminate with an error.

Table continues on the next page...

Table 22. MSCM core configuration partitioning (continued)

Offset address range	Purpose
A0h - B8h	Defines the configuration information for core 4 (CP4). A bus master can access this region in either user or privileged mode. Write attempts are not permitted and terminate with an error.
C0h - D8h	Defines the configuration information for core 5 (CP5). A bus master can access this region in either user or privileged mode. Write attempts are not permitted and terminate with an error.
E0h - F8h	Defines the configuration information for core 6 (CP6). A bus master can access this region in either user or privileged mode. Write attempts are not permitted and terminate with an error.

1. A non-core bus master is one that is not Cortex-A53 or Cortex-M7.

NOTE

Attempted accesses to reserved locations are not permitted and terminate with an error. Debugger accesses referred to in this document are accesses from the system MEM-AP and not the core accesses through debugger.

8.2.2 Shared peripheral interrupt (SPI) routing

The SPI router portion of MSCM provides a set of memory-mapped registers defining the interrupt routing for all SPIs on this chip.

8.2.3 MSCM register descriptions

8.2.3.1 MSCM memory map

MSCM base address: 4019_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Processor X Type Register (CPXTYPE)	32	RO	See description
4h	Processor X Number Register (CPXNUM)	32	RO	See description
8h	Processor X Revision Register (CPXREV)	32	RO	See description
Ch	Processor X Configuration 0 Register (CPXCFG0)	32	RO	See description
10h	Processor X Configuration 1 Register (CPXCFG1)	32	RO	See description
14h	Processor X Configuration 2 Register (CPXCFG2)	32	RO	See description
18h	Processor X Configuration 3 Register (CPXCFG3)	32	RO	See description
20h	Processor Type Register (CP0TYPE)	32	RO	4135_3330h
24h	Processor Number Register (CP0NUM)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
28h	Processor Count Register (CP0REV)	32	RO	0000_0004h
2Ch	Processor Configuration 0 Register (CP0CFG0)	32	RO	0702_0704h
30h	Processor Configuration 1 Register (CP0CFG1)	32	RO	0B10_0000h
34h	Processor Configuration 2 Register (CP0CFG2)	32	RO	0000_0000h
38h	Processor Configuration 3 Register (CP0CFG3)	32	RO	0000_0017h
40h	Processor Type Register (CP1TYPE)	32	RO	4135_3331h
44h	Processor Number Register (CP1NUM)	32	RO	0000_0001h
48h	Processor Count Register (CP1REV)	32	RO	0000_0004h
4Ch	Processor Configuration 0 Register (CP1CFG0)	32	RO	0702_0704h
50h	Processor Configuration 1 Register (CP1CFG1)	32	RO	0B10_0000h
54h	Processor Configuration 2 Register (CP1CFG2)	32	RO	0000_0000h
58h	Processor Configuration 3 Register (CP1CFG3)	32	RO	0000_0017h
60h	Processor Type Register (CP2TYPE)	32	RO	4135_3332h
64h	Processor Number Register (CP2NUM)	32	RO	0000_0002h
68h	Processor Count Register (CP2REV)	32	RO	0000_0004h
6Ch	Processor Configuration 0 Register (CP2CFG0)	32	RO	0702_0704h
70h	Processor Configuration 1 Register (CP2CFG1)	32	RO	0B10_0000h
74h	Processor Configuration 2 Register (CP2CFG2)	32	RO	0000_0000h
78h	Processor Configuration 3 Register (CP2CFG3)	32	RO	0000_0017h
80h	Processor Type Register (CP3TYPE)	32	RO	4135_3333h
84h	Processor Number Register (CP3NUM)	32	RO	0000_0003h
88h	Processor Count Register (CP3REV)	32	RO	0000_0004h
8Ch	Processor Configuration 0 Register (CP3CFG0)	32	RO	0702_0704h
90h	Processor Configuration 1 Register (CP3CFG1)	32	RO	0B10_0000h
94h	Processor Configuration 2 Register (CP3CFG2)	32	RO	0000_0000h
98h	Processor Configuration 3 Register (CP3CFG3)	32	RO	0000_0017h
A0h	Processor Type Register (CP4TYPE)	32	RO	434D_3730h
A4h	Processor Number Register (CP4NUM)	32	RO	0000_0004h
A8h	Processor Count Register (CP4REV)	32	RO	0000_0011h
ACh	Processor Configuration 0 Register (CP4CFG0)	32	RO	0702_0704h
B0h	Processor Configuration 1 Register (CP4CFG1)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
B4h	Processor Configuration 2 Register (CP4CFG2)	32	RO	0800_0000h
B8h	Processor Configuration 3 Register (CP4CFG3)	32	RO	0000_000Bh
C0h	Processor Type Register (CP5TYPE)	32	RO	434D_3731h
C4h	Processor Number Register (CP5NUM)	32	RO	0000_0005h
C8h	Processor Count Register (CP5REV)	32	RO	0000_0011h
CCh	Processor Configuration 0 Register (CP5CFG0)	32	RO	0702_0704h
D0h	Processor Configuration 1 Register (CP5CFG1)	32	RO	0000_0000h
D4h	Processor Configuration 2 Register (CP5CFG2)	32	RO	0800_0000h
D8h	Processor Configuration 3 Register (CP5CFG3)	32	RO	0000_000Bh
E0h	Processor Type Register (CP6TYPE)	32	RO	434D_3732h
E4h	Processor Number Register (CP6NUM)	32	RO	0000_0006h
E8h	Processor Count Register (CP6REV)	32	RO	0000_0011h
ECh	Processor Configuration 0 Register (CP6CFG0)	32	RO	0702_0704h
F0h	Processor Configuration 1 Register (CP6CFG1)	32	RO	0000_0000h
F4h	Processor Configuration 2 Register (CP6CFG2)	32	RO	0800_0000h
F8h	Processor Configuration 3 Register (CP6CFG3)	32	RO	0000_000Bh
200h	Interrupt Router CP0 Interrupt0 Status Register (IRCP0ISR0)	32	W1C	0000_0000h
204h	Interrupt Router CP0 Interrupt0 Generation Register (IRCP0IGR0)	32	WORZ	0000_0000h
208h	Interrupt Router CP0 Interrupt1 Status Register (IRCP0ISR1)	32	W1C	0000_0000h
20Ch	Interrupt Router CP0 Interrupt1 Generation Register (IRCP0IGR1)	32	WORZ	0000_0000h
210h	Interrupt Router CP0 Interrupt2 Status Register (IRCP0ISR2)	32	W1C	0000_0000h
214h	Interrupt Router CP0 Interrupt2 Generation Register (IRCP0IGR2)	32	WORZ	0000_0000h
218h	Interrupt Router CP0 Interrupt3 Status Register (IRCP0ISR3)	32	W1C	0000_0000h
21Ch	Interrupt Router CPn Interruptx Generation Register (IRCP0IGR3)	32	WORZ	0000_0000h
220h	Interrupt Router CP1 Interrupt0 Status Register (IRCP1ISR0)	32	W1C	0000_0000h
224h	Interrupt Router CP1 Interrupt0 Generation Register (IRCP1IGR0)	32	WORZ	0000_0000h
228h	Interrupt Router CP1 Interrupt1 Status Register (IRCP1ISR1)	32	W1C	0000_0000h
22Ch	Interrupt Router CP1 Interrupt1 Generation Register (IRCP1IGR1)	32	WORZ	0000_0000h
230h	Interrupt Router CP1 Interrupt2 Status Register (IRCP1ISR2)	32	W1C	0000_0000h
234h	Interrupt Router CP1 Interrupt2 Generation Register (IRCP1IGR2)	32	WORZ	0000_0000h
238h	Interrupt Router CP1 Interrupt3 Status Register (IRCP1ISR3)	32	W1C	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
23Ch	Interrupt Router CPn Interruptx Generation Register (IRCP1IGR3)	32	WORZ	0000_0000h
240h	Interrupt Router CP2 Interrupt0 Status Register (IRCP2ISR0)	32	W1C	0000_0000h
244h	Interrupt Router CP2 Interrupt0 Generation Register (IRCP2IGR0)	32	WORZ	0000_0000h
248h	Interrupt Router CP2 Interrupt1 Status Register (IRCP2ISR1)	32	W1C	0000_0000h
24Ch	Interrupt Router CP2 Interrupt1 Generation Register (IRCP2IGR1)	32	WORZ	0000_0000h
250h	Interrupt Router CP2 Interrupt2 Status Register (IRCP2ISR2)	32	W1C	0000_0000h
254h	Interrupt Router CP2 Interrupt2 Generation Register (IRCP2IGR2)	32	WORZ	0000_0000h
258h	Interrupt Router CP2 Interrupt3 Status Register (IRCP2ISR3)	32	W1C	0000_0000h
25Ch	Interrupt Router CPn Interruptx Generation Register (IRCP2IGR3)	32	WORZ	0000_0000h
260h	Interrupt Router CP3 Interrupt0 Status Register (IRCP3ISR0)	32	W1C	0000_0000h
264h	Interrupt Router CP3 Interrupt0 Generation Register (IRCP3IGR0)	32	WORZ	0000_0000h
268h	Interrupt Router CP3 Interrupt1 Status Register (IRCP3ISR1)	32	W1C	0000_0000h
26Ch	Interrupt Router CP3 Interrupt1 Generation Register (IRCP3IGR1)	32	WORZ	0000_0000h
270h	Interrupt Router CP3 Interrupt2 Status Register (IRCP3ISR2)	32	W1C	0000_0000h
274h	Interrupt Router CP3 Interrupt2 Generation Register (IRCP3IGR2)	32	WORZ	0000_0000h
278h	Interrupt Router CP3 Interrupt3 Status Register (IRCP3ISR3)	32	W1C	0000_0000h
27Ch	Interrupt Router CPn Interruptx Generation Register (IRCP3IGR3)	32	WORZ	0000_0000h
280h	Interrupt Router CP4 Interrupt0 Status Register (IRCP4ISR0)	32	W1C	0000_0000h
284h	Interrupt Router CP4 Interrupt0 Generation Register (IRCP4IGR0)	32	WORZ	0000_0000h
288h	Interrupt Router CP4 Interrupt1 Status Register (IRCP4ISR1)	32	W1C	0000_0000h
28Ch	Interrupt Router CP4 Interrupt1 Generation Register (IRCP4IGR1)	32	WORZ	0000_0000h
290h	Interrupt Router CP4 Interrupt2 Status Register (IRCP4ISR2)	32	W1C	0000_0000h
294h	Interrupt Router CP4 Interrupt2 Generation Register (IRCP4IGR2)	32	WORZ	0000_0000h
298h	Interrupt Router CP4 Interrupt3 Status Register (IRCP4ISR3)	32	W1C	0000_0000h
29Ch	Interrupt Router CPn Interruptx Generation Register (IRCP4IGR3)	32	WORZ	0000_0000h
2A0h	Interrupt Router CP5 Interrupt0 Status Register (IRCP5ISR0)	32	W1C	0000_0000h
2A4h	Interrupt Router CP5 Interrupt0 Generation Register (IRCP5IGR0)	32	WORZ	0000_0000h
2A8h	Interrupt Router CP5 Interrupt1 Status Register (IRCP5ISR1)	32	W1C	0000_0000h
2ACh	Interrupt Router CP5 Interrupt1 Generation Register (IRCP5IGR1)	32	WORZ	0000_0000h
2B0h	Interrupt Router CP5 Interrupt2 Status Register (IRCP5ISR2)	32	W1C	0000_0000h
2B4h	Interrupt Router CP5 Interrupt2 Generation Register (IRCP5IGR2)	32	WORZ	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2B8h	Interrupt Router CP5 Interrupt3 Status Register (IRCP5ISR3)	32	W1C	0000_0000h
2BCh	Interrupt Router CPn Interruptx Generation Register (IRCP5IGR3)	32	WORZ	0000_0000h
2C0h	Interrupt Router CP6 Interrupt0 Status Register (IRCP6ISR0)	32	W1C	0000_0000h
2C4h	Interrupt Router CP6 Interrupt0 Generation Register (IRCP6IGR0)	32	WORZ	0000_0000h
2C8h	Interrupt Router CP6 Interrupt1 Status Register (IRCP6ISR1)	32	W1C	0000_0000h
2CCh	Interrupt Router CP6 Interrupt1 Generation Register (IRCP6IGR1)	32	WORZ	0000_0000h
2D0h	Interrupt Router CP6 Interrupt2 Status Register (IRCP6ISR2)	32	W1C	0000_0000h
2D4h	Interrupt Router CP6 Interrupt2 Generation Register (IRCP6IGR2)	32	WORZ	0000_0000h
2D8h	Interrupt Router CP6 Interrupt3 Status Register (IRCP6ISR3)	32	W1C	0000_0000h
2DCh	Interrupt Router CPn Interruptx Generation Register (IRCP6IGR3)	32	WORZ	0000_0000h
318h	Interrupt Router CP0 Interrupt4 Status Register (IRCP0ISR4)	32	W1C	0000_0000h
31Ch	Interrupt Router CPn Interruptx Generation Register (IRCP0IGR4)	32	WORZ	0000_0000h
338h	Interrupt Router CP1 Interrupt4 Status Register (IRCP1ISR4)	32	W1C	0000_0000h
33Ch	Interrupt Router CPn Interruptx Generation Register (IRCP1IGR4)	32	WORZ	0000_0000h
358h	Interrupt Router CP2 Interrupt4 Status Register (IRCP2ISR4)	32	W1C	0000_0000h
35Ch	Interrupt Router CPn Interruptx Generation Register (IRCP2IGR4)	32	WORZ	0000_0000h
378h	Interrupt Router CP3 Interrupt4 Status Register (IRCP3ISR4)	32	W1C	0000_0000h
37Ch	Interrupt Router CPn Interruptx Generation Register (IRCP3IGR4)	32	WORZ	0000_0000h
398h	Interrupt Router CP4 Interrupt4 Status Register (IRCP4ISR4)	32	W1C	0000_0000h
39Ch	Interrupt Router CPn Interruptx Generation Register (IRCP4IGR4)	32	WORZ	0000_0000h
3B8h	Interrupt Router CP5 Interrupt4 Status Register (IRCP5ISR4)	32	W1C	0000_0000h
3BCh	Interrupt Router CPn Interruptx Generation Register (IRCP5IGR4)	32	WORZ	0000_0000h
3D8h	Interrupt Router CP6 Interrupt4 Status Register (IRCP6ISR4)	32	W1C	0000_0000h
3DCh	Interrupt Router CPn Interruptx Generation Register (IRCP6IGR4)	32	WORZ	0000_0000h
400h	Interrupt Router Configuration Register (IRCPCFG)	32	RW	0000_0000h
800h	Interrupt Router Non-Maskable Interrupt Control Register (IRNMIC)	32	RW	0000_0000h
880h - A5Eh	Interrupt Router Shared Peripheral Routing Control Register (IRSPRC0 - IRSPRC239)	16	RW	000Fh

8.2.3.2 Processor X Type Register (CPXTYPE)

Offset

Register	Offset
CPXTYPE	0h

Function

This register provides a CPU-specific response indicating the personality of the core making the access. The 32-bit response includes four ASCII characters defining the CPU type (Cortex-A53 or Cortex-M7) along with a byte defining the logical revision number. This number follows Arm's rYpZ nomenclature.

A read from the Cortex-A53 or Cortex-M7 returns the appropriate processor information. Reads from any other bus master return all 0s. Attempted write accesses terminate with an error.

Access: Read by core masters Cortex-M7 and Cortex-A53 only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PERSONALITY															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PERSONALITY															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-0 PERSONALITY	Processor Personality This field defines the processor personality for CP*x*. **Table 23. Processor x personality**

Processor	Personality
CP*x* = Cortex-A53 cluster 0 core 0	41_35_33_30h (Cortex-A53_0)
CP*x* = Cortex-A53 cluster 0 core 1	41_35_33_31h (Cortex-A53_1)
CP*x* = Cortex-A53 cluster 1 core 0	41_35_33_34h (Cortex-A53_4)
CP*x* = Cortex-A53 cluster 1 core 1	41_35_33_35h (Cortex-A53_5)
CP*x* = Cortex-M7_0	43_4D_37_30h (Cortex-M7_0)
CP*x* = Cortex-M7_1	43_4D_37_31h (Cortex-M7_1)

Table 23. Processor x personality (continued)

Field	Function
	Table 23. Processor x personality (continued)

Processor	Personality
CP*x* = Cortex-M7_2	43_4D_37_32h (Cortex-M7_2)

8.2.3.3 Processor X Number Register (CPXNUM)

Offset

Register	Offset
CPXNUM	4h

Function

This register provides a CPU-specific response indicating the logical processor number of the core making the access.

A read from the Cortex-A53 or the Cortex-M7 cores returns the appropriate processor information. Reads from any other bus master return all 0s. Attempted write accesses terminate with an error.

Access: Read by core masters Cortex-M7 and Cortex-A53 only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CPN							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-8 —	Reserved
7-0 CPN	Processor Number This zero-filled word defines the logical processor number for CP*x*.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cc462e5-4821-48a8-b251-320d3ab624c1

Table continued from the previous page...

Field	Function
	Table 24. Processor x number

Processor description	Logical processor number
Cortex-A53 cluster 0 core 0	CPN = 0
Cortex-A53 cluster 0 core 1	CPN = 1
Cortex-A53 cluster 1 core 0	CPN = 2
Cortex-A53 cluster 1 core 1	CPN = 3
Cortex-M7 core 0	CPN = 4
Cortex-M7 core 1	CPN = 5
Cortex-M7 core 2	CPN = 6

8.2.3.4 Processor X Revision Register (CPXREV)

Offset

Register	Offset
CPXREV	8h

Function

This register provides a CPU-specific response indicating the logical revision number of the core. The number follows Arm's rYpZ nomenclature.

A read from the Cortex-A53 or the Cortex-M7 cores returns the appropriate processor information. Reads from any other bus master return all 0s. Attempted write accesses terminate with an error.

Access: Read by core masters Cortex-M7 and Cortex-A53 only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RYPZ							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEI Automotive Technology Co., Ltd.
ca462e5-4821-45a8-b251-390d3ab624c1

Fields

Field	Function
31-8 —	Reserved
7-0 RYPZ	Processor Revision This field defines the processor revision for CP*x*. **Table 25. Processor x revision** Processor description: Revision Cortex-A53: RYPZ = 04h corresponding to the r0p4 core release Cortex-M7: RYPZ = 11h corresponding to the r1p1 core release

8.2.3.5 Processor X Configuration 0 Register (CPXCFG0)

Offset

Register	Offset
CPXCFG0	Ch

Function

This register provides a CPU-specific response detailing configuration information. In this case, it is information on level 1 (L1) cache, if present.

A read from Cortex-A53 or Cortex-M7 returns the appropriate processor information. Reads from any other bus master return all 0s. Attempted write accesses terminate with an error.

Access: Read by core masters Cortex-M7 and Cortex-A53 only

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ICSZ								ICWY							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCSZ								DCWY							
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-482f-48a8-9251-390d3a6b24c1

Fields

Field	Function
31-24 ICSZ	L1 Instruction Cache Size This field provides an encoded value of the instruction cache size. The capacity of the memory is derived using the formula 2^(8+SZ) and expressed as bytes. Here, SZ is a non-zero value, and SZ = 0 indicates that the memory is not present. **Table 26. L1 instruction cache size x ICSZ**

Instruction cache size	ICSZ
No instruction cache	0h
4 KB instruction cache	4h
8 KB instruction cache	5h
16 KB instruction cache	6h
32 KB instruction cache	7h
64 KB instruction cache	8h
128 KB instruction cache	9h
256 KB instruction cache	Ah
512 KB instruction cache	Bh
Cortex-A53 cores	7h (32 KB)
Cortex-M7 cores	7h (32 KB)

23-16 ICWY	L1 Instruction Cache Ways This field provides the number of cache ways for the instruction cache. • For the Cortex-A53 cores in this chip, ICWY = 2h (2-way set-associative). • For the Cortex-M7 cores in this chip, ICWY = 2h (2-way set-associative).
15-8 DCSZ	L1 Data Cache Size This field provides an encoded value of the data cache size. The capacity of the memory is derived using the formula 2^(8+SZ) and expressed as bytes. Here, SZ is a non-zero value, and SZ = 0 indicates that the memory is not present. **Table 27. Data cache size x DCSZ**

Data cache size	DCSZ
No data cache	0h
4 KB data cache	4h
8 KB data cache	5h
16 KB data cache	6h
32 KB data cache	7h
64 KB data cache	8h

Table 27. Data cache size x DCSZ (continued)

Field	Function
	Table 27. Data cache size x DCSZ (continued) Data cache size / DCSZ 128 KB data cache / 9h 256 KB data cache / Ah 512 KB data cache / Bh Cortex-A53 cores / 7h (32 KB) Cortex-M7 cores / 7h (32 KB)
7-0 DCWY	L1 Data Cache Ways This field provides the number of cache ways for the data cache. • For the Cortex-A53 cores in this chip, DCWY = 4h (4-way set-associative). • For the Cortex-M7 cores in this chip, DCWY = 4h (4-way set-associative).

8.2.3.6 Processor X Configuration 1 Register (CPXCFG1)

Offset

Register	Offset
CPXCFG1	10h

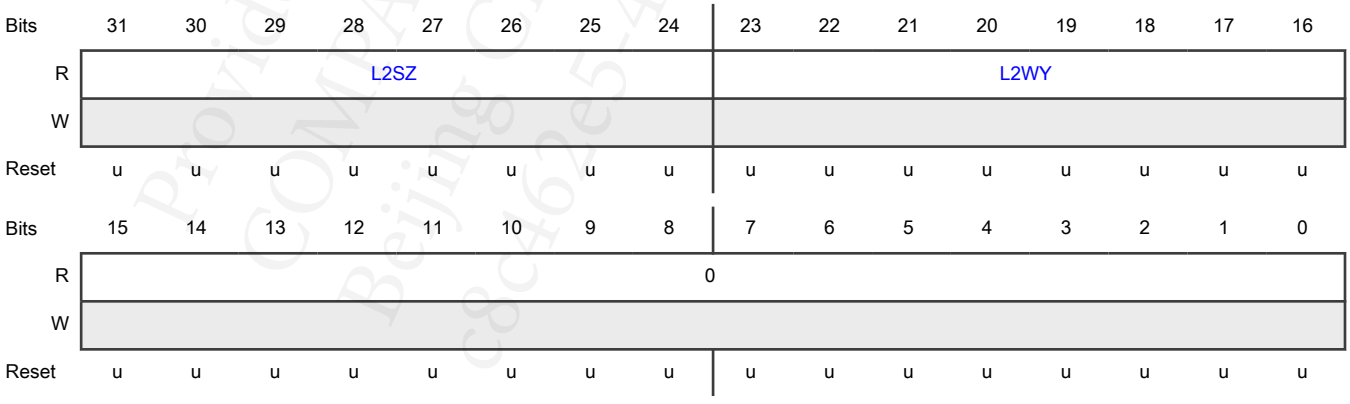
Function

This register provides a CPU-specific response detailing configuration information. In this case, it is information on level 2 (L2) cache, if present.

A read from the Cortex-A53 or Cortex-M7 cores returns the appropriate processor information. Reads from any other bus master return all 0s. Attempted write accesses terminate with an error.

Access: Read by core masters Cortex-M7 and Cortex-A53 only

Diagram



Fields

Field	Function
31-24 L2SZ	L2 Cache Size This field provides an encoded value of the L2 cache size. The capacity of the memory is derived using the formula 2^(8+SZ) and expressed as bytes. Here, SZ is a non-zero value, and SZ = 0 indicates that the memory is not present. **Table 28. L2 cache size x L2SZ** L2 cache size / L2SZ: No L2 cache — 0h 4 KB L2 cache — 4h 8 KB L2 cache — 5h 16 KB L2 cache — 6h 32 KB L2 cache — 7h 64 KB L2 cache — 8h 128 KB L2 cache — 9h 256 KB L2 cache — Ah 512 KB L2 cache — Bh Cortex-A53 cores — Ah (256 KB) Cortex-M7 cores — 0h (not present)
23-16 L2WY	L2 Cache Ways This field provides the number of cache ways for the L2 cache. • For the Cortex-A53 cores in this chip, L2WY = 10h (16-way set-associative). • For the Cortex-M7 cores in this chip, L2WY = 0h (not present).
15-0 —	Reserved

8.2.3.7 Processor X Configuration 2 Register (CPXCFG2)

Offset

Register	Offset
CPXCFG2	14h

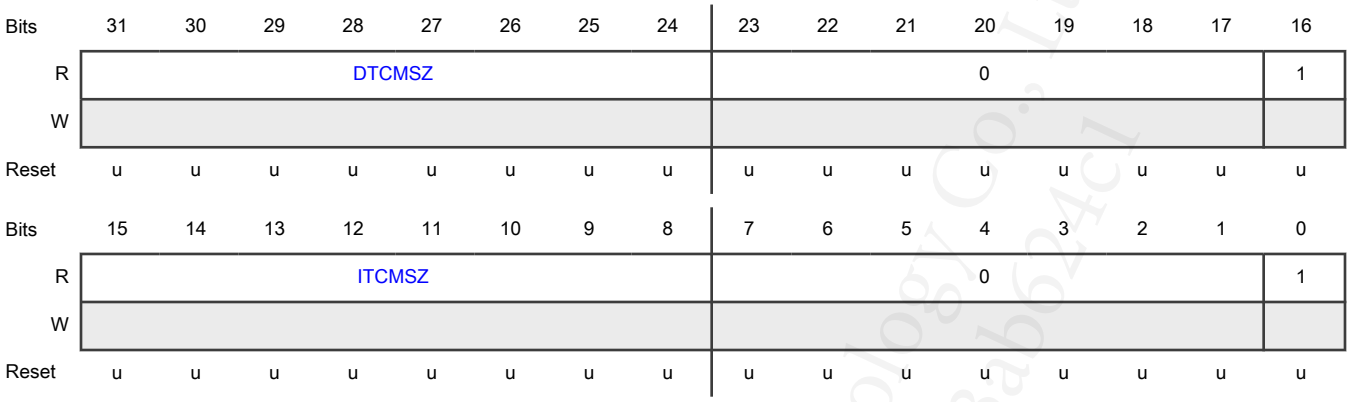
Function

This register provides a CPU-specific response detailing configuration information. In this case, it is information on tightly coupled local memories, if present.

A read from the Cortex-A53 or the Cortex-M7 cores returns the appropriate processor information. Reads from any other bus master return all 0s. Attempted write accesses terminate with an error.

Access: Read by core masters Cortex-M7 and Cortex-A53 only

Diagram



Fields

Field	Function
31-24 DTCMSZ	Tightly Coupled Data Memory Size This field provides an encoded value of the tightly coupled local data memory size. The capacity of the memory is derived using the formula 2^(8+SZ) and expressed as bytes. Here, SZ is a non-zero value, and SZ = 0 indicates that the memory is not present.
23-17 —	Reserved

Table 29. DTCML x DTCMSZ

DTCML	DTCMSZ
No DTCML	0h
4 KB DTCML	4h
8 KB DTCML	5h
16 KB DTCML	6h
32 KB DTCML	7h
64 KB DTCML	8h
128 KB DTCML	9h
256 KB DTCML	Ah
512 KB DTCML	Bh
Cortex-A53 cores	0h (not present)
Cortex-M7 cores	8h (64 KB)

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 —	Reserved
15-8 ITCMSZ	Instruction Tightly Coupled Memory Size This field provides an encoded value of the tightly coupled local instruction memory size. The capacity of the memory is derived using the formula 2^(8+SZ) and expressed as bytes. Here, SZ is a non-zero value, and SZ = 0 indicates that the memory is not present. **Table 30. ITCMU x ITCMSZ**
7-1 —	Reserved
0 —	Reserved

ITCMU	ITCMSZ
No ITCM	0h
4 KB ITCMU	4h
8 KB ITCMU	5h
16 KB ITCMU	6h
32 KB ITCMU	7h
64 KB ITCMU	8h
128 KB ITCMU	9h
256 KB ITCMU	10h
512 KB ITCMU	11h
Cortex-A53 cores	0h (not present)
Cortex-M7 cores	0h (not present)

8.2.3.8 Processor X Configuration 3 Register (CPXCFG3)

Offset

Register	Offset
CPXCFG3	18h

Function

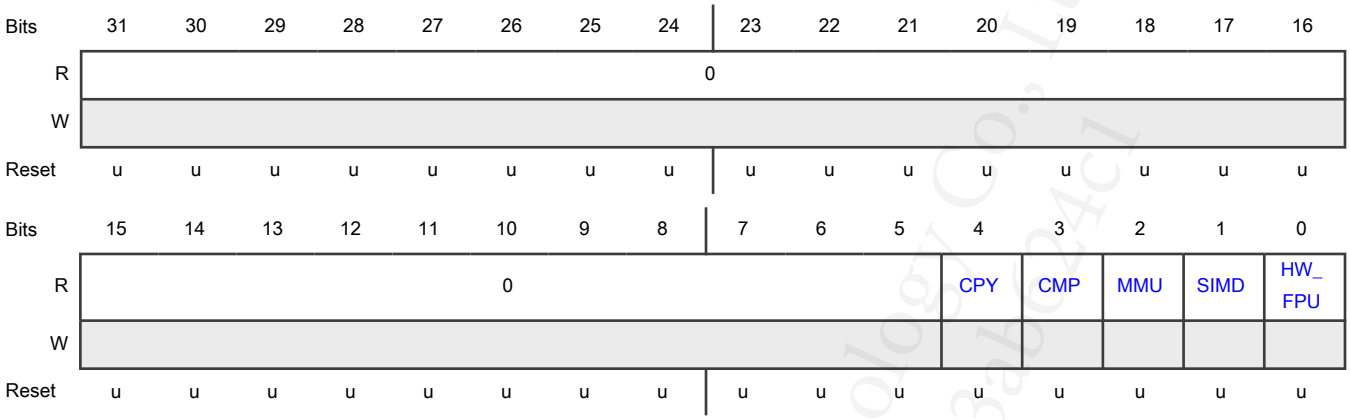
This register provides a CPU-specific response detailing configuration information. In this case, it is information on processor options.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-1821-48a6-9251-390d3ab624c

A privileged read from Cortex-A53 or Cortex-M7 returns the appropriate processor information. Reads from any other bus master return all 0s. Attempted write accesses terminate with an error.

Access: Read by core masters Cortex-M7 and Cortex-A53 only

Diagram



Fields

Field	Function
31-5 —	Reserved
4 CPY	Cryptography This field indicates if cryptography extensions are supported in the core. • If cryptography support is not included, CPY = 0h. • If cryptography support is included, CPY = 1h. • For the Cortex-A53 cores in this chip, CPY = 1h (supported). • For the Cortex-M7 cores in this chip, CPY = 0h (not supported).
3 CMP	Core Memory Protection Unit This field indicates if the core memory protection hardware is included in this core. • If core memory protection is not included, CMP = 0h. • If core memory protection is included, CMP = 1h. • For the Cortex-A53 cores in this chip, CMP = 0h (not supported). • For the Cortex-M7 cores in this chip, CMP = 1h (supported).
2 MMU	Memory Management Unit This field indicates if the virtual management capabilities are supported in this core. • If MMU support is not included, MMU = 0h. • If MMU support is included, MMU = 1h.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• For the Cortex-A53 cores in this chip, MMU = 1h (supported). • For the Cortex-M7 cores in this chip, MMU = 0h (not supported).
1 SIMD	SIMD/Neon Instruction Support This field indicates if the instruction set extensions supporting SIMD and/or Neon capabilities are supported in the processor. • If SIMD/Neon support is not included, SIMD = 0h. • If SIMD/Neon support is included, SIMD = 1h. • For the Cortex-A53 cores in this chip, SIMD = 1h (supported). • For the Cortex-M7 cores in this chip, SIMD = 1h (supported).
0 HW_FPU	Floating Point Unit This field indicates if hardware support for floating point capabilities is supported in the processor. • If HW_FPU support is not included, HW_FPU = 0h. • If HW_FPU support is included, HW_FPU = 1h. • For the Cortex-A53 cores in this chip, HW_FPU = 1h (supported). • For the Cortex-M7 cores in this chip, HW_FPU = 1h (supported).

8.2.3.9 Processor Type Register (CP0TYPE - CP6TYPE)

Offset

Register	Offset
CP0TYPE	20h
CP1TYPE	40h
CP2TYPE	60h
CP3TYPE	80h
CP4TYPE	A0h
CP5TYPE	C0h
CP6TYPE	E0h

Function

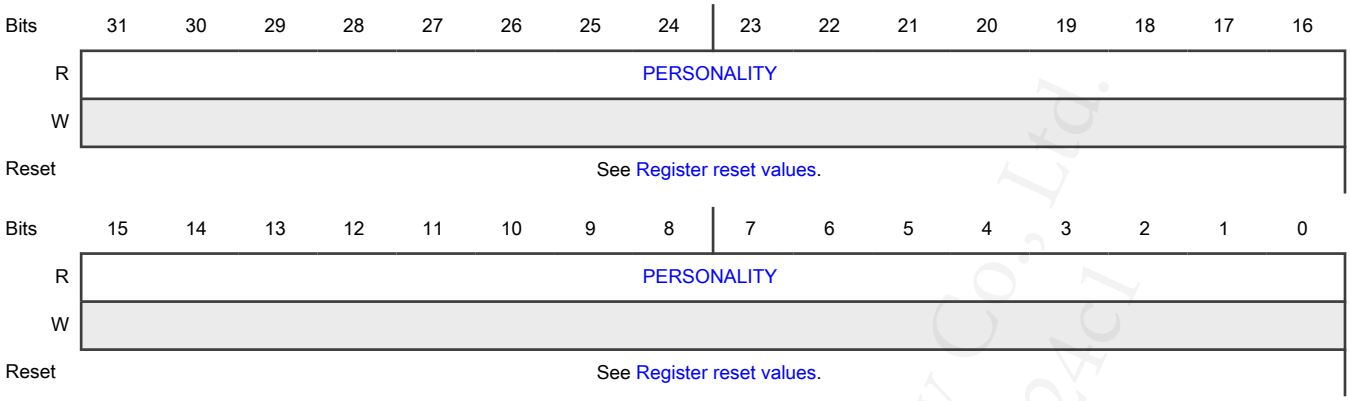
This register defines the configuration information for CP0-6 . It has the same field definitions and functionality as provided in Processor X Type Register (CPXTYPE).

A read from any bus master returns the appropriate processor information and attempted write accesses terminate with an error.

NOTE

The reset value depends on the processor.

Diagram



Register reset values

Register	Reset value
CP0TYPE	4135_3330h
CP1TYPE	4135_3331h
CP2TYPE	4135_3332h
CP3TYPE	4135_3333h
CP4TYPE	434D_3730h
CP5TYPE	434D_3731h
CP6TYPE	434D_3732h

Fields

Field	Function
31-0 PERSONALITY	Processor Personality This field defines the processor personality for CP*x*. **Table 31. Processor x personality**

Processor	Personality
CP*x* = Cortex-A53 cluster 0 core 0	41_35_33_30h (Cortex-A53_0)
CP*x* = Cortex-A53 cluster 0 core 1	41_35_33_31h (Cortex-A53_1)
CP*x* = Cortex-A53 cluster 1 core 0	41_35_33_34h (Cortex-A53_4)
CP*x* = Cortex-A53 cluster 1 core 1	41_35_33_35h (Cortex-A53_5)
CP*x* = Cortex-M7_0	43_4D_37_30h (Cortex-M7_0)
CP*x* = Cortex-M7_1	43_4D_37_31h (Cortex-M7_1)
CP*x* = Cortex-M7_2	43_4D_37_32h (Cortex-M7_2)

Preliminary NDA only
COMPANY PROPRIETARY
Copyright (c) Automotive Technology Co., Ltd.
8c4625-4821-48a8-b251-390d8ab624c1

8.2.3.10 Processor Number Register (CP0NUM - CP6NUM)

Offset

Register	Offset
CP0NUM	24h
CP1NUM	44h
CP2NUM	64h
CP3NUM	84h
CP4NUM	A4h
CP5NUM	C4h
CP6NUM	E4h

Function

This register defines the configuration information for CP0-6 . It has the same field definitions and functionality as provided in Processor X Number Register (CPXNUM).

A privileged read from any bus master returns the appropriate processor information. Attempted write accesses terminate with an error.

NOTE

The reset value depends on the processor.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CPN							
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
CP0NUM	0000_0000h
CP1NUM	0000_0001h
CP2NUM	0000_0002h

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4824-8a8-b251-3903ab624c1

Table continued from the previous page...

Register	Reset value
CP3NUM	0000_0003h
CP4NUM	0000_0004h
CP5NUM	0000_0005h
CP6NUM	0000_0006h

Fields

Field	Function
31-8 —	Reserved
7-0 CPN	Processor Number This zero-filled word defines the logical processor number for CP*x*. **Table 32. Processor x number** Processor description / Logical processor number: Cortex-A53 cluster 0 core 0 — CPN = 0 Cortex-A53 cluster 0 core 1 — CPN = 1 Cortex-A53 cluster 1 core 0 — CPN = 2 Cortex-A53 cluster 1 core 1 — CPN = 3 Cortex-M7 core 0 — CPN = 4 Cortex-M7 core 1 — CPN = 5 Cortex-M7 core 2 — CPN = 6

8.2.3.11 Processor Count Register (CP0REV - CP6REV)

Offset

Register	Offset
CP0REV	28h
CP1REV	48h
CP2REV	68h
CP3REV	88h
CP4REV	A8h

Table continues on the next page...

Table continued from the previous page...

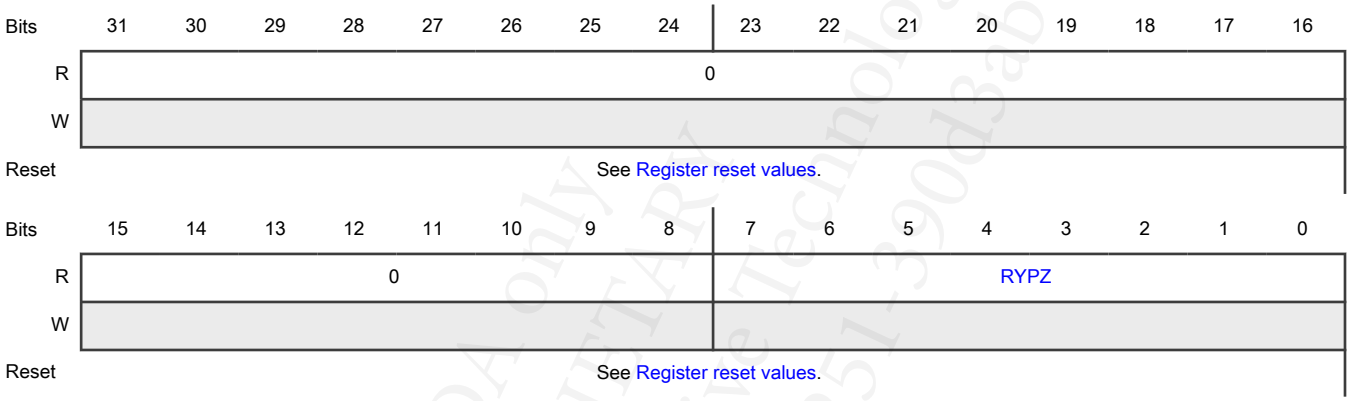
Register	Offset
CP5REV	C8h
CP6REV	E8h

Function

This register defines the configuration information for CP0-6 . It has the same field definitions and functionality as provided in Processor X Revision Register (CPXREV).

A read from any bus master returns the appropriate processor information. Attempted write accesses terminate with an error.

Diagram



Register reset values

Register	Reset value
CP0REV–CP3REV	0000_0004h
CP4REV–CP6REV	0000_0011h

Fields

Field	Function
31-8 —	Reserved
7-0 RYPZ	Processor Revision This field defines the processor revision for CPx: **Table 33. Processor x Revision** Processor description: Cortex-A53 — Revision: RYPZ = 4h corresponding to the r0p4 core release Processor description: Cortex-M7 — Revision: RYPZ = 11h corresponding to the r1p1 core release

Table continued from the previous page...

Field	Function
	Processor description | Revision

8.2.3.12 Processor Configuration 0 Register (CP0CFG0 - CP6CFG0)

Offset

Register	Offset
CP0CFG0	2Ch
CP1CFG0	4Ch
CP2CFG0	6Ch
CP3CFG0	8Ch
CP4CFG0	ACh
CP5CFG0	CCh
CP6CFG0	ECh

Function

This register defines the configuration 0 information for CP0-6 . It has the same field definitions and functionality as provided in Processor X Configuration 0 Register (CPXCFG0).

A read from any bus master returns the appropriate processor information. Attempted write accesses terminate with an error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ICSZ								ICWY							
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCSZ								DCWY							
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-8a8-b251-390d3ab624c1

Fields

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>31-24
ICSZ</td><td>L1 Instruction Cache Size
This field provides an encoded value of the instruction cache size. The capacity of the memory is derived using the formula 2^(8+SZ) and expressed as bytes. Here, SZ is a non-zero value, and SZ = 0 indicates that the memory is not present.
Table 34. L1 instruction cache size x ICSZ
<table>
<tr><th>Instruction cache size</th><th>ICSZ</th></tr>
<tr><td>No instruction cache</td><td>0h</td></tr>
<tr><td>4 KB instruction cache</td><td>4h</td></tr>
<tr><td>8 KB instruction cache</td><td>5h</td></tr>
<tr><td>16 KB instruction cache</td><td>6h</td></tr>
<tr><td>32 KB instruction cache</td><td>7h</td></tr>
<tr><td>64 KB instruction cache</td><td>8h</td></tr>
<tr><td>128 KB instruction cache</td><td>9h</td></tr>
<tr><td>256 KB instruction cache</td><td>Ah</td></tr>
<tr><td>512 KB instruction cache</td><td>Bh</td></tr>
<tr><td>Cortex-A53 cores</td><td>7h (32 KB)</td></tr>
<tr><td>Cortex-M7 cores</td><td>7h (32 KB)</td></tr>
</table></td></tr>
<tr><td>23-16
ICWY</td><td>L1 Instruction Cache Ways
This field provides the number of cache ways for the instruction cache.
• For the Cortex-A53 core in this chip, ICWY = 2h (2-way set-associative).
• For the Cortex-M7 cores in this chip, ICWY = 2h (2-way set-associative).</td></tr>
</table>

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b621-390d3ab624c1

Table continued from the previous page...

Field	Function
15-8 DCSZ	L1 Data Cache Size This field provides an encoded value of the data cache size. The capacity of the memory is derived using the formula 2^(8+SZ) and expressed as bytes. Here, SZ is a non-zero value, and SZ = 0 indicates that the memory is not present. **Table 35. Data cache size x DCSZ**

Data cache size	DCSZ
No data cache	0h
4 KB data cache	4h
8 KB data cache	5h
16 KB data cache	6h
32 KB data cache	7h
64 KB data cache	8h
128 KB data cache	9h
256 KB data cache	Ah
512 KB data cache	Bh
Cortex-A53 cores	7h (32 KBs)
Cortex-M7 cores	7h (32 KBs)

7-0 DCWY	L1 Data Cache Ways This field provides the number of cache ways for the data cache. • For the Cortex-A53 core in this chip, DCWY = 4h (4-way set-associative). • For the Cortex-M7 cores in this chip, DCWY = 4h (4-way set-associative).

8.2.3.13 Processor Configuration 1 Register (CP0CFG1 - CP6CFG1)

Offset

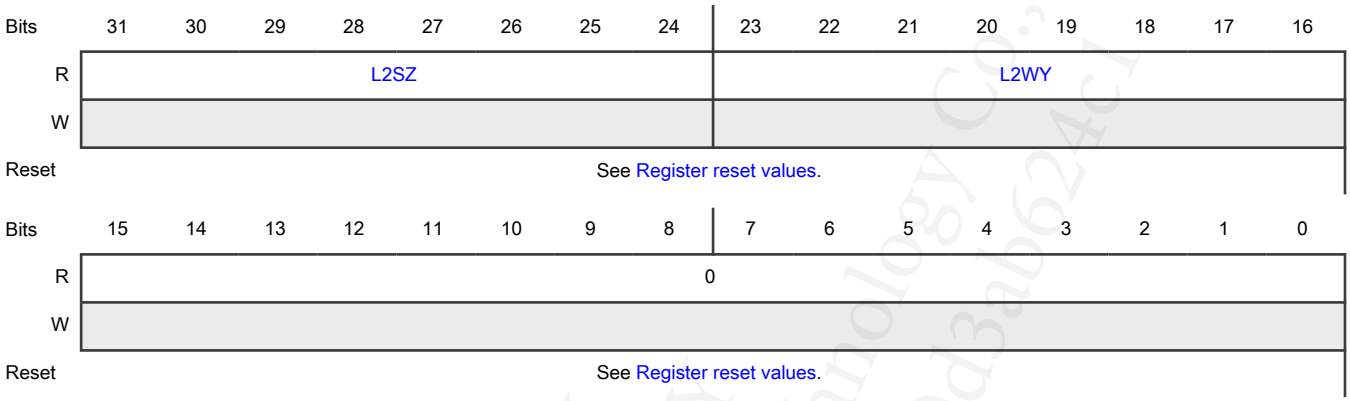
Register	Offset
CP0CFG1	30h
CP1CFG1	50h
CP2CFG1	70h
CP3CFG1	90h
CP4CFG1	B0h
CP5CFG1	D0h
CP6CFG1	F0h

Function

This register defines the configuration 1 information for CP0-6 . It has the same field definitions and functionality as provided in Processor X Configuration 1 Register (CPXCFG1).

A privileged read from any bus master returns the appropriate processor information. Attempted write accesses terminate with an error.

Diagram



Register reset values

Register	Reset value
CP0CFG1–CP3CFG1	0B10_0000h
CP4CFG1–CP6CFG1	0000_0000h

Fields

Field	Function
31-24 L2SZ	L2 Cache Size This field provides an encoded value of the L2 cache size. The capacity of the memory is derived using the formula 2^(8+SZ) and expressed as bytes. Here, SZ is a non-zero value, and SZ = 0 indicates that the memory is not present.

Table 36. L2 cache size x L2SZ

L2 cache size	L2SZ
No L2 cache	0h
4 KB L2 cache	4h
8 KB L2 cache	5h
16 KB L2 cache	6h
32 KB L2 cache	7h
64 KB L2 cache	8h
128 KB L2 cache	9h

Table 36. L2 cache size x L2SZ (continued)

Field	Function
	Table 36. L2 cache size x L2SZ (continued) L2 cache size: L2SZ 256 KB L2 cache: Ah 512 KB L2 cache: Bh Cortex-A53 cores: Ah (256 KB) Cortex-M7 cores: 0h (not present)
23-16 L2WY	L2 Cache Ways This field provides the number of cache ways for the L2 cache. • For the Cortex-A53 cores in this chip, L2WY = 10h (16-way set-associative). • For the Cortex-M7 cores in this chip, L2WY = 0h (not present).
15-0 —	Reserved

8.2.3.14 Processor Configuration 2 Register (CP0CFG2 - CP6CFG2)

Offset

Register	Offset
CP0CFG2	34h
CP1CFG2	54h
CP2CFG2	74h
CP3CFG2	94h
CP4CFG2	B4h
CP5CFG2	D4h
CP6CFG2	F4h

Function

This register defines the configuration 2 information for CP0-6 . It has the same field definitions and functionality as provided in Processor X Configuration 2 Register (CPXCFG2).

A privileged read from any bus master returns the appropriate processor information. Attempted write accesses terminate with an error.

Freescale Under NDA Only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4025-4821-4848-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TMLSZ								0							1
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TMUSZ								0							1
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
CP0CFG2–CP3CFG2	0000_0000h
CP4CFG2–CP6CFG2	0800_0000h

Fields

Field	Function
31-24 TMLSZ	Tightly Coupled Memory Lower Size This field provides an encoded value of the tightly coupled local memory lower size. The capacity of the memory is derived using the formula 2^(8+SZ) and expressed as bytes. Here, SZ is a non-zero value, and SZ = 0 indicates that the memory is not present.

Table 37. TCML x TMLSZ

TCML	TMLSZ
No TCML	0h
4 KB TCML	4h
8 KB TCML	5h
16 KB TCML	6h
32 KB TCML	7h
64 KB TCML	8h
128 KB TCML	9h
256 KB TCML	Ah
512 KB TCML	Bh
Cortex-A53 cores	0h (not present)
Cortex-M7 cores	7h (32 KB)

Table continued from the previous page...

Field	Function
23-17 —	Reserved
16 —	Reserved
15-8 TMUSZ	Tightly Coupled Memory Upper Size This field provides an encoded value of the tightly coupled local memory upper size. The capacity of the memory is derived using the formula 2^(8+SZ) and expressed as bytes. Here, SZ is a non-zero value, and SZ = 0 indicates that the memory is not present. **Table 38. TCMU x TMUSZ** TCMU / TMUSZ No TCMU / 0h 4 KB TCMU / 4h 8 KB TCMU / 5h 16 KB TCMU / 6h 32 KB TCMU / 7h 64 KB TCMU / 8h 128 KB TCMU / 9h 256 KB TCMU / Ah 512 KB TCMU / Bh Cortex-A53 cores / 0h (not present) Cortex-M7 cores / 7h (32 KB)
7-1 —	Reserved
0 —	Reserved

8.2.3.15 Processor Configuration 3 Register (CP0CFG3 - CP6CFG3)

Offset

Register	Offset
CP0CFG3	38h

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca4625-4821-48e6-9251-390d3ab624c1

Table continued from the previous page...

Register	Offset
CP1CFG3	58h
CP2CFG3	78h
CP3CFG3	98h
CP4CFG3	B8h
CP5CFG3	D8h
CP6CFG3	F8h

Function

This register defines the configuration 3 information for CP0-6 . It has the same field definitions and functionality as provided in Processor X Configuration 3 Register (CPXCFG3).

A privileged read from any bus master returns the appropriate processor information. Attempted write accesses terminate with an error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											CPY	CMP	MMU	SIMD	HW_FPU
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
CP0CFG3–CP3CFG3	0000_0017h
CP4CFG3–CP6CFG3	0000_000Bh

Fields

Field	Function
31-5 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4921-49a8-b251-290da3a624c1

Table continued from the previous page...

Field	Function
4 CPY	Cryptography This field indicates if the core supports cryptography extensions. • If cryptography support is not included, CPY = 0h. • If cryptography support is included, CPY = 1h. • For the Cortex-A53 cores in this chip, CPY = 1h (supported). • For the Cortex-M7 cores in this chip, CPY = 0h (not supported).
3 CMP	Core Memory Protection Unit This field indicates if the core inlcudes memory protection hardware. • If core memory protection is not included, CMP = 0h. • If core memory protection is included, CMP = 1h. • For the Cortex-A53 cores in this chip, CMP = 0h (not supported). • For the Cortex-M7 cores in this chip, CMP = 1h (supported).
2 MMU	Memory Management Unit This field indicates if the core includes virtual management capabilities. • If MMU support is not included, MMU = 0h. • If MMU support is included, MMU = 1h. • For the Cortex-A53 cores in this chip, MMU = 1h (supported). • For the Cortex-M7 cores in this chip, MMU = 0h (not supported).
1 SIMD	SIMD/Neon instruction support This field indicates if the processor includes instruction set extensions supporting SIMD and/or Neon capabilities. • If SIMD/Neon support is not included, SIMD = 0h. • If SIMD/Neon support is included, SIMD = 1h. • For the Cortex-A53 cores in this chip, SIMD = 1h (supported). • For the Cortex-M7 cores in this chip, SIMD = 1h (supported).
0 HW_FPU	Floating Point Unit This field indicates if the processor includes hardware support for floating point capabilities. • If HW_FPU support is not included, HW_FPU = 0h. • If HW_FPU support is included, HW_FPU = 1h. • For the Cortex-A53 cores in this chip, HW_FPU = 1h (supported). • For the Cortex-M7 cores in this chip, HW_FPU = 1h (supported).

Preliminary Information COMPANY CONFIDENTIAL Beijing CHANGAN Automotive Technology Co., Ltd. c8d5e5_4a2f4818a2951390d3ab624c1

8.2.3.16 Interrupt Router CPa Interruptb Status Register (IRCP0ISR0 - IRCP6ISR2)

Offset

For a = 0 to 6; b = 0 to 2:

Register	Offset
IRCPaISRb	200h + (a × 20h) + (b × 8h)

Function

The IRCP*n*ISR{0,1,2} set of registers provides an interrupt bitmap, in which each bit defines the state of a unique MSI based on the initiating core. An MSI interrupt is cleared in an interrupt handler by writing a 1 to the appropriate field in the IRCP*n*ISR*x* register.

In this discussion, CP*m* represents the initiating core and CP*n* represents the target core for a core-to-core interrupt.

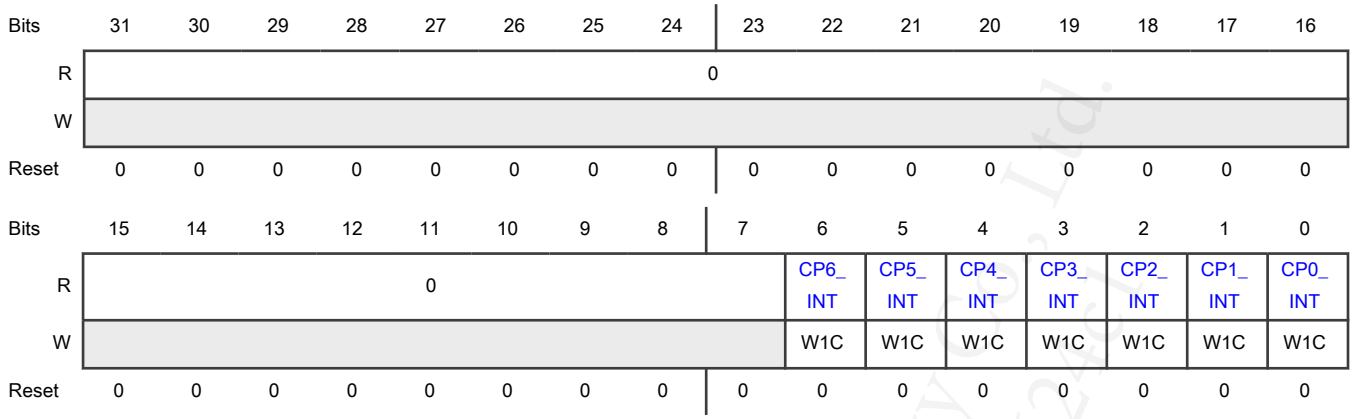
Read access

- Reads to IRCP*n*ISR{0,1,2} are only accessible in Privileged mode using 32-bit (word) accesses.
- Privileged, 32-bit read accesses from non-core (and non-debugger) bus masters are treated as Read-as-Zero (RAZ).
- Attempted accesses in User mode or the ones using a size other than 32 bits are not permitted. They terminate with an error.
- When CP*n* requests to read IRCP*n*ISR{0,1,2}, MSCM returns the entire content of IRCP*n*ISR{0,1,2}.
- When a trusted core, as indicated by IRCPCFG, requests to read IRCP*n*ISR{0,1,2}, MSCM returns the entire content of IRCP*n*ISR{0,1,2}.
- When the debugger requests to read IRCP*n*ISR{0,1,2}, MSCM returns the entire content of IRCP*n*ISR{0,1,2}.
- When CP*m* requests to read IRCP*n*ISR{0,1,2}, MSCM only returns the value of the corresponding status, CP*m*_INT, while not exposing all other pending interrupts initiated by the other cores.
- When CP*m* requests to read IRCP*n*ISR{0,1,2}, MSCM returns the value of the corresponding status, CP*m*_INT, in bit position 0, reflecting how CP*m* set the MSI when it wrote to IRCP*n*IGR{0,1,2}. All other fields on the returned read value are zero-filled.

Write access

- Writes to IRCP*n*ISR{0,1,2} are only accessible in Privileged mode using 32-bit (word) accesses.
- Attempted accesses in User mode or the ones using a size other than 32 bits are not permitted. They terminate with an error.
- Writes to IRCP*n*ISR{0,1,2} follow the Write 1 to Clear (W1C) protocol, whereby writing a 1 clears the corresponding field, and writing 0 is ignored.
- PCIe_0 access is restricted to registers IRCPnIGR3 and IRCPnISR3, and PCIe_1 access is restricted to registers IRCPnIGR4 and IRCPnISR4.
- The target core, CP*n*, has full access to write to all the fields of IRCP*n*ISR*x*.
- A trusted core, as indicated by IRCPCFG, has full access to write to all the fields of IRCP*n*ISR{0,1,2}.
- When CP*m* is different from CP*n*, the W1C action by CP*m* only clears IRCP*n*ISR{0,1,2}[CP*m*_INT].
- The CP*m* field must present W1C in bit position 0 to clear its corresponding interrupt. Write data bits 1-31 that CP*m* presents are ignored.
- The privileged write accesses from non-core (and non-debugger) bus masters are treated as Writes Ignored (WI).
- The privileged write accesses from the debugger are treated as WI.

Diagram



Fields

Field	Function
31-7 —	Reserved
6 CP6_INT	CP6-to-CPn Interrupt This field generates a directed interrupt initiated by core 6 targeting core *n*, if the appropriate interrupt routing bit is enabled. The interrupt is negated when the target core, a trusted core, or core 6 writes a 1 to clear the field. 0b - CP6 does not assert any interrupt to CPn. 1b - Interrupt to CPn, initiated by CP6, is asserted.
5 CP5_INT	CP5-to-CPn Interrupt This field generates a directed interrupt initiated by core 5 targeting core *n*, if the appropriate interrupt routing bit is enabled. The interrupt is negated when the target core, a trusted core, or core 5 writes a 1 to clear the field. 0b - CP5 does not assert any interrupt to CPn. 1b - Interrupt to CPn, initiated by CP5, is asserted.
4 CP4_INT	CP4-to-CPn Interrupt This field generates a directed interrupt initiated by core 4 targeting core *n*, if the appropriate interrupt routing bit is enabled. The interrupt is negated when the target core, a trusted core, or core 4 writes a 1 to clear the field. 0b - CP4 does not assert any interrupt to CPn. 1b - Interrupt to CPn, initiated by CP4, is asserted.
3 CP3_INT	CP3-to-CPn Interrupt This field generates a directed interrupt initiated by core 3 targeting core *n*, if the appropriate interrupt routing bit is enabled. The interrupt is negated when the target core, a trusted core, or core 3 writes a 1 to clear the field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - CP3 does not assert any interrupt to CPn. 1b - Interrupt to CPn, initiated by CP3, is asserted.
2 CP2_INT	CP2-to-CPn Interrupt This field generates a directed interrupt initiated by core 2 targeting core *n*, if the appropriate interrupt routing bit is enabled. The interrupt is negated when the target core, a trusted core, or core 2 writes a 1 to clear the field. 0b - CP2 does not assert any interrupt to CPn. 1b - Interrupt to CPn, initiated by CP2, is asserted.
1 CP1_INT	CP1-to-CPn Interrupt This field generates a directed interrupt initiated by core 1 targeting core *n*, if the appropriate interrupt routing field is enabled. The interrupt is negated when the target core, a trusted core, or core 1 writes a 1 to clear the field. 0b - CP1 does not assert any interrupt to CPn. 1b - Interrupt to CPn, initiated by CP1, is asserted.
0 CP0_INT	CP0-to-CPn Interrupt This field generates a directed interrupt initiated by core 0 targeting core *n*, if the appropriate interrupt routing bit is enabled. The interrupt is negated when the target core, a trusted core, or core 0 writes a 1 to clear the field. 0b - CP0 does not assert any interrupt to CPn. 1b - Interrupt to CPn, initiated by CP0, is asserted.

8.2.3.17 Interrupt Router CPa Interruptb Generation Register (IRCP0IGR0 - IRCP6IGR2)

Offset

For a = 0 to 6; b = 0 to 2:

Register	Offset
IRCPaIGRb	204h + (a × 20h) + (b × 8h)

Function

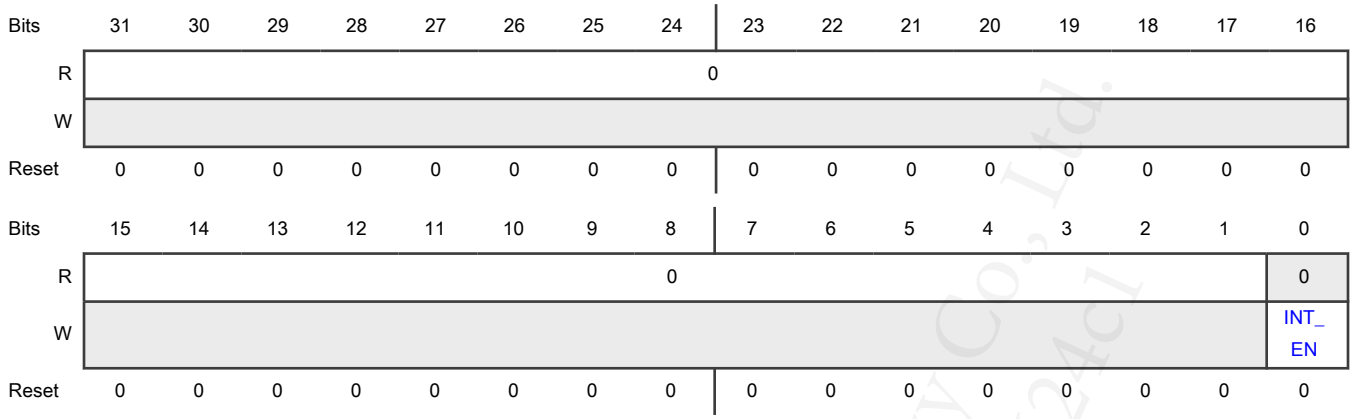
The IRCP*n*IGR{0,1,2} set of registers provides a mechanism for cores to initiate an MSI to another core in the system.

Privileged, 32-bit accesses from the:

- Cortex-A53 or Cortex-M7 cores are treated as RAZ/W
- Debugger are treated as RAZ/WI
- Non-core (and non-debugger) bus masters are treated as RAZ/WI

Attempted accesses in User mode or the ones using a size other than 32 bits are not permitted. They terminate with an error.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 INT_EN	Interrupt Enable A write to this field by CP*m* initiates a core-to-core interrupt targeting CP*n*.

8.2.3.18 Interrupt Router CPa Interruptb Status Register (IRCP0ISR3 - IRCP6ISR4)

Offset

For a = 0 to 6; b = 3 to 4:

Register	Offset
IRCPaISRb	-E8h + (a × 20h) + (b × 100h)

Function

CP*n* represents the target core for a PCIe-to-core interrupt.

The fields of this register generate a directed interrupt initiated by PCIe, targeting core *n*, if the appropriate interrupt routing bit is enabled. The interrupt is negated when the target core or a trusted core writes 1 to clear the field.

NOTE

IRCPnISR3 and IRCPnIGR3 correspond to the PCIe_0 interrupt; IRCPnISR4 and IRCPnIGR4 correspond to the PCIe_1 interrupt.

Read access

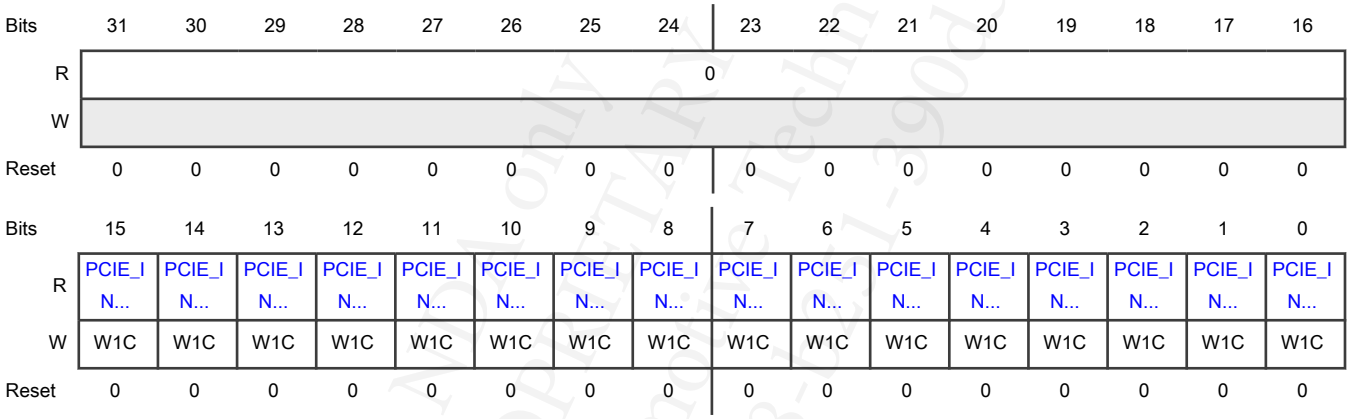
- Reads to IRCP*n*ISR*p* are only accessible in Privileged mode using 32-bit (word) accesses.
- Privileged, 32-bit read accesses from non-core (and non-debugger) bus masters are treated as Read-as-Zero (RAZ).
- Privileged, 32-bit read accesses from any other core, except CP*n*, are treated as RAZ.
- Attempted accesses in User mode or the ones using a size other than 32 bits are not permitted. They terminate with an error.

- When CP*n* requests to read IRCP*n*ISR*p*, MSCM returns the entire content of IRCP*n*ISR*p*.
- When the debugger requests to read IRCP*n*ISR*p*, MSCM returns the entire content of IRCP*n*ISR*p*.
- When a trusted core, as indicated by IRCPCFG, requests to read IRCP*n*ISR*p*, MSCM returns the entire content of the IRCP0ISR*p* register.

Write access

- Writes to IRCP*n*ISR*x* are only accessible in Privileged mode using 32-bit (word) accesses.
- Writes to IRCP*n*ISR*x* follow the W1C protocol, whereby writing 1 clears the corresponding field and writing 0 is ignored.
- Privileged, 32-bit write accesses from the debugger are treated as WI.
- Privileged, 32-bit write accesses from non-core (and non-debugger) bus masters are treated as WI.
- Attempted accesses in User mode or the ones using a size other than 32 bits are not permitted. They terminate with an error.
- The target core, CP*n*, has full access to write to all the fields of the IRCP0ISR*x* register.
- A trusted core, as indicated by IRCPCFG, has full access to write to all the fields of IRCP*n*ISR*x*.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 PCIE_INT15	PCIe-to-CPn Interrupt15 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt15 to CPn, initiated by PCIe, is asserted.
14 PCIE_INT14	PCIe-to-CPn Interrupt14 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt14 to CPn, initiated by PCIe, is asserted.
13 PCIE_INT13	PCIe-to-CPn Interrupt13

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt13 to CPn, initiated by PCIe, is asserted.
12 PCIE_INT12	PCIe-to-CPn Interrupt12 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt12 to CPn, initiated by PCIe, is asserted.
11 PCIE_INT11	PCIe-to-CPn Interrupt11 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt11 to CPn, initiated by PCIe, is asserted.
10 PCIE_INT10	PCIe-to-CPn Interrupt10 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt10 to CPn, initiated by PCIe, is asserted.
9 PCIE_INT9	PCIe-to-CPn Interrupt9 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt9 to CPn, initiated by PCIe, is asserted.
8 PCIE_INT8	PCIe-to-CPn Interrupt8 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt8 to CPn, initiated by PCIe, is asserted.
7 PCIE_INT7	PCIe-to-CPn Interrupt7 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt7 to CPn, initiated by PCIe, is asserted.
6 PCIE_INT6	PCIe-to-CPn Interrupt6 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt6 to CPn, initiated by PCIe, is asserted.
5 PCIE_INT5	PCIe-to-CPn Interrupt5 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt5 to CPn, initiated by PCIe, is asserted.
4 PCIE_INT4	PCIe-to-CPn Interrupt4 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt4 to CPn, initiated by PCIe, is asserted.
3 PCIE_INT3	PCIe-to-CPn Interrupt3 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt3 to CPn, initiated by PCIe, is asserted.

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 PCIE_INT2	PCIe-to-CPn Interrupt2 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt2 to CPn, initiated by PCIe, is asserted.
1 PCIE_INT1	PCIe-to-CPn Interrupt1 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt1 to CPn, initiated by PCIe, is asserted.
0 PCIE_INT0	PCIe-to-CPn Interrupt0 0b - PCIe does not assert any interrupt to CPn. 1b - Interrupt0 to CPn, initiated by PCIe, is asserted.

8.2.3.19 Interrupt Router CPn Interruptx Generation Register (IRCP0IGR3 - IRCP6IGR3)

Offset

Register	Offset
IRCP0IGR3	21Ch
IRCP1IGR3	23Ch
IRCP2IGR3	25Ch
IRCP3IGR3	27Ch
IRCP4IGR3	29Ch
IRCP5IGR3	2BCh
IRCP6IGR3	2DCh

Function

This register provides a mechanism for PCIe to initiate an MSI to a core in the system. When PCIe writes to any of the fields in this register, the action initiates a core-to-core interrupt targeting CP*n*.

Privileged, 32-bit accesses from the:

- PCIe are treated as WORZ
- Debugger are treated as RAZ/WI
- Bus masters other than the PCIe or debugger are treated as RAZ/WI

Attempted accesses in User mode or the ones using a size other than 32 bits are not permitted. They terminate with an error.

Processed under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4c4e5-4c61-48a8-b251-390c3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	INT15_EN	INT14_EN	INT13_EN	INT12_EN	INT11_EN	INT10_EN	INT9_EN	INT8_EN	INT7_EN	INT6_EN	INT5_EN	INT4_EN	INT3_EN	INT2_EN	INT1_EN	INT0_EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 INT15_EN	Interrupt15 Enable
14 INT14_EN	Interrupt14 Enable
13 INT13_EN	Interrupt13 Enable
12 INT12_EN	Interrupt12 Enable
11 INT11_EN	Interrupt11 Enable
10 INT10_EN	Interrupt10 Enable
9 INT9_EN	Interrupt9 Enable
8 INT8_EN	Interrupt8 Enable
7 INT7_EN	Interrupt7 Enable

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab62...

Table continued from the previous page...

Field	Function
6 INT6_EN	Interrupt6 Enable
5 INT5_EN	Interrupt5 Enable
4 INT4_EN	Interrupt4 Enable
3 INT3_EN	Interrupt3 Enable
2 INT2_EN	Interrupt2 Enable
1 INT1_EN	Interrupt1 Enable
0 INT0_EN	Interrupt0 Enable

8.2.3.20 Interrupt Router CPn Interruptx Generation Register (IRCP0IGR4 - IRCP6IGR4)

Offset

Register	Offset
IRCP0IGR4	31Ch
IRCP1IGR4	33Ch
IRCP2IGR4	35Ch
IRCP3IGR4	37Ch
IRCP4IGR4	39Ch
IRCP5IGR4	3BCh
IRCP6IGR4	3DCh

Function

This register provides a mechanism for PCIe to initiate an MSI to a core in the system. When PCIe writes to any of the fields in this register, the action initiates a core-to-core interrupt targeting CP*n*.

Privileged, 32-bit accesses from the:

- PCIe are treated as WORZ

- Debugger are treated as RAZ/WI
- Bus masters other than the PCIe or debugger are treated as RAZ/WI

Attempted accesses in User mode or the ones using a size other than 32 bits are not permitted. They terminate with an error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	INT15_EN	INT14_EN	INT13_EN	INT12_EN	INT11_EN	INT10_EN	INT9_EN	INT8_EN	INT7_EN	INT6_EN	INT5_EN	INT4_EN	INT3_EN	INT2_EN	INT1_EN	INT0_EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 INT15_EN	Interrupt15 Enable
14 INT14_EN	Interrupt14 Enable
13 INT13_EN	Interrupt13 Enable
12 INT12_EN	Interrupt12 Enable
11 INT11_EN	Interrupt11 Enable
10 INT10_EN	Interrupt10 Enable
9 INT9_EN	Interrupt9 Enable
8	Interrupt8 Enable

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3e6240c1

Table continued from the previous page...

Field	Function
INT8_EN	
7 INT7_EN	Interrupt7 Enable
6 INT6_EN	Interrupt6 Enable
5 INT5_EN	Interrupt5 Enable
4 INT4_EN	Interrupt4 Enable
3 INT3_EN	Interrupt3 Enable
2 INT2_EN	Interrupt2 Enable
1 INT1_EN	Interrupt1 Enable
0 INT0_EN	Interrupt0 Enable

8.2.3.21 Interrupt Router Configuration Register (IRCPCFG)

Offset

Register	Offset
IRCPCFG	400h

Function

This register provides a mechanism to designate specific cores in the system as trusted. These trusted cores are allowed to access and manage outstanding MSIs.

Privileged, 32-bit accesses from the:

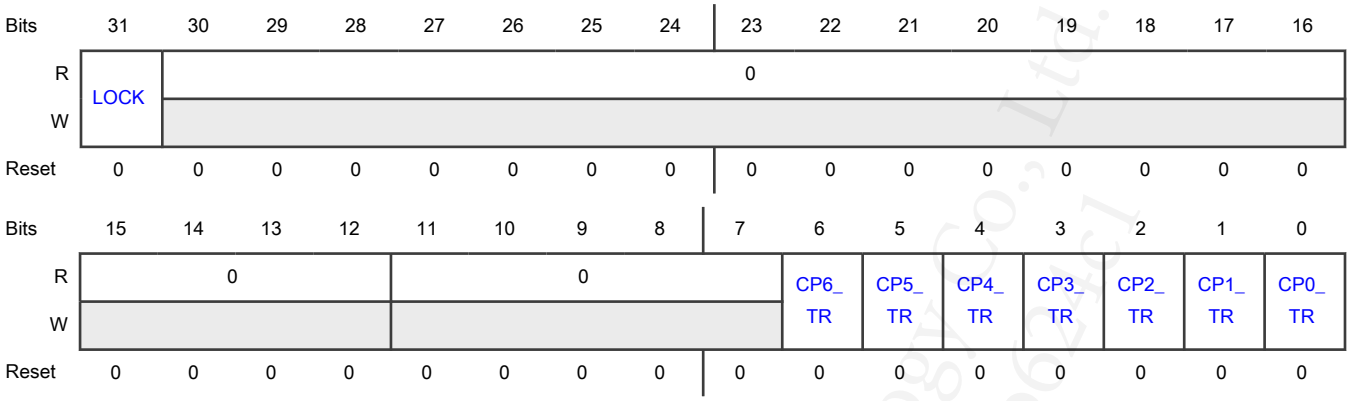
- Cortex-A53 or Cortex-M7 cores are treated as R/W
- Debugger are treated as R/WI
- Non-core (and non-debugger) bus master are not supported

Attempted accesses in User mode or the ones using a size other than 32 bits are not permitted. They terminate with an error.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c4625-4821-48a8-b251-390d3ab624c1

You can access this register only in Privileged mode for both reads and writes.

Diagram



Fields

Field	Function
31 LOCK	This field provides a locking mechanism that can be used to limit the ability to write to the register. If the value of this field is 1, it remains asserted until the next reset. 0b - Register can be written to by any privileged write. 1b - Register is locked (read-only) until the next reset.
30-12 —	Reserved
11-7 —	Reserved
6 CP6_TR	This field indicates that CP6 is a trusted core with access to read the full contents of the IRCP*n*ISR*x* register. 0b - CP6 is not trusted. 1b - CP6 is trusted.
5 CP5_TR	This field indicates that CP5 is a trusted core with access to read the full contents of the IRCP*n*ISR*x* register. 0b - CP5 is not trusted. 1b - CP5 is trusted.
4 CP4_TR	This field indicates that CP4 is a trusted core with access to read the full contents of the IRCP*n*ISR*x* register. 0b - CP4 is not trusted. 1b - CP4 is trusted.
3	This field indicates that CP3 is a trusted core with access to read the full contents of the IRCP*n*ISR*x* register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
CP3_TR	0b - CP3 is not trusted. 1b - CP3 is trusted.
2 CP2_TR	This field indicates that CP2 is a trusted core with access to read the full contents of the IRCP*n*ISR*x* register. 0b - CP2 is not trusted. 1b - CP2 is trusted.
1 CP1_TR	This field indicates that CP1 is a trusted core with access to read the full contents of the IRCP*n*ISR*x* register. 0b - CP1 is not trusted. 1b - CP1 is trusted.
0 CP0_TR	This field indicates that CP0 is a trusted core with access to read the full contents of the IRCP*n*ISR*x* register. 0b - CP0 is not trusted. 1b - CP0 is trusted.

8.2.3.22 Interrupt Router Non-Maskable Interrupt Control Register (IRNMIC)

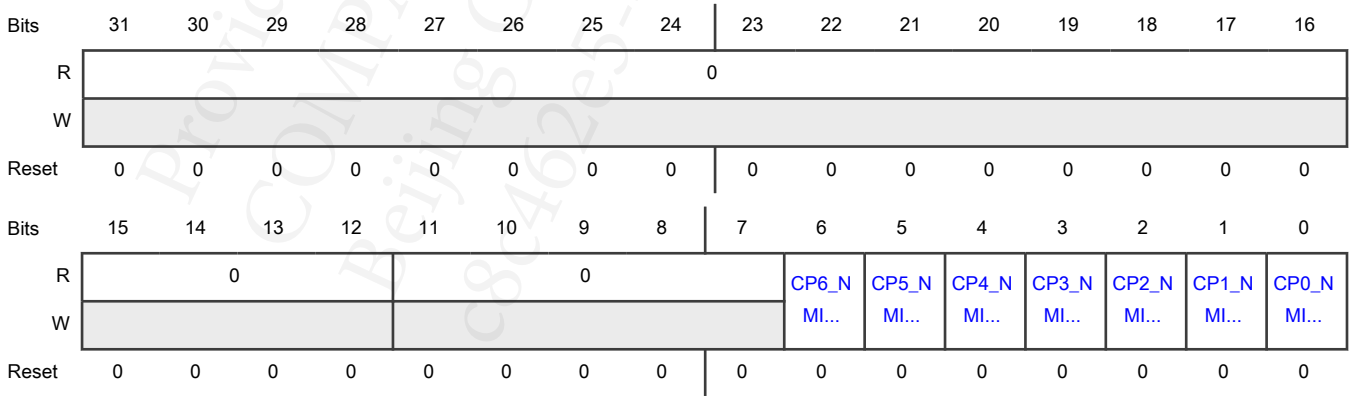
Offset

Register	Offset
IRNMIC	800h

Function

This register enables the routing control for the non-maskable interrupt (NMI) request to specific cores. For this chip, each interrupt request can be routed using the enable fields in this register. If the value of all the fields is 0, the interrupt request is disabled. Privileged accesses from non-core (and non-debug) bus masters are treated as RAZ/WI and any attempted User mode reference is terminated with an error.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-7 —	Reserved
6 CP6_NMI_EN	This field enables CP6 NMI interrupt steering to Cortex-M7 core 2 NMI. 0b - CP6 is not enabled. 1b - CP6 is enabled.
5 CP5_NMI_EN	This field enables CP5 NMI interrupt steering to Cortex-M7 core 1 NMI. 0b - CP5 is not enabled. 1b - CP5 is enabled.
4 CP4_NMI_EN	This field enables CP4 NMI interrupt steering to Cortex-M7 core 0 NMI. 0b - CP4 is not enabled. 1b - CP4 is enabled.
3 CP3_NMI_EN	This field enables CP3 NMI interrupt steering to Cortex-A53 cluster 1 core 1 nSEI. 0b - CP3 is not enabled. 1b - CP3 is enabled.
2 CP2_NMI_EN	This field enables CP2 NMI interrupt steering to Cortex-A53 cluster 1 core 0 nSEI. 0b - CP2 is not enabled. 1b - CP2 is enabled.
1 CP1_NMI_EN	This field enables CP1 NMI interrupt steering to Cortex-A53 cluster 0 core 1 nSEI. 0b - CP1 is not enabled. 1b - CP1 is enabled.
0 CP0_NMI_EN	This field enables CP0 NMI interrupt steering to Cortex-A53 cluster 0 core 0 nSEI. 0b - CP0 is not enabled. 1b - CP0 is enabled.

8.2.3.23 Interrupt Router Shared Peripheral Routing Control Register (IRSPRC0 - IRSPRC239)

Offset

For a = 0 to 239:

Register	Offset
IRSPRCa	880h + (a × 2h)

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-4821-4aa8-2513-9003ab624c1

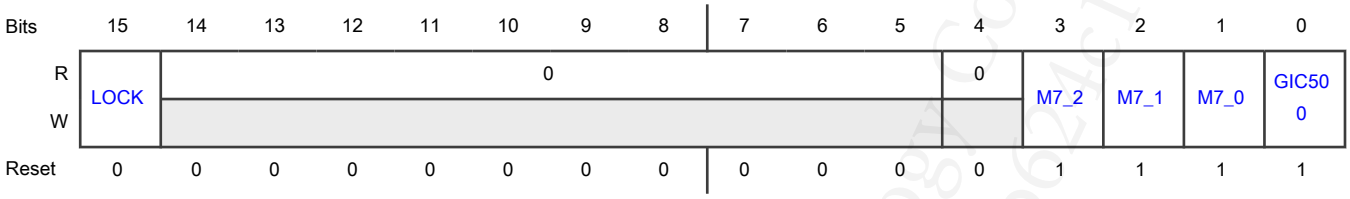
Function

This register provides an array of halfword (16-bit) registers, where each register defines the routing control for the corresponding interrupt request. For this chip, each interrupt request can be routed using the bit-mapped fields in IRSPRC*n*. If all the CP*x*E*n* bits are cleared, the interrupt request is disabled. For each routing control, halfword can be locked by asserting the LOCK field.

Non-core, non-debugger accesses to these registers receive errors and the read data is undefined.

Access: Privileged read/write

Diagram



Fields

Field	Function
15 LOCK	Lock Interrupt Request This field provides a mechanism to lock the routing of the corresponding interrupt request. After the field is asserted, attempted writes to the IRSPRC*n* register are ignored until the next reset clears the flag. 0b - Writes to the IRSPRCn are allowed. 1b - Writes to the IRSPRCn are ignored.
14-5 —	Reserved
4 —	Reserved
3 M7_2	Enable Cortex-M7_2 Interrupt Steering This field enables the routing of the corresponding interrupt request to Cortex-M7_2.
2 M7_1	Enable Cortex-M7_1 Interrupt Steering This field enables the routing of the corresponding interrupt request to Cortex-M7_1.
1 M7_0	Enable Cortex-M7_0 Interrupt Steering This field enables the routing of the corresponding interrupt request to Cortex-M7_0.
0 GIC500	Enable GIC500 Interrupt Steering This field enables the routing of the corresponding interrupt request to GIC500 associated with the Cortex-A53 clusters.

8.3 Functional description

This section discusses the following topics:

- MSI routing

— Core-to-core MSIs

— PCIe-to-core MSIs

- Interrupt steering and semaphores
  - — Interrupt handling overview
  - — MSCM interrupt router functional description

8.3.1 MSI routing

MSIs are interrupts that are indirectly broadcast to a target core by writing configuration bits in MSCM. These interrupts can either be initiated by one core targeting another core in the system (known as core-to-core interrupts), or by PCIe targeting a core in the system (known as PCIe-to-core interrupts).

These MSIs are initiated via writes to IRCP*n*IGR*x* and managed through the IRCP*n*ISR*x* registers, where *n* indicates the logical core number (0-6) and *x* represents the interrupt number (0-3). Each of the seven cores can support up to three outstanding core-to-core interrupts and one outstanding PCIe-to-core interrupt.

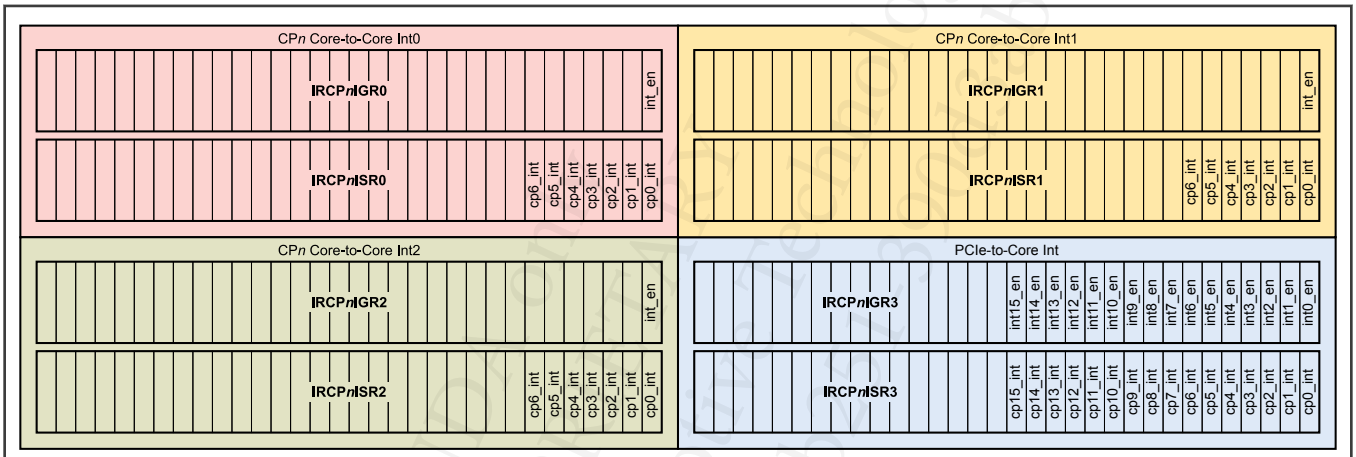


Figure 16. IRCP*n*IGR*x*/IRCP*n*ISR*x* pairs for one core

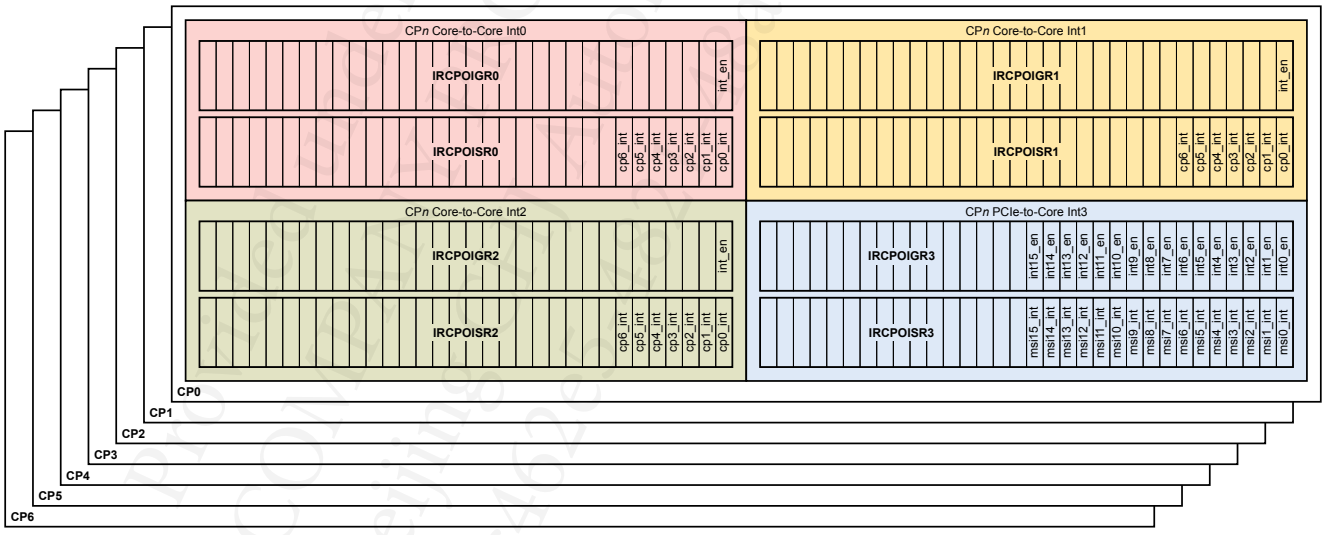


Figure 17. IRCP*n*IGR*x*/IRCP*n*ISR*x* pairs per core

8.3.1.1 Core-to-core MSIs

The next figure depicts the sequence for initiating a core-to-core MSI, in which CP*m* represents the initiating core, CP*n* represents the target core, and CP*x* indicates the MSI number. CP*m* writes to the IRCP*n*IGR*x* to initiate an MSI. The outstanding MSI that CP*m* initiates, targeting CP*n*, is reflected in the corresponding bit-mapped field in IRCP*n*ISR*x*. For example, an MSI is initiated by CP6 by writing to IRCP*n*IGR[INT_EN], and then reflected in IRCP*n*ISR*x*[CP6_INT].

At any given time, there can be up to three pending MSIs initiated by different cores.

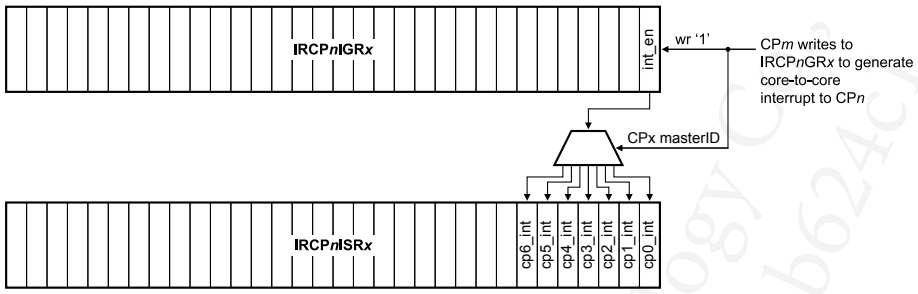


Figure 18. Initiating a core-to-core MSI via IRCP*n*IGR{0,1,2}/IRCP*n*ISR{0,1,2}

8.3.1.2 PCIe-to-core MSIs

The following figure depicts the sequence for initiating a PCIe-to-core interrupt, in which CP*n* represents the target core. PCIe writes to IRCP*n*IGR*p* to initiate an MSI. The outstanding MSI targeting CP*n* is reflected in IRCP*n*ISR3. This register contains bit-mapped fields indicating the initiating source of the MSI. A given core can support up to 16 outstanding PCIe-initiated MSIs.

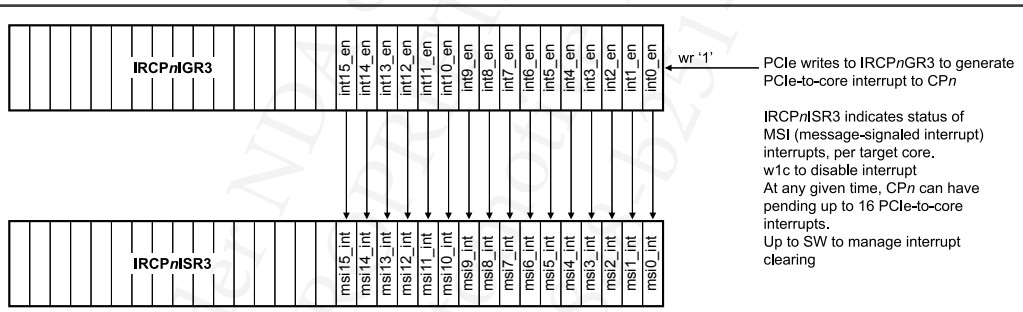


Figure 19. Initiating a core-to-core MSI via IRCP*n*IGR*p*/IRCP*n*ISR*p*

8.3.2 Interrupt steering and semaphores

This section discusses these topics:

- Interrupt handling overview
- MSCM interrupt router functional description

8.3.2.1 Interrupt handling overview

The interrupt handling mechanisms of the Cortex-A53 and Cortex-M7 cores differ significantly. The Cortex-A53 processor uses a simple, two-interrupt request mechanism: fast interrupt request(FIQ) and interrupt request (IRQ), with two corresponding exception vector instructions, while the Cortex-M7 has a nested vector interrupt controller (NVIC) tightly coupled to the processor core. The real-time performance of Cortex-M7 means that the NVIC directly provides the appropriate interrupt vector, in the form of the starting instruction address for the interrupt handler, to the core. These Cortex-M7 architectural features directly translate into faster interrupt status register (ISR) entries and exit capabilities, with an improved runtime performance. See the Arm core technical reference manuals for details.

In the Cortex-A53 core complex, a generic interrupt controller (GIC) manages the IRQ collection and prioritization for the processor and provides a functionality similar to the NVIC of Cortex-M7. To minimize software complexity, a common view of the interrupt requests and their routing mechanisms is implemented, although it must be recognized that the differences between the NVIC and the GIC do require the ISRs to be customized for the appropriate processor.

In this architecture, a total of 240 IRQs are supported where this parameter is defined by the realistic limits of the NVIC implementation, both in terms of silicon size and supported frequency of operation. These 240 IRQs are split into a total of eight directed requests and 232+ shared peripheral requests. Unless noted otherwise, let the directed requests be defined as INT[7:0] and the shared peripheral requests as INT[239:8]. The association of the interrupt request bit number and its corresponding position within the IRQ vectors is needed for both the NVIC and GIC service routines.

8.3.2.2 MSCM interrupt router functional description

As described in MSCM register descriptions, the interrupt routing registers enable the steering of requests to the processor cores.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 9
Miscellaneous Control Module (MCM)

9.1 Introduction

The Miscellaneous Control Module (MCM) provides miscellaneous control functions and contains Cortex-M7 local memory descriptors.

NOTE

MCM is accessible using the Private Peripheral Bus of Cortex-M7.

9.1.1 Features

The MCM includes the following features:

- Program-visible information on the platform configuration and revision
- Floating Point Exception monitor and interrupt control
- Local memory descriptors:
  - ITCM
  - D0TCM
  - D1TCM
  - ICACHE
  - DCACHE

9.2 Memory map/register descriptions

The memory map and register descriptions below describe the registers using byte addresses.

9.2.1 MCM register descriptions

9.2.1.1 MCM memory map

MCM base address: E008_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	SoC-defined Platform Revision (PLREV)	16	RO	0000h
2h	Processor Core Type (PCT)	16	RO	AC70h
Ch	Core Platform Control (CPCR)	32	RW	0000_0200h
10h	Interrupt Status and Control (ISCR)	32	RW	0000_0000h
400h	Local Memory Descriptor 0 (LMEM_DESC_0)	32	RW	0000_0000h
404h - 408h	Local Memory Descriptor a (LMEM_DESC_1 - LMEM_DESC_2)	32	RW	8604_2000h
40Ch	Local Memory Descriptor 3 (LMEM_DESC_3)	32	RW	8626_4000h
410h	Local Memory Descriptor 4 (LMEM_DESC_4)	32	RW	8644_6000h

9.2.1.2 SoC-defined Platform Revision (PLREV)

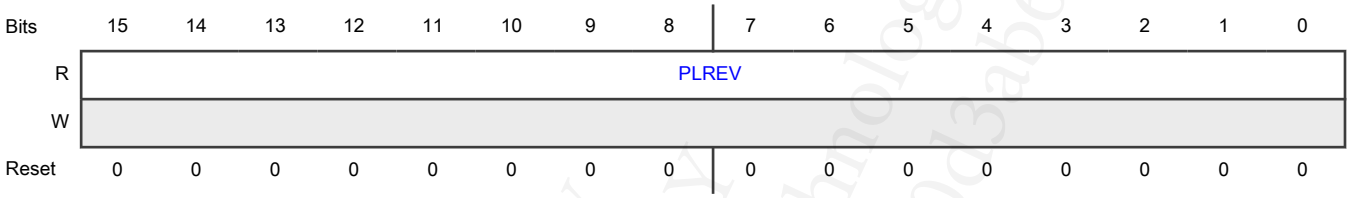
Offset

Register	Offset
PLREV	0h

Function

The PLREV is a 16-bit read-only register specifying an SoC-defined platform revision number. The state of this register is defined by a platform input signal; it can only be read from the IPS programming model. Any attempted write is ignored.

Diagram



Fields

Field	Function
15-0 PLREV	The PLREV[15:0] field is specified by a platform input signal to define a software-visible revision number.

9.2.1.3 Processor Core Type (PCT)

Offset

Register	Offset
PCT	2h

Function

The PCT is a 16-bit read-only register specifying the architecture of the processor core within the platform on the device. The state of this register is defined by a module input signal, which can only be read from the IPS programming model. Any attempted write is ignored.

Provided under NDA only
COMPANY PROPRIETARY
Beijing ZHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b231-3903ab6241

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PCT															
W																
Reset	1	0	1	0	1	1	0	0	0	1	1	1	0	0	0	0

Fields

Field	Function
15-0 PCT	This MCM design supports the Arm Cortex M7 core. The following value identifies this core complex. 1010110001110000b - Arm Cortex M7

9.2.1.4 Core Platform Control (CPCR)

Offset

Register	Offset
CPCR	Ch

Function

CR defines the arbitration and protection schemes for the two system RAM arrays.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				CM7_AHB...	Reserved										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					Reserved	Reserved		Reserved							
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a3-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
27 CM7_AHBSPRI	AHB Slave Priority This field indicates the access priority on the AHBS port of the Cortex-M7. **NOTE** This setting has no effect unless enabled by AHBSCR[CTL][1] of the CM7 core. 0b - Uses a round-robin arbitration scheme 1b - AHB-slave access has priority over a core access
26-11 —	Reserved
10 —	Reserved
9 —	Reserved
8-0 —	Reserved

1. For more information see Cortex-M7 documentation: Arm Cortex-M7 Processor Technical Reference Manual at www.arm.com.

9.2.1.5 Interrupt Status and Control (ISCR)

Offset

Register	Offset
ISCR	10h

Function

The MCM_ISCR register defines the configuration of, and reports status for, a number of core-related interrupt exception conditions. It includes:

- Enable and status fields associated with the core's floating-point exceptions
- Bus errors associated with the core's cache write buffer
- Events associated with the debug ETB module

The individual event indicators are first qualified with their exception enables, and then logically summed to form an interrupt request sent to the core's NVIC.

Bits 15-8 are read-only indicator flags based on the processor's FPSCR register. Attempted writes to these fields are ignored. When these flags are 1, they retain this value until software clears the corresponding FPSCR field. For more information see Cortex-M7 documentation: Arm Cortex-M7 Processor Technical Reference Manual at www.arm.com.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FIDCE	0		FIXCE	FUFCE	FOFCE	FDZCE	FIOCE	0		WABE	0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FIDC	0		FIXC	FUFC	FOFC	FDZC	FIOC	0	WABSO	WABS	0				
W											W1C					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 FIDCE	FPU Input Denormal Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
30-29 —	Reserved
28 FIXCE	FPU Inexact Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
27 FUFCE	FPU Underflow Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
26 FOFCE	FPU Overflow Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
25 FDZCE	FPU Divide-by-Zero Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
24 FIOCE	FPU Invalid Operation Interrupt Enable 0b - Disable interrupt 1b - Enable interrupt
23-22	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
21 WABE	TCM Write Abort Interrupt Enable 0b - Disable interrupt 1b - Enable Interrupt
20-16 —	Reserved
15 FIDC	FPU Input Denormal Interrupt Status This read-only field is a copy of the core's FPSCR[IDC] field. It indicates that an input denormalized number has been detected in the processor's FPU. When this field is 1, it retains this value until software clears the FPSCR[IDC] field. 0b - No interrupt 1b - Interrupt occurred
14-13 —	Reserved
12 FIXC	FPU Inexact Interrupt Status This read-only field is a copy of the core's FPSCR[IXC] field. It indicates that an inexact number has been detected in the processor's FPU. When this field is 1, it retains this value until software clears the FPSCR[IXC] field. 0b - No interrupt 1b - Interrupt occurred
11 FUFC	FPU Underflow Interrupt Status This read-only field is a copy of the core's FPSCR[UFC] field. It indicates that an underflow has been detected in the processor's FPU. When this field is 1, it retains this value until software clears the FPSCR[UFC] field. 0b - No interrupt 1b - Interrupt occurred
10 FOFC	FPU Overflow Interrupt Status This read-only field is a copy of the core's FPSCR[OFC] field. It indicates that an overflow has been detected in the processor's FPU. When this field is 1, it retains this value until software clears the FPSCR[OFC] field. 0b - No interrupt 1b - Interrupt occurred
9	FPU Divide-by-Zero Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
FDZC	This read-only field is a copy of the core's FPSCR[DZC] field. It indicates that a divide-by-zero operation has been detected in the processor's FPU. When this field is 1, it retains this value until software clears the FPSCR[DZC] field. 0b - No interrupt 1b - Interrupt occurred
8 FIOC	FPU Invalid Operation Interrupt Status This read-only field is a copy of the core's FPSCR[IOC] field. It indicates that an illegal operation has been detected in the processor's FPU. When this field is 1, it retains this value until software clears the FPSCR[IOC] field. 0b - No interrupt 1b - Interrupt occurred
7 —	Reserved
6 WABSO	Write Abort on Slave Overrun The overrun conditions are reported only if WABE=1. 0b - No write abort overrun 1b - Write abort overrun occurred
5 WABS	Write Abort on Slave This field indicates when a write abort has occurred on the AHBS interface. 0b - No write abort occurred on AHBS interface 1b - Write abort occurred on AHBS interface
4-0 —	Reserved

9.2.1.6 Local Memory Descriptor 0 (LMEM_DESC_0)

Offset

Register	Offset
LMEM_DESC_0	400h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a3-b251-390d3ab6f4c1

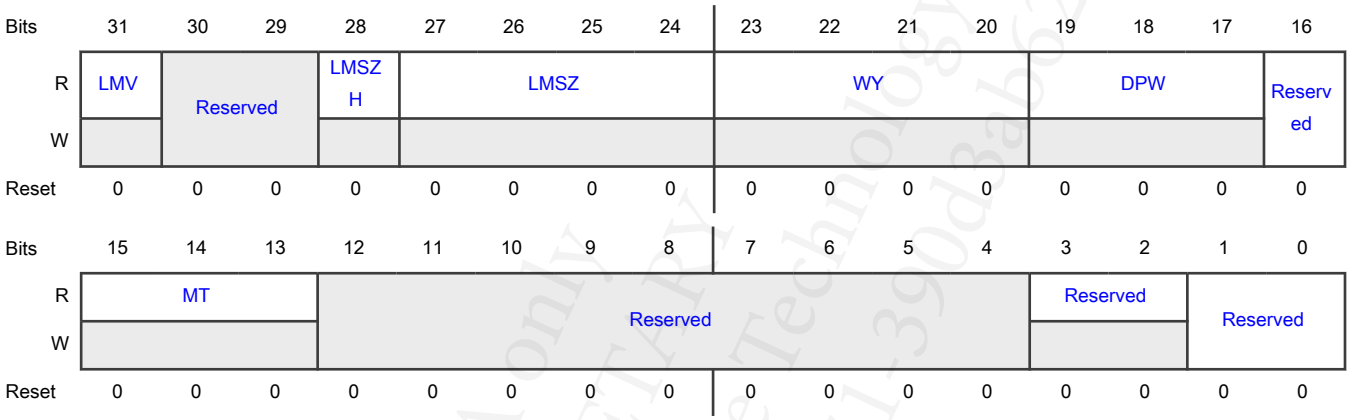
Function

NOTE

The DESC_a registers map to the LMEMs like this:

- DESC_0: Not present
- DESC_1: D0TCM
- DESC_2: D1TCM
- DESC_3: ICACHE
- DESC_4: DCACHE

Diagram



Fields

Field	Function
31 LMV	Local Memory Valid This read-only field defines the validity (presence) of the local memory. 0b - LMEM*n* not present 1b - LMEM*n* present
30-29 —	Reserved
28 LMSZH	LMEM Size Hole This field is used for local memories that are not fully populated (that is, local memories that include a memory "hole" in the upper 25 % of the address range). 0b - LMEM*n* is a power-of-2 capacity 1b - LMEM*n* is not a power-of-2, with capacity of 0.75 x LMSZ
27-24 LMSZ	Local Memory Size 0000b - 0 KB

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0001b - 1 KB 0010b - 2 KB 0011b - 4 KB 0100b - 8 KB 0101b - 16 KB 0110b - 32 KB 0111b - 64 KB 1000b - 128 KB 1001b - 256 KB 1010b - 512 KB 1011b - 1024 KB 1100b - 2048 KB 1101b - 4096 KB 1110b - 8192 KB 1111b - 16384 KB
23-20 WY	Level 1 Cache Ways 0000b - No cache 0010b - 2-way set associative 0100b - 4-way set associative
19-17 DPW	Data Path Width LMEM data path width. This read-only field defines the width of the local memory. 000-001b - Reserved 010b - LMEM*n* is 32-bits wide 011b - LMEM*n* is 64-bits wide 100-111b - Reserved
16 —	Reserved
15-13 MT	Memory Type 000b - ITCM 001b - DTCM 010b - ICACHE 011b - DCACHE

Table continues on the next page...

Table continued from the previous page...

Field	Function
12-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

9.2.1.7 Local Memory Descriptor a (LMEM_DESC_1 - LMEM_DESC_2)

Offset

Register	Offset
LMEM_DESC_1	404h
LMEM_DESC_2	408h

Function

NOTE

The DESC_a registers map to the LMEMs like this:

- DESC_0: Not present
- DESC_1: D0TCM
- DESC_2: D1TCM
- DESC_3: ICACHE
- DESC_4: DCACHE

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LMV	Reserved		LMSZH	LMSZ				WY				DPW			Reserved
W																
Reset	1	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MT			Reserved									Reserved		Reserved	
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 LMV	Local Memory Valid This read-only field defines the validity (presence) of the local memory. 0b - LMEM*n* not present 1b - LMEM*n* present
30-29 —	Reserved
28 LMSZH	LMEM Size Hole This field is used for local memories that are not fully populated (that is, local memories that include a memory "hole" in the upper 25 % of the address range). 0b - LMEM*n* is a power-of-2 capacity 1b - LMEM*n* is not a power-of-2, with capacity of 0.75 x LMSZ
27-24 LMSZ	Local Memory Size 0000b - 0 KB 0001b - 1 KB 0010b - 2 KB 0011b - 4 KB 0100b - 8 KB 0101b - 16 KB 0110b - 32 KB 0111b - 64 KB 1000b - 128 KB 1001b - 256 KB 1010b - 512 KB 1011b - 1024 KB 1100b - 2048 KB 1101b - 4096 KB 1110b - 8192 KB 1111b - 16384 KB
23-20 WY	Level 1 Cache Ways 0000b - No cache 0010b - 2-way set associative 0100b - 4-way set associative
19-17	Data Path Width

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3a1624c1

Table continued from the previous page...

Field	Function
DPW	LMEM data path width. This read-only field defines the width of the local memory. 000-001b - Reserved 010b - LMEM*n* is 32-bits wide 011b - LMEM*n* is 64-bits wide 100-111b - Reserved
16 —	Reserved
15-13 MT	Memory Type 000b - ITCM 001b - DTCM 010b - ICACHE 011b - DCACHE
12-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

9.2.1.8 Local Memory Descriptor 3 (LMEM_DESC_3)

Offset

Register	Offset
LMEM_DESC_3	40Ch

Function

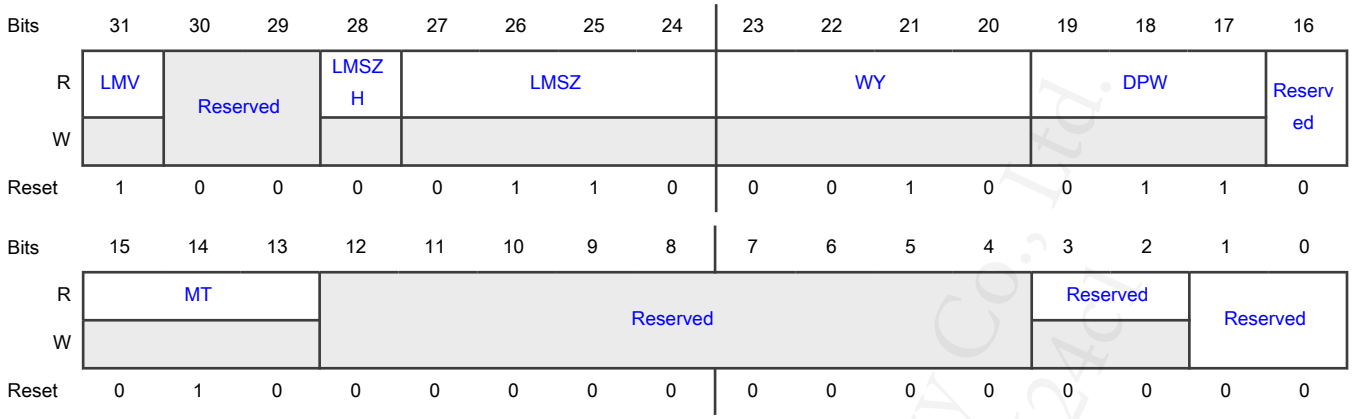
NOTE

The DESC_a registers map to the LMEMs like this:

- DESC_0: Not present
- DESC_1: D0TCM
- DESC_2: D1TCM
- DESC_3: ICACHE
- DESC_4: DCACHE

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
Date: 2021-8-25 1:39:0d3ab624c1

Diagram



Fields

Field	Function
31 LMV	Local Memory Valid This read-only field defines the validity (presence) of the local memory. 0b - LMEM*n* not present 1b - LMEM*n* present
30-29 —	Reserved
28 LMSZH	LMEM Size Hole This field is used for local memories that are not fully populated (that is, local memories that include a memory "hole" in the upper 25 % of the address range). 0b - LMEM*n* is a power-of-2 capacity 1b - LMEM*n* is not a power-of-2, with capacity of 0.75 x LMSZ
27-24 LMSZ	Local Memory Size 0000b - 0 KB 0001b - 1 KB 0010b - 2 KB 0011b - 4 KB 0100b - 8 KB 0101b - 16 KB 0110b - 32 KB 0111b - 64 KB 1000b - 128 KB 1001b - 256 KB

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1010b - 512 KB 1011b - 1024 KB 1100b - 2048 KB 1101b - 4096 KB 1110b - 8192 KB 1111b - 16384 KB
23-20 WY	Level 1 Cache Ways 0000b - No cache 0010b - 2-way set associative 0100b - 4-way set associative
19-17 DPW	Data Path Width LMEM data path width. This read-only field defines the width of the local memory. 000-001b - Reserved 010b - LMEM*n* is 32-bits wide 011b - LMEM*n* is 64-bits wide 100-111b - Reserved
16 —	Reserved
15-13 MT	Memory Type 000b - ITCM 001b - DTCM 010b - ICACHE 011b - DCACHE
12-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

9.2.1.9 Local Memory Descriptor 4 (LMEM_DESC_4)

Offset

Register	Offset
LMEM_DESC_4	410h

Function

NOTE

The DESC_a registers map to the LMEMs like this:

- DESC_0: Not present
- DESC_1: D0TCM
- DESC_2: D1TCM
- DESC_3: ICACHE
- DESC_4: DCACHE

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LMV	Reserved		LMSZH	LMSZ				WY				DPW			Reserved
W																
Reset	1	0	0	0	0	1	1	0	0	1	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MT			Reserved									Reserved		Reserved	
W																
Reset	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 LMV	Local Memory Valid This read-only field defines the validity (presence) of the local memory. 0b - LMEM*n* not present 1b - LMEM*n* present
30-29 —	Reserved
28	LMEM Size Hole

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-42a8-b251-590d3ab624c1

Table continued from the previous page...

Field	Function
LMSZH	This field is used for local memories that are not fully populated (that is, local memories that include a memory "hole" in the upper 25 % of the address range). 0b - LMEM*n* is a power-of-2 capacity 1b - LMEM*n* is not a power-of-2, with capacity of 0.75 x LMSZ
27-24 LMSZ	Local Memory Size 0000b - 0 KB 0001b - 1 KB 0010b - 2 KB 0011b - 4 KB 0100b - 8 KB 0101b - 16 KB 0110b - 32 KB 0111b - 64 KB 1000b - 128 KB 1001b - 256 KB 1010b - 512 KB 1011b - 1024 KB 1100b - 2048 KB 1101b - 4096 KB 1110b - 8192 KB 1111b - 16384 KB
23-20 WY	Level 1 Cache Ways 0000b - No cache 0010b - 2-way set associative 0100b - 4-way set associative
19-17 DPW	Data Path Width LMEM data path width. This read-only field defines the width of the local memory. 000-001b - Reserved 010b - LMEM*n* is 32-bits wide 011b - LMEM*n* is 64-bits wide 100-111b - Reserved
16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-13 MT	Memory Type 000b - ITCM 001b - DTCM 010b - ICACHE 011b - DCACHE
12-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

9.3 Functional description

9.3.1 Interrupts

MCM generates an interrupt if any of the following are true:

- FPU input denormal interrupt is enabled (FIDCE) and an input is denormalized (FIDC).
- FPU inexact interrupt is enabled (FIXCE) and a number is inexact (FIXC).
- FPU underflow interrupt is enabled (FUFCE) and an underflow occurs (FUFC).
- FPU overflow interrupt is enabled (FOFCE) and an overflow occurs (FOFC).
- FPU divide-by-zero interrupt is enabled (FDZCE) and a divide-by-zero occurs (FDZC).
- FPU invalid operation interrupt is enabled (FDZCE) and an invalid operation occurs (FDZC).
- Write abort interrupt is enabled (WABE) and a write abort occurs (CM7 WABORTS INDICATOR).

Determining interrupt source

To determine the exact source of the interrupt for Cortex-M7 core, qualify the interrupt status flags with the corresponding interrupt enable fields.

- MCM_ISCR[31:16] && MCM_ISCR[15:0]
- Search the result for asserted flags, which indicate the exact interrupt sources.

9.4 Glossary

ITCM	Instruction Tightly Coupled Memory
DTCM	Data Tightly Coupled Memory
ICACHE	Instruction Cache Memory
DCACHE	Data Cache Memory

Chapter 10
A53 Cluster General Purpose Registers (A53_GPR)

10.1 Introduction

The Cortex-A53 cluster GPRs provide status and control the configuration of the Cortex-A53 clusters/cores and GIC500.

10.2 Technical reference

Many of the register fields in this chapter are documented in more detail in *Arm® Cortex®-A53 MPCore Processor Technical Reference Manual*, Revision r0p4, at https://www.arm.com.

10.3 A53_CLUSTER_GPR register descriptions

10.3.1 A53_CLUSTER_GPR memory map

A53_GPR base address: 4007_C400h

Offset	Register	Width (In bits)	Access	Reset value
0h	GPR00 (GPR00)	32	RW	0700_036Fh
4h	GPR01 (GPR01)	32	RW	0000_0000h
18h	GPR06 (GPR06)	32	RW	0000_0004h
1Ch	GPR07 (GPR07)	32	RW	0001_0000h
20h	GPR08 (GPR08)	32	RO	0000_0000h
24h	GPR09 (GPR09)	32	RW	0000_0000h
28h	Cluster0 Core0 CPUMERRSR High (GPR10)	32	RO	0000_0000h
2Ch	Cluster0 Core0 CPUMERRSR Low (GPR11)	32	RO	0000_0000h
30h	Cluster0 Core1 CPUMERRSR High (GPR12)	32	RO	0000_0000h
34h	Cluster0 Core1 CPUMERRSR Low (GPR13)	32	RO	0000_0000h
38h	Cluster1 Core0 CPUMERRSR High (GPR14)	32	RO	0000_0000h
3Ch	Cluster1 Core0 CPUMERRSR Low (GPR15)	32	RO	0000_0000h
40h	Cluster1 Core1 CPUMERRSR High (GPR16)	32	RO	0000_0000h
44h	Cluster1 Core1 CPUMERRSR Low (GPR17)	32	RO	0000_0000h
48h	Cluster0 L2MERRSR High (GPR18)	32	RO	0000_0000h
4Ch	Cluster0 L2MERRSR Low (GPR19)	32	RO	0000_0000h
50h	Cluster1 L2MERRSR High (GPR20)	32	RO	0000_0000h
54h	GPR21 (GPR21)	32	RO	0000_0000h

Preliminary NDA only
COMPANY CONFIDENTIAL
Beijing Qiyuan Automotive Technology Co., Ltd.
cac2e3b1-6521-48a8-b251-29ed3ab6240d

10.3.2 GPR00 (GPR00)

Offset

Register	Offset
GPR00	0h

Function

Controls CA53 system counter clock division factor and configurations for Cortex-A53 clusters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved					CA53_COUNTER_CLK_DIV_VAL			Reserved		CA53_1_...	CA53_1_...	CA53_0_...	CA53_0_...	CA53_1_...	CA53_1_...
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CA53_1_...	CA53_1_...	CA53_0_...	CA53_0_...	CA53_0_...	CA53_0_...	CA53_1_...	CA53_1_...	CA53_1_...	CA53_0_...	CA53_0_...	CA53_0_...	CA53_1_...	CA53_1_...	CA53_0_...	CA53_0_...
W																
Reset	0	0	0	0	0	0	1	1	0	1	1	0	1	1	1	1

Fields

Field	Function
31-27 —	Reserved
26-24 CA53_COUNTER_CLK_DIV_VAL	CA53 System Counter Clock Divide Value A programmed value of N divides the counter clock frequency by N+1.
23-22 —	Reserved
21 CA53_1_CORE1_VINITHI	VINITHI Of Core1 Of Cluster1 Reflects the state of the VINITHI signal. See Technical reference.
20 CA53_1_CORE0_VINITHI	VINITHI Of Core0 Of Cluster1 Reflects the state of the VINITHI signal. See Technical reference.
19	VINITHI Of Core1 Of Cluster0

Table continues on the next page...

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
846255-4821-48a8-b251-50d5a624c1

Table continued from the previous page...

Field	Function
CA53_0_CORE1_VINITHI	Reflects the state of the VINITHI signal. See Technical reference.
18 CA53_0_CORE0_VINITHI	VINITHI Of Core0 Of Cluster0 Reflects the state of the VINITHI signal. See Technical reference.
17 CA53_1_CORE1_CFGTE	CFGTE Of Core1 Of Cluster1 Reflects the state of the CFGTE signal that enables T32 exceptions at reset. See Technical reference.
16 CA53_1_CORE0_CFGTE	CFGTE Of Core0 Of Cluster1 Reflects the state of the CFGTE signal that enables T32 exceptions at reset. See Technical reference.
15 CA53_1_CORE1_CFGEND	CFGEND Of Core1 Of Cluster1 Reflects the state of the CFGEND signal that describes the endianness configuration at reset. See Technical reference.
14 CA53_1_CORE0_CFGEND	CFGEND Of Core0 Of Cluster1 Reflects the state of the CFGEND signal that describes the endianness configuration at reset. See Technical reference.
13 CA53_0_CORE1_CFGTE	CFGTE Of Core1 Of Cluster0 Reflects the state of the CFGTE signal that enables T32 exceptions at reset. See Technical reference.
12 CA53_0_CORE0_CFGTE	CFGTE Of Core0 Of Cluster0 Reflects the state of the CFGTE signal that enables T32 exceptions at reset. See Technical reference.
11 CA53_0_CORE1_CFGEND	CFGEND Of Core1 Of Cluster0 Reflects the state of the CFGEND signal that describes the endianness configuration at reset. See Technical reference.
10 CA53_0_CORE0_CFGEND	CFGEND Of Core0 Of Cluster0 Reflects the state of the CFGEND signal that describes the endianness configuration at reset. See Technical reference.
9 CA53_1_BROADCASTOUTER	BROADCASTOUTER Of Cluster1 Reflects the state of the BROADCASTOUTER signal. See Technical reference.
8	BROADCASTINNER Of Cluster1

Table continues on the next page...

Table continued from the previous page...

Field	Function
CA53_1_BROADCASTINNER	Reflects the state of the BROADCASTINNER signal. See Technical reference.
7 CA53_1_BROADCASTCACHEMAINT	BROADCASTCACHEMAINT Of Cluster1 Reflects the state of the BROADCASTCACHEMAINT signal. See Technical reference.
6 CA53_0_BROADCASTOUTER	BROADCASTOUTER Of Cluster0 Reflects the state of the BROADCASTOUTER signal. See Technical reference.
5 CA53_0_BROADCASTINNER	BROADCASTINNER Of Cluster0 Reflects the state of the BROADCASTINNER signal. See Technical reference.
4 CA53_0_BROADCASTCACHEMAINT	BROADCASTCACHEMAINT Of Cluster0 Reflects the state of the BROADCASTCACHEMAINT signal. See Technical reference.
3 CA53_1_CORE1_AA64nAA32	AA64nAA32 Of Core1 Of Cluster1 Reflects the state of the register width state (AA64nAA32) signal. See Technical reference.
2 CA53_1_CORE0_AA64nAA32	AA64nAA32 Of Core0 Of Cluster1 Reflects the state of the register width state (AA64nAA32) signal. See Technical reference.
1 CA53_0_CORE1_AA64nAA32	AA64nAA32 Of Core1 Of Cluster0 Reflects the state of the register width state (AA64nAA32) signal. See Technical reference.
0 CA53_0_CORE0_AA64nAA32	AA64nAA32 Of Core0 Of Cluster0 Reflects the state of the register width state (AA64nAA32) signal. See Technical reference.

10.3.3 GPR01 (GPR01)

Offset

Register	Offset
GPR01	4h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4848-b251-590a8ab024c1

Function

Controls WFE event generation, clock gating, and CP15SDISABLE for Cortex-A53 clusters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				CA53_1_...	CA53_1_...	CA53_0_...	CA53_0_...	Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						CLUSTER...	CLUSTER...	Reserved						WFE_EVT...	WFE_EVT...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27 CA53_1_CORE1_CP15SDISABLE	CP15SDISABLE Of Core1 Of Cluster1 Reflects the state of the CP15SDISABLE signal. See Technical reference.
26 CA53_1_CORE0_CP15SDISABLE	CP15SDISABLE Of Core0 Of Cluster1 Reflects the state of the CP15SDISABLE signal. See Technical reference.
25 CA53_0_CORE1_CP15SDISABLE	CP15SDISABLE Of Core1 Of Cluster0 Reflects the state of the CP15SDISABLE signal. See Technical reference.
24 CA53_0_CORE0_CP15SDISABLE	CP15SDISABLE Of Core0 Of Cluster0 Reflects the state of the CP15SDISABLE signal. See Technical reference.
23-10 —	Reserved
9	Cluster1 Clock Gating Enable Enables or disables the gating of cluster1 clock during STANDBYWFIL2 retention state.

Table continues on the next page...

Table continued from the previous page...

Field	Function
CLUSTER1_CG_EN	0b - Disable 1b - Enable
8 CLUSTER0_CG_EN	Cluster0 Clock Gating Enable Enables or disables the gating of cluster0 clock during STANDBYWFIL2 retention state. 0b - Disable 1b - Enable
7-2 —	Reserved
1 WFE_EVT_CA53_CLUSTER1	WFE Event for Cortex-A53 Cluster1 Generates a wake-up event for cluster1 to exit from WFE retention state. It is particularly useful for heterogeneous cores in the system to wake the cluster from WFE state. Writing a 1 to this field generates an event. If you write a 1 to this field, you must then write a 0 to this field.
0 WFE_EVT_CA53_CLUSTER0	WFE Event For Cortex-A53 Cluster0 Generates a wake-up event for cluster0 to exit from WFE retention state. It is particularly useful for heterogeneous cores in the system to wake the cluster from WFE state. Writing a 1 to this field generates an event. If you write a 1 to this field, you must then write a 0 to this field.

10.3.4 GPR06 (GPR06)

Offset

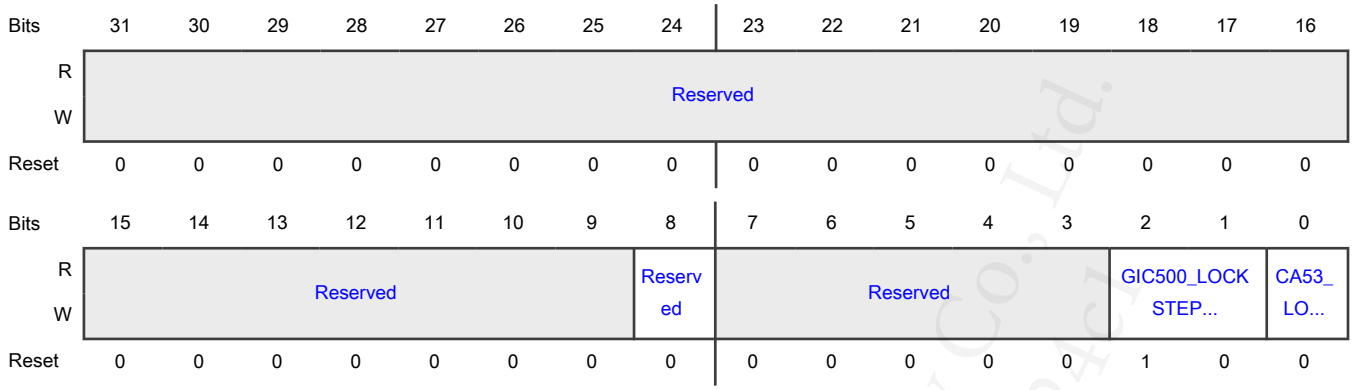
Register	Offset
GPR06	18h

Function

Controls lockstep mode for Cortex-A53 clusters and GIC500 modules. See "Arm CoreLink Generic Interrupt Controller (GIC-500)" in the "ARM Modules" chapter.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b253-3903ab624c1

Diagram



Fields

Field	Function
31-9 —	Reserved
8 —	Reserved
7-3 —	Reserved
2-1 GIC500_LOCKSTEP_EN	Lockstep Mode For GIC500 Enables or disables Lockstep mode. 00b - Lockstep disabled with GIC0 enabled and GIC1 disabled 01b - Lockstep disabled with GIC0 disabled and GIC1 enabled 10b - Lockstep enabled 11b - Lockstep enabled
0 CA53_LOCKSTEP_EN	Lockstep Mode For Cortex-A53 Clusters Enables or disables Lockstep mode. Do not enable ACE DVM Snoop for Cortex-A53 cluster1 in Ncore if Lockstep mode is enabled. 0b - Disable 1b - Enable

10.3.5 GPR07 (GPR07)

Offset

Register	Offset
GPR07	1Ch

Provided under NDA only
COMPANY CONFIDENTIAL
Beijing CSI Automotive Technology Co., Ltd.
c8c4625-4821-482-b251-390d3ab624

Function

Drives cluster ID affinity values for Cortex-A53 clusters.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CA53_1_CLUSTERIDAFF2								CA53_1_CLUSTERIDAFF1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CA53_0_CLUSTERIDAFF2								CA53_0_CLUSTERIDAFF1							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 CA53_1_CLUSTERIDAFF2	Cluster ID Affinity Level 2 For Cortex-A53 Cluster1
23-16 CA53_1_CLUSTERIDAFF1	Cluster ID Affinity Level 1 For Cortex-A53 Cluster1
15-8 CA53_0_CLUSTERIDAFF2	Cluster ID Affinity Level 2 For Cortex-A53 Cluster0
7-0 CA53_0_CLUSTERIDAFF1	Cluster ID Affinity Level 1 For Cortex-A53 Cluster0

10.3.6 GPR08 (GPR08)

Offset

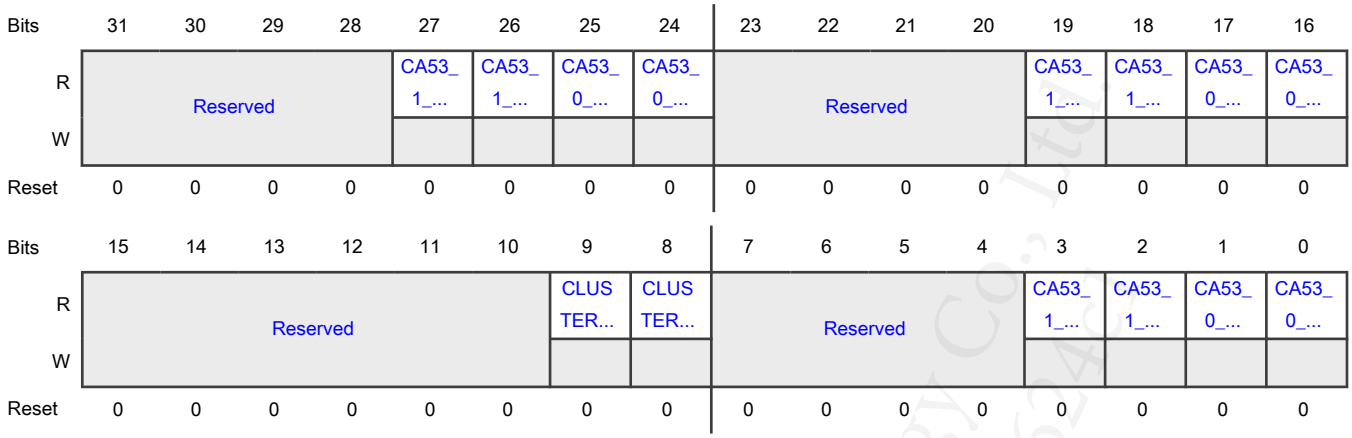
Register	Offset
GPR08	20h

Function

Provides the status of WFI, WFE, WFIL2, and warm reset of Cortex-A53 cluster cores.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cc4625-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-28 —	Reserved
27 CA53_1_CORE1_STANDBYWFI_STATUS	STANDBYWFI Status Of Core1 Of Cluster1 Indicates the STANDBYWFI status of core1 of cluster1. 0b - Not in STANDBYWFI state 1b - In STANDBYWFI state
26 CA53_1_CORE0_STANDBYWFI_STATUS	STANDBYWFI Status Of Core0 Of Cluster1 Indicates the STANDBYWFI status of core0 of cluster1. 0b - Not in STANDBYWFI state 1b - In STANDBYWFI state
25 CA53_0_CORE1_STANDBYWFI_STATUS	STANDBYWFI Status Of Core1 Of Cluster0 Indicates the STANDBYWFI status of core1 of cluster0. 0b - Not in STANDBYWFI state 1b - In STANDBYWFI state
24 CA53_0_CORE0_STANDBYWFI_STATUS	STANDBYWFI Status Of Core0 Of Cluster0 Indicates the STANDBYWFI status of core0 of cluster0. 0b - Not in STANDBYWFI state 1b - In STANDBYWFI state
23-20 —	Reserved
19	STANDBYWFE Status Of Core1 Of Cluster1

Table continues on the next page...

Table continued from the previous page...

Field	Function
CA53_1_CORE1_STANDBYWFE_STATUS	Indicates the STANDBYWFE status of core1 of cluster1. 0b - Not in STANDBYWFE state 1b - In STANDBYWFE state
18 CA53_1_CORE0_STANDBYWFE_STATUS	STANDBYWFE Status Of Core0 Of Cluster1 Indicates the STANDBYWFE status of core0 of cluster1. 0b - Not in STANDBYWFE state 1b - In STANDBYWFE state
17 CA53_0_CORE1_STANDBYWFE_STATUS	STANDBYWFE Status Of Core1 Of Cluster0 Indicates the STANDBYWFE status of core1 of cluster0. 0b - Not in STANDBYWFE state 1b - In STANDBYWFE state
16 CA53_0_CORE0_STANDBYWFE_STATUS	STANDBYWFE Status Of Core0 Of Cluster0 Indicates the STANDBYWFE status of core0 of cluster0. 0b - Not in STANDBYWFE state 1b - In STANDBYWFE state
15-10 —	Reserved
9 CLUSTER1_STANDBYWFIL2_STATUS	STANDBYWFIL2 Status Of Cortex-A53 Cluster1 Indicates the STANDBYWFIL2 status of Cortex-A53 cluster1. 0b - Not in STANDBYWFIL2 state 1b - In STANDBYWFIL2 state
8 CLUSTER0_STANDBYWFIL2_STATUS	STANDBYWFIL2 Status Of Cortex-A53 Cluster0 Indicates the STANDBYWFIL2 status of Cortex-A53 cluster0. 0b - Not in STANDBYWFIL2 state 1b - In STANDBYWFIL2 state
7-4 —	Reserved
3 CA53_1_CORE1_WARM_RESET_STATUS	Warm Reset Status Of Core1 Of Cluster1 Indicates the warm reset status of core1 of cluster1. 0b - Reset in progress 1b - Reset request completed

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 CA53_1_CORE0_WARM_RESET_STATUS	Warm Reset Status Of Core0 Of Cluster1 Indicates the warm reset status of core0 of cluster1. 0b - Reset in progress 1b - Reset request completed
1 CA53_0_CORE1_WARM_RESET_STATUS	Warm Reset Status Of Core1 of Cluster0 Indicates the warm reset status of core1 of cluster0. 0b - Reset in progress 1b - Reset request completed
0 CA53_0_CORE0_WARM_RESET_STATUS	Warm Reset Status Of Core0 Of Cluster0 Indicates the warm reset status of core0 of cluster0. 0b - Reset in progress 1b - Reset request completed

10.3.7 GPR09 (GPR09)

Offset

Register	Offset
GPR09	24h

Function

Drives the uppermost byte of the 40-bit reset vector base address of Cortex-A53 cores.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CA53_1_CORE1_RVBARADDR_39_32								CA53_1_CORE0_RVBARADDR_39_32							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CA53_0_CORE1_RVBARADDR_39_32								CA53_0_CORE0_RVBARADDR_39_32							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-4aa8-b251-390d3ab624c1

Fields

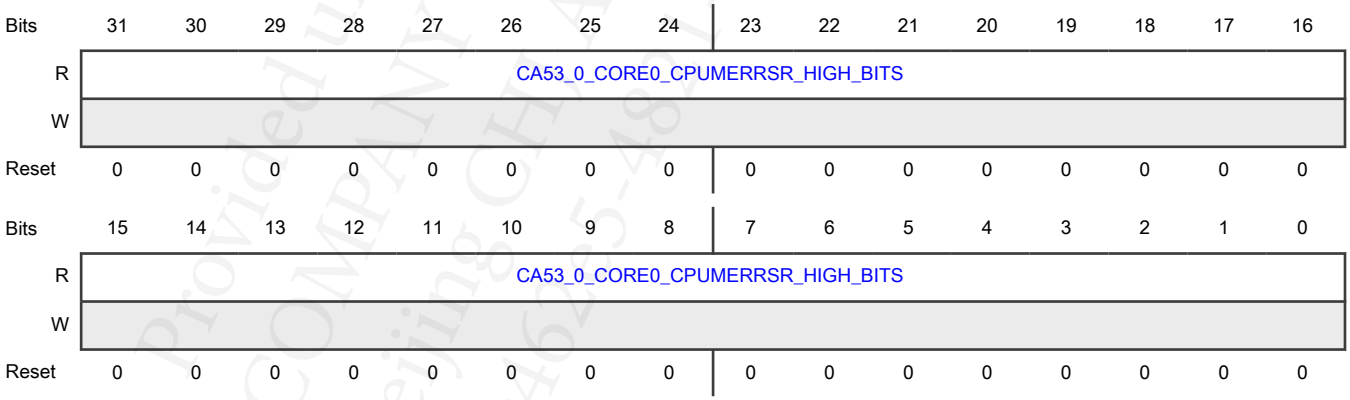
Field	Function
31-24 CA53_1_CORE1_RVBARADDR_39_32	Uppermost Byte Of 40-bit Reset Vector Base Address Of Core1 Of Cortex-A53 cluster1 Drives RVBARADDR1[39:32] of Cortex-A53 cluster1.
23-16 CA53_1_CORE0_RVBARADDR_39_32	Uppermost Byte Of 40-bit Reset Vector Base Address Of Core0 Of Cortex-A53 Cluster1 Drives RVBARADDR0[39:32] of Cortex-A53 cluster1.
15-8 CA53_0_CORE1_RVBARADDR_39_32	Uppermost Byte Of 40-bit Reset Vector Base Address Of Core1 Of Cortex-A53 Cluster0 Drives RVBARADDR1[39:32] of Cortex-A53 cluster0.
7-0 CA53_0_CORE0_RVBARADDR_39_32	Uppermost Byte Of 40-bit Reset Vector Base Address Of Core0 Of Cortex-A53 Cluster0 Drives RVBARADDR0[39:32] of Cortex-A53 cluster0.

10.3.8 Cluster0 Core0 CPUMERRSR High (GPR10)

Offset

Register	Offset
GPR10	28h

Diagram



Provided under NDA
COMPANY PROPRIETARY
Beijing CHA Automotive Technology Co., Ltd.
c8c46e25-482-48a8-b251-390d3ab624c

Fields

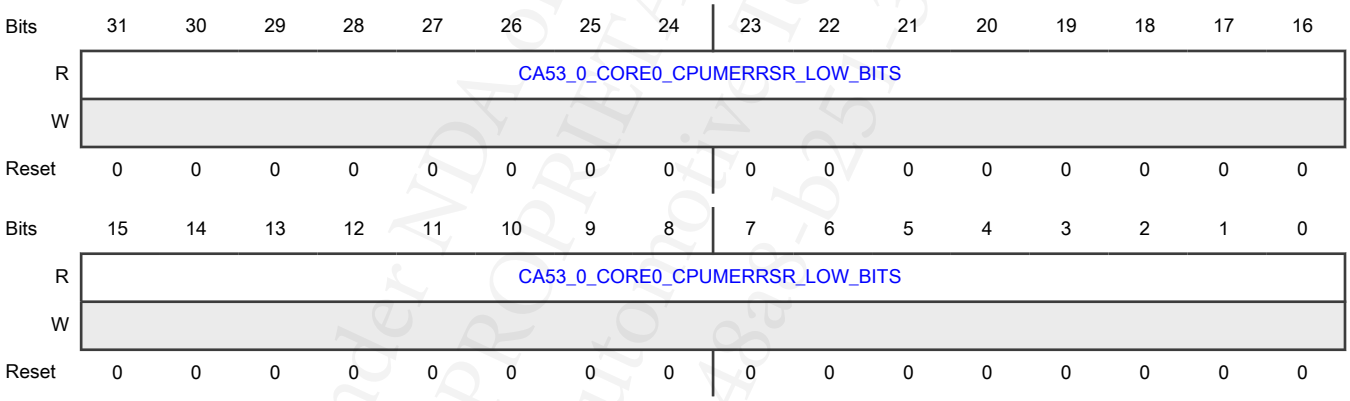
Field	Function
31-0 CA53_0_CORE0_CPUMERRSR_HIGH_BITS	Upper 32 bits Of CPUMERRSR Of Cortex-A53 Cluster0 Core0 Reflects the upper 32 bits of CPUMERRSR of Cortex-A53 Cluster0 Core0. For details of CPUMERRSR, see Technical reference. The values of the error count fields ("Repeat error count" and "Other error count") are not reliable and as long as these are nonzero, it should not be considered as significant. Use other register fields to take necessary actions in case of ECC errors.

10.3.9 Cluster0 Core0 CPUMERRSR Low (GPR11)

Offset

Register	Offset
GPR11	2Ch

Diagram



Fields

Field	Function
31-0 CA53_0_CORE0_CPUMERRSR_LOW_BITS	Lower 32 Bits Of CPUMERRSR Of Cortex-A53 Cluster0 Core0 Reflects the lower 32 bits of CPUMERRSR of Cortex-A53 Cluster0 Core0. For details of CPUMERRSR, see Technical reference.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462c2-1821-48a8-b25f-390d3ab624c1

10.3.10 Cluster0 Core1 CPUMERRSR High (GPR12)

Offset

Register	Offset
GPR12	30h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CA53_0_CORE1_CPUMERRSR_HIGH_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CA53_0_CORE1_CPUMERRSR_HIGH_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CA53_0_CORE1_CPUMERRSR_HIGH_BITS	Upper 32 bits Of CPUMERRSR Of Cortex-A53 Cluster0 Core1 Reflects the upper 32 bits of CPUMERRSR of Cortex-A53 Cluster0 Core1. For details of CPUMERRSR, see Technical reference. The values of the error count fields ("Repeat error count" and "Other error count") are not reliable and as long as these are nonzero, it should not be considered as significant. Use other register fields to take necessary actions in case of ECC errors.

10.3.11 Cluster0 Core1 CPUMERRSR Low (GPR13)

Offset

Register	Offset
GPR13	34h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-482-4a36-8251-390d34b624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CA53_0_CORE1_CPUMERRSR_LOW_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CA53_0_CORE1_CPUMERRSR_LOW_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CA53_0_CORE1_CPUMERRSR_LOW_BITS	Lower 32 bits Of CPUMERRSR Of Cortex-A53 Cluster0 Core1 Reflects the lower 32 bits of CPUMERRSR of Cortex-A53 Cluster0 Core1. For details of CPUMERRSR, see Technical reference.

10.3.12 Cluster1 Core0 CPUMERRSR High (GPR14)

Offset

Register	Offset
GPR14	38h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CA53_1_CORE0_CPUMERRSR_HIGH_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CA53_1_CORE0_CPUMERRSR_HIGH_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e1-4824-48a8-b251-390d3ab624c1

Fields

Field	Function
31-0 CA53_1_CORE0_CPUMERRSR_HIGH_BITS	Upper 32 bits Of CPUMERRSR Of Cortex-A53 Cluster1 Core0 Reflects the upper 32 bits of CPUMERRSR of Cortex-A53 Cluster1 Core0. For details of CPUMERRSR, see Technical reference. The values of the error count fields ("Repeat error count" and "Other error count") are not reliable and as long as these are nonzero, it should not be considered as significant. Use other register fields to take necessary actions in case of ECC errors.

10.3.13 Cluster1 Core0 CPUMERRSR Low (GPR15)

Offset

Register	Offset
GPR15	3Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CA53_1_CORE0_CPUMERRSR_LOW_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CA53_1_CORE0_CPUMERRSR_LOW_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CA53_1_CORE0_CPUMERRSR_LOW_BITS	Lower 32 bits Of CPUMERRSR Of Cortex-A53 Cluster1 Core0 Reflects the lower 32 bits of CPUMERRSR of Cortex-A53 Cluster1 Core0. For details of CPUMERRSR, see Technical reference.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462c-1821-48a8-b25-390d3ab624c1

10.3.14 Cluster1 Core1 CPUMERRSR High (GPR16)

Offset

Register	Offset
GPR16	40h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CA53_1_CORE1_CPUMERRSR_HIGH_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CA53_1_CORE1_CPUMERRSR_HIGH_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CA53_1_CORE1_CPUMERRSR_HIGH_BITS	Upper 32 bits Of CPUMERRSR Of Cortex-A53 Cluster1 Core1 Reflects the upper 32 bits of CPUMERRSR of Cortex-A53 Cluster1 Core1. For details of CPUMERRSR, see Technical reference. The values of the error count fields ("Repeat error count" and "Other error count") are not reliable and as long as these are nonzero, it should not be considered as significant. Use other register fields to take necessary actions in case of ECC errors.

10.3.15 Cluster1 Core1 CPUMERRSR Low (GPR17)

Offset

Register	Offset
GPR17	44h

Provided under NDA only
CONFIDENTIAL
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-482-4...-251-390d3...624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CA53_1_CORE1_CPUMERRSR_LOW_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CA53_1_CORE1_CPUMERRSR_LOW_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CA53_1_CORE1_CPUMERRSR_LOW_BITS	Lower 32 bits Of CPUMERRSR Of Cortex-A53 Cluster1 Core1 Reflects the lower 32 bits of CPUMERRSR of Cortex-A53 Cluster1 Core1. For details of CPUMERRSR, see Technical reference.

10.3.16 Cluster0 L2MERRSR High (GPR18)

Offset

Register	Offset
GPR18	48h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CA53_0_L2MERRSR_HIGH_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CA53_0_L2MERRSR_HIGH_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e1-4824-48a8-b251-390d3ab624c1

Fields

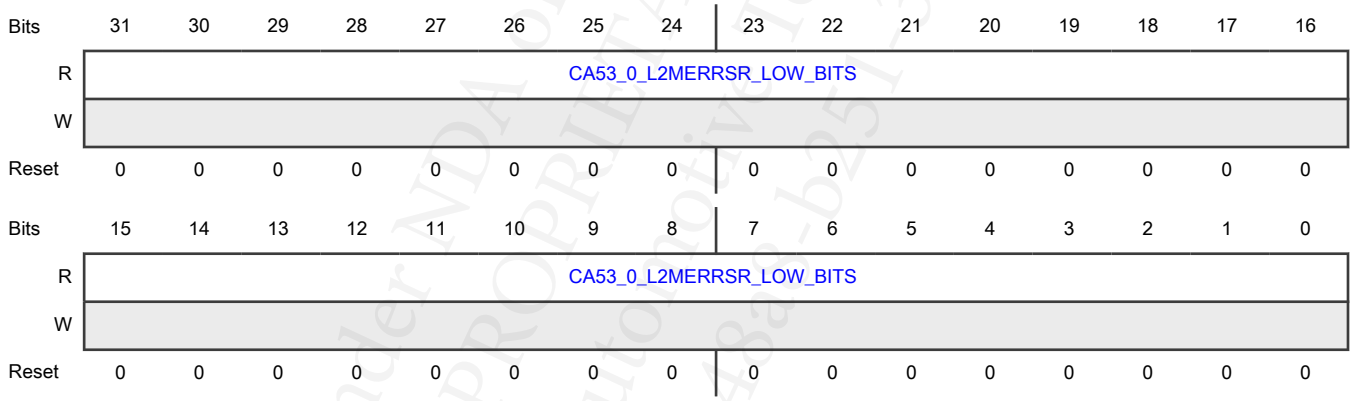
Field	Function
31-0 CA53_0_L2MERRSR_HIGH_BITS	Upper 32 Bits Of L2MERRSR Of Cortex-A53 Cluster0 Reflects the upper 32 bits of L2MERRSR of Cortex-A53 Cluster0. For details of L2MERRSR, see Technical reference. The values of the error count fields ("Repeat error count" and "Other error count") are not reliable and as long as these are nonzero, it should not be considered as significant. Use other register fields to take necessary actions in case of ECC errors.

10.3.17 Cluster0 L2MERRSR Low (GPR19)

Offset

Register	Offset
GPR19	4Ch

Diagram



Fields

Field	Function
31-0 CA53_0_L2MERRSR_LOW_BITS	Lower 32 Bits Of L2MERRSR of Cortex-A53 Cluster0 Reflects the lower 32 bits of L2MERRSR of Cortex-A53 Cluster0. For details of L2MERRSR, see Technical reference.

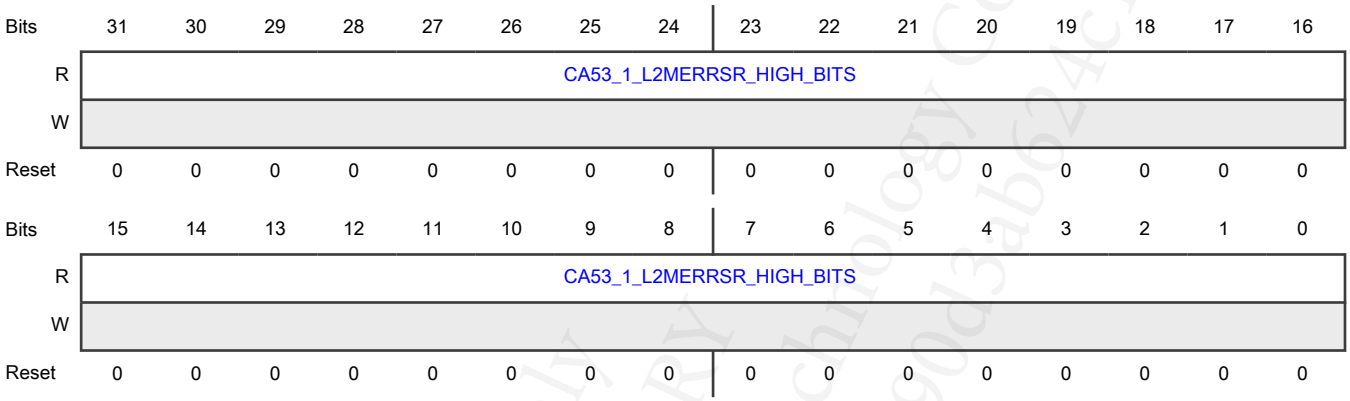
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462c3-821-48a-b251-390d3ab624c1

10.3.18 Cluster1 L2MERRSR High (GPR20)

Offset

Register	Offset
GPR20	50h

Diagram



Fields

Field	Function
31-0 CA53_1_L2MERRSR_HIGH_BITS	Upper 32 Bits Of L2MERRSR Of Cortex-A53 Cluster1 Reflects the upper 32 bits of L2MERRSR of Cortex-A53 Cluster1. For details of L2MERRSR, see Technical reference. The values of the error count fields ("Repeat error count" and "Other error count") are not reliable and as long as these are nonzero, it should not be considered as significant. Use other register fields to take necessary actions in case of ECC errors.

10.3.19 GPR21 (GPR21)

Offset

Register	Offset
GPR21	54h

Function

Cluster1 L2MERRSR Low

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-4838-8251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CA53_1_L2MERRSR_LOW_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CA53_1_L2MERRSR_LOW_BITS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CA53_1_L2MERRSR_LOW_BITS	Lower 32 Bits Of L2MERRSR Register of Cortex-A53 Cluster1 Reflects the lower 32 bits of L2MERRSR of Cortex-A53 Cluster1. For details of L2MERRSR, see Technical reference.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8ab624c1

Chapter 11
Cortex-M7 Cluster General Purpose Registers (CM7_GPR)

11.1 Introduction

CM7_CLUSTER_GPR configures certain Cortex-M7 features.

11.2 CM7_CLUSTER_GPR register descriptions

11.2.1 CM7_CLUSTER_GPR memory map

CM7_GPR_0 base address: 4007_C100h

CM7_GPR_1 base address: 4007_C200h

CM7_GPR_2 base address: 4007_C300h

Offset	Register	Width (In bits)	Access	Reset value
0h	Cortex-M7 GPR0 (CORTEXM7_GPR0)	32	RW	0000_0000h

11.2.2 Cortex-M7 GPR0 (CORTEXM7_GPR0)

Offset

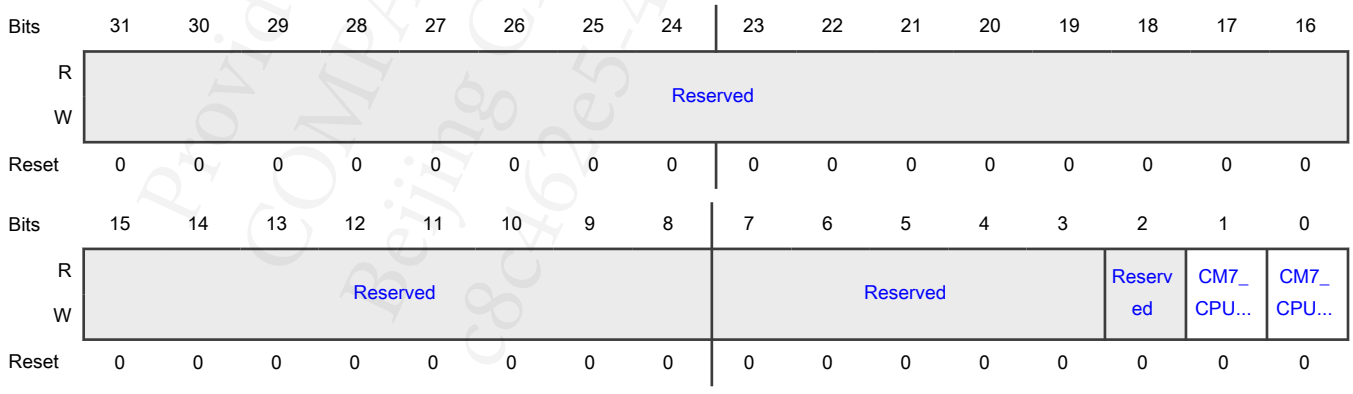
Register	Offset
CORTEXM7_GPR0	0h

Function

Configures Cortex-M7 memory retention, wait assertion, and WFE behavior.

You must access this register using only 32-bit accesses.

Diagram



Fields

Field	Function
31-8 —	Reserved. Writes to this field result in transfer error.
7-3 —	Reserved
2 —	Reserved
1 CM7_CPU_EVENT_GENERATE	Cortex-M7 Event Generate Sets the Event register in the processor that is defined in the Arm v7-M architecture. This causes a WFE instruction to complete. It also wakes up the processor, if it is sleeping, because it executed a WFE instruction. 0b - Do not set event 1b - Set event
0 CM7_CPU_WAIT	Cortex-M7 CPU Wait Asserts wait after reset during TCM initialization. See "Preloading TCM" section in Arm Cortex-M7 Processor Technical Reference Manual, DDI 0489D Revision r1p1 at www.arm.com. 0b - Do not assert wait 1b - Assert wait

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3a6624c1

Chapter 12
Enhanced Direct Memory Access (eDMA)

12.1 Chip-specific eDMA information

12.1.1 eDMA instances

This device has the following system eDMA instances:

- eDMA_0
- eDMA_1

Each of these instances also has a shadow instance tied to it in delayed lockstep and CRC is integrated with each of them.

As the following figure shows:

- The system eDMA instances, eDMA_0 and eDMA_1, each have a "shadow" eDMA instance that operates in delayed lockstep. All four of these instances have 32 channels.
- The delayed-lockstep pair eDMA_0 and its "shadow" instance integrate with DMAMUX_0 and DMAMUX_1.
- The delayed-lockstep pair eDMA_1 and its "shadow" instance integrate with DMAMUX_2 and DMAMUX_3.

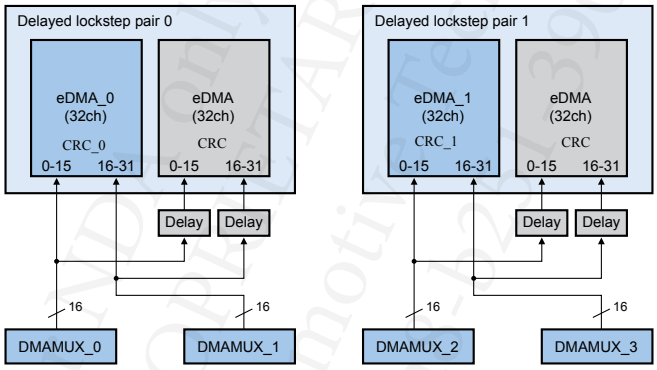


Figure 20. eDMA instances and DMAMUX integration

12.1.2 ECC support

TCD memory is covered by 8-bit ECC on a 64-bit basis using the H-matrix that includes address in the syndrome calculation, which enables single-error correction, double-error detection (SECDED), and allows for error injection.

12.1.3 eDMA bandwidth management

The eDMA bandwidth is shared across peripherals, limiting how much bandwidth the eDMA can support simultaneously on all peripherals. For example, if all four SPI modules are operating together in full-duplex mode with 8-bit frame size, they might get only up to 4 Mbit/s per SPI module for transmit/receive. Therefore, to increase the efficiency of high-bandwidth SPI transfers, use larger transfer frame sizes.

12.2 Introduction

The enhanced direct memory access (eDMA) controller is capable of performing complex data transfers with minimal intervention from a host processor. The hardware microarchitecture includes:

- A DMA engine that performs:
  - Source address and destination address calculations

— Data-movement operations

- Local memory containing transfer control descriptors for each of the 32 channels

12.2.1 eDMA system block diagram

Figure 21 illustrates the components of the eDMA system, including the eDMA module (engine).

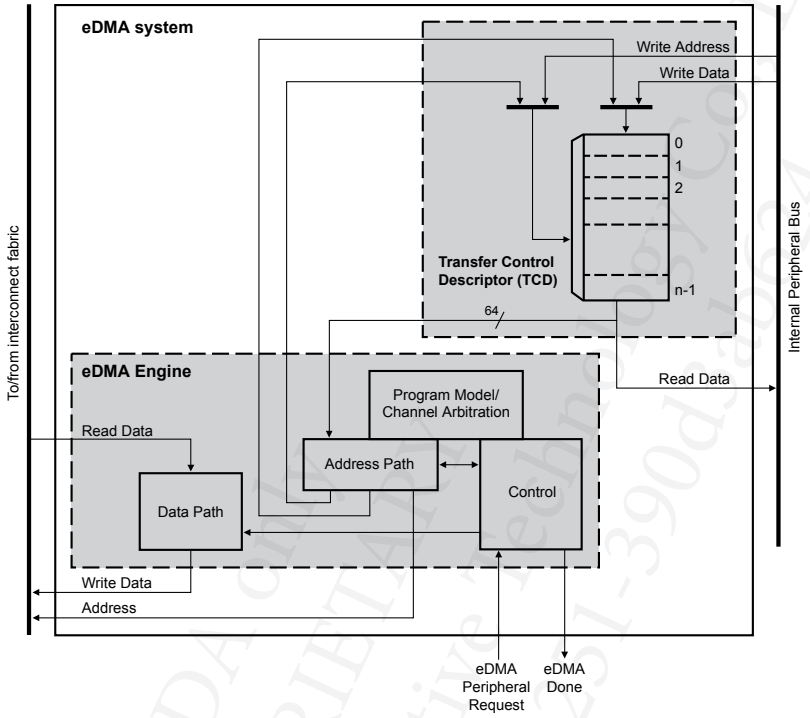


Figure 21. eDMA system block diagram

12.2.2 Block parts

The eDMA module is partitioned into two major modules: the eDMA engine and the transfer control descriptor local memory.

The eDMA engine is further partitioned into four submodules:

Table 39. eDMA engine submodules

Submodule	Function
Address path	This block: • Implements a primary channel and secondary (preempt) channel • Manages all master bus-address calculations All the channels provide the same functionality. This structure allows data transfers associated with one channel to be preempted after the completion of a read/write sequence if a higher priority channel activation is asserted while the primary channel is active. After a channel is activated, it runs until the minor loop is completed, unless preempted by a higher priority channel. This provides a mechanism (enabled by CH*n*_PRI[ECP]) where a large data transfer can be preempted to minimize the time another channel is blocked from execution.

Table continues on the next page...

Table 39. eDMA engine submodules (continued)

Submodule	Function
	When any channel is selected to execute, the contents of its TCD are read from local memory and loaded into the address path channel x registers for a normal start and into channel y registers for a preemption start. After the minor loop completes execution, the address path hardware writes the new values for the TCD*n*_{SADDR, DADDR, CITER} back to local memory. If the major iteration count is exhausted, additional processing is performed, including the final address pointer updates, reloading the TCD*n*_CITER field, and a possible fetch of a new TCD*n* from memory as part of a scatter/gather operation. See Dynamic scatter/gather for more details.
Data path	This block implements the bus master read/write data path. It includes a data buffer and the necessary multiplex logic to support any required data alignment. The internal read data bus is the primary input, and the internal write data bus is the primary output. The address and data path modules directly support the 2-stage pipelined internal bus. The address path module represents the first stage of the bus pipeline (address phase), and the data path module implements the second stage of the pipeline (data phase).
Program model/channel arbitration	This block implements the first section of the eDMA programming model as well as the channel arbitration logic. The programming model registers are connected to the internal peripheral bus. The eDMA peripheral request inputs and interrupt request outputs are also connected to this block (via control logic).
Control	This block provides all the control functions for the eDMA engine. For data transfers where the source and destination sizes are equal, the eDMA engine performs a series of source read/destination write operations until the number of bytes specified in the minor loop byte count has been moved from the source to the destination. For descriptors where the sizes are not equal, multiple accesses of the smaller size data are required for each reference of the larger size. As an example, if the source size references 16-bit data and the destination is 32-bit data, the eDMA performs two reads, then one 32-bit write.

The transfer control descriptor local memory is further partitioned into:

Table 40. Transfer control descriptor memory

Submodule	Description
Memory controller	This logic implements the required dual-ported controller, and manages accesses from the eDMA engine as well as references from the internal peripheral bus. As noted earlier, in simultaneous accesses, the eDMA engine is given priority and the peripheral transaction is stalled.
Memory array	TCD storage for each channel's transfer profile.

12.2.3 Features

The eDMA is a highly programmable data-transfer engine optimized to minimize any required intervention from the host processor. It is intended for use in applications where the data size to be transferred is statically known and not defined within the transferred data itself. The eDMA module features:

- All data movement via dual-address transfers: read from source, write to destination
  - Programmable source and destination addresses and transfer size
  - Support for complex address calculations
- 32-channel implementation that performs complex data transfers with minimal intervention from a host processor
  - Internal data buffer, used as temporary storage for all transfers

  - Connections to the crossbar switch for bus mastering the data movement
- TCD organized to support two-deep, nested transfer operations
  - 32-byte TCD stored in local memory for each channel
  - An inner data transfer loop defined by a minor byte transfer count
  - An outer data transfer loop defined by a major iteration count
- Channel activation via one of three methods:
  - Explicit software initiation
  - Initiation via a channel-to-channel linking mechanism for continuous transfers
  - Peripheral-paced hardware requests, one per channel
- Fixed-priority and round-robin channel arbitration
- Channel completion reported via programmable interrupt requests
  - One interrupt per channel, which can be asserted at completion of major iteration count
  - Programmable error terminations per channel that are logically summed together to form one error interrupt to the interrupt controller
- Programmable support for scatter/gather DMA processing
- Support for complex data structures

In the discussion of this module, *n* is used to reference the channel number.

12.3 Modes of operation

The eDMA operates in the following modes:

Table 41. Modes of operation

Mode	Description
Normal	In Normal mode, eDMA transfers data between a source and a destination. The source and destination can be a memory block or an I/O block capable of operation with eDMA. A service request initiates a transfer of a specific number of bytes (NBYTES) as specified in the TCD. The minor loop is the sequence of read-write operations that transfers these NBYTES per service request. Each service request executes one iteration of the major loop, which transfers NBYTES of data.
Debug	eDMA operation is configurable in Debug mode via the control register: • If CSR[EDBG] is cleared to 0, eDMA continues to operate. • If CSR[EDBG] is set to 1, eDMA stops transferring data. If Debug mode is entered when a channel is active, eDMA continues operation until the channel retires.
Wait	Before entering Wait mode, the eDMA attempts to complete an active channel. After the channel completes, no other channels are allowed to start and the device enters Wait mode.

12.4 Memory map/register definition

The eDMA programming model is partitioned into three parts:

1. The first part defines a number of registers providing overall control functions and is known as the management page.
2. The second part corresponds to the channel (CH) control, status, and configuration.

3. The third part corresponds to the local TCD memory.

TCD memory

Each channel requires a 32-byte transfer control descriptor for defining the data movement operation. Each TCDn definition is presented as 11 registers of 16 or 32 bits.

TCD initialization

The TCD memory is in an unknown state after reset. Only the TCD START bit is initialized to 0. Prior to activating a channel, you must initialize its TCD with the appropriate transfer profile.

TCD structure

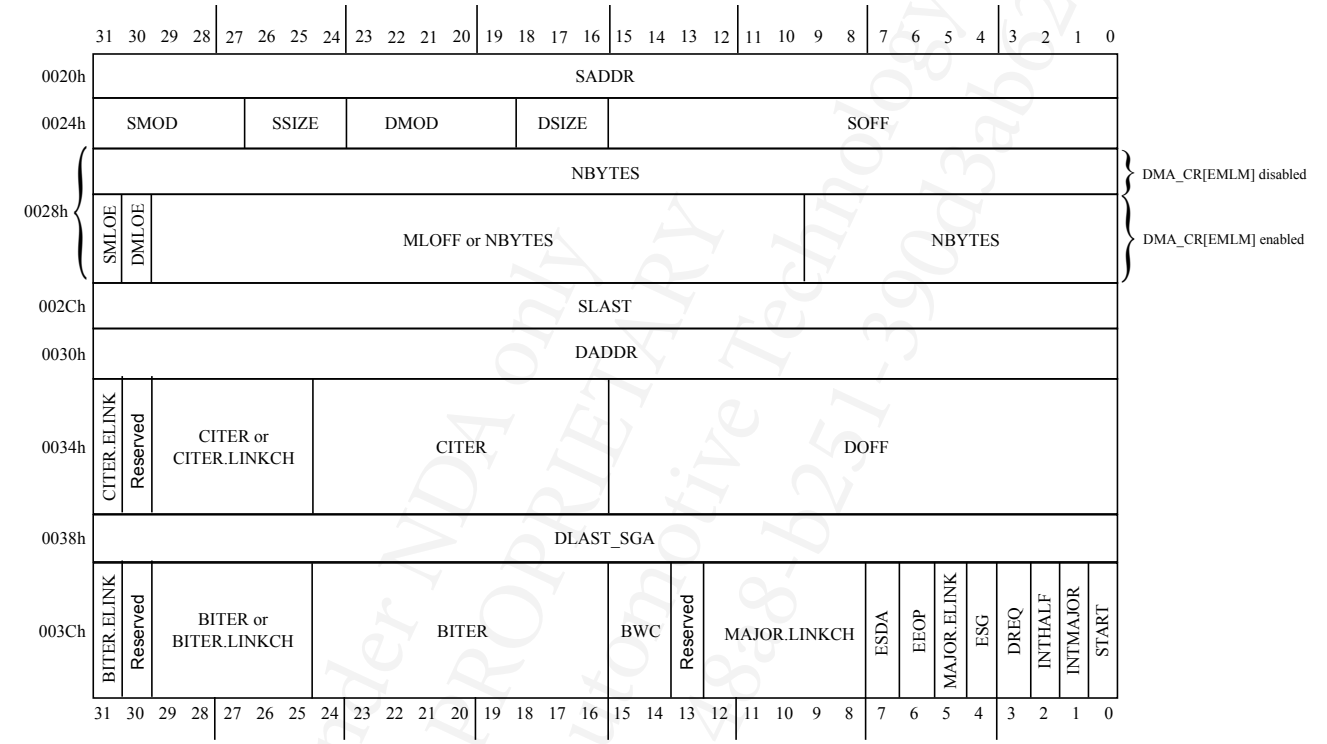


Figure 22. TCD structure

Accesses to reserved memory and fields

- Reading reserved fields in a register returns the value of zero.
- Writes to reserved fields in a register are ignored.
- Reading or writing a reserved memory location generates a bus error.

12.4.1 DMA MP register descriptions

12.4.1.1 MP memory map

eDMA_0_MP base address: 4014_4000h

eDMA_1_MP base address: 4024_4000h

Processed under NDA only
CONFIDENTIAL PROPRIETARY
Reijns@HJ Automotive Technology Co. Ltd
c8c46265-1821-48a8-b251-990a3ab024c1

Offset	Register	Width (In bits)	Access	Reset value
0h	Management Page Control (CSR)	32	RW	0030_0000h
4h	Management Page Error Status (ES)	32	RO	0000_0000h
8h	Management Page Interrupt Request Status (INT)	32	RO	0000_0000h
Ch	Management Page Hardware Request Status (HRS)	32	RO	0000_0000h
100h - 17Ch	Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI)	32	RW	0000_0000h

12.4.1.2 Management Page Control (CSR)

Offset

Register	Offset
CSR	0h

Function

The Management Page Control register defines the basic operating configuration of the DMA.

Arbitration uses a two-tier priority system; group and channel priority. The eDMA assigns each channel to a priority group. Group arbitration is fixed-priority and cannot be changed. Channel arbitration uses fixed priority and may be configured to use a selective round-robin scheme for specified channels within each priority group. For fixed-priority arbitration, eDMA selects for execution the highest priority channel requesting service in the highest priority arbitration group.

The channel priority registers assign the relative priorities within each arbitration group; see CHn_PRI. All channels with a non-zero CHn_PRI value use fixed-priority arbitration.

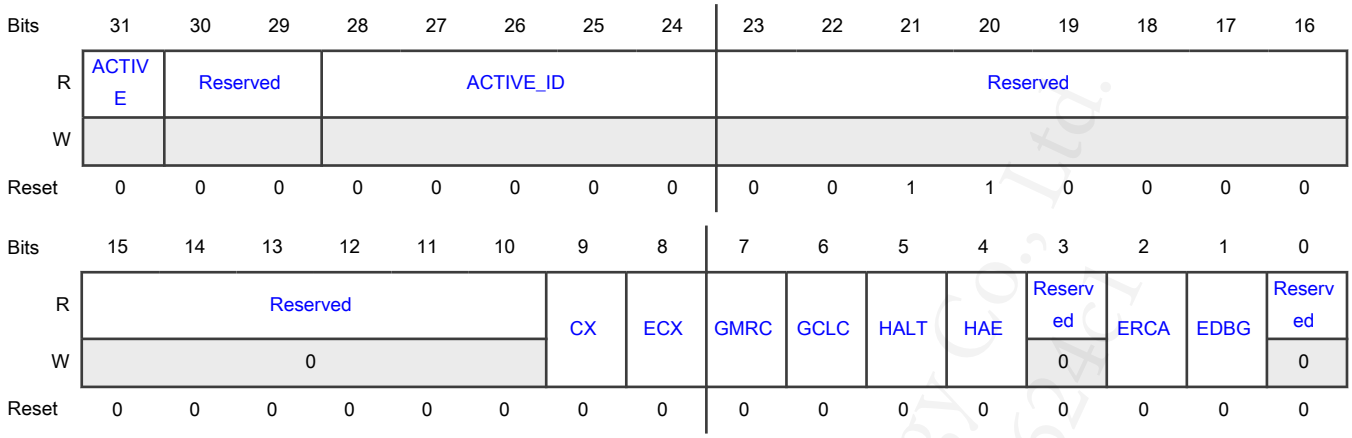
When you enable round-robin arbitration, all channels with channel priority set to zero do not have a priority and, of those channels requesting service, are cycled through (from high to low channel number) without regard to priority relative to each other within the same priority group. Any channel with a non-zero CHn_PRI value automatically has a higher priority over the round-robin channels. A channel's priority group is assigned in Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI).

NOTE

For correct operation, changes to the CSR[ERCA, GCLC, GMRC] fields must be performed when the DMA channels are inactive; that is, when the CSR[ACTIVE] field is 0.

Provided under NXP only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-487c-4258-9f-390d3ab624

Diagram



Fields

Field	Function
31 ACTIVE	DMA Active Status 0b - eDMA is idle 1b - eDMA is executing a channel
30-29 —	Reserved
28-24 ACTIVE_ID	Active Channel ID This field identifies the channel number that is executing when the ACTIVE bit is 1.
23-16 —	Reserved
15-10 —	Reserved
9 CX	Cancel Transfer When set to 1, this field cancels the remaining data transfer, stops the executing channel, and forces the minor loop to finish. The cancel takes effect after the last write of the current read/write sequence. CX clears itself to 0 after the cancel has been honored. This cancel retires the channel normally as if the minor loop had been completed. 0b - Normal operation 1b - Cancel the remaining data transfer
8 ECX	Cancel Transfer With Error Cancellation of the remaining data transfer is similar to that of the CX field. Execution of the the channel is stopped and the minor loop is forced to finish. The cancellation takes effect after the last write of the current read/write sequence. The ECX field clears itself to 0 after the cancel is honored. In addition to

Table continues on the next page...

Table continued from the previous page...

Field	Function
	cancelling the transfer, ECX treats the cancel as an error condition, thus updating Management Page Error Status (ES) and generating an optional error interrupt. 0b - Normal operation 1b - Cancel the remaining data transfer
7 GMRC	Global Master ID Replication Control **NOTE** If master ID replication is disabled, the privileged protection level (Supervisor mode) for DMA transfers is used. 0b - Master ID replication disabled for all channels 1b - Master ID replication available and controlled by each channel's CHn_SBR[EMI] setting
6 GCLC	Global Channel Linking Control 0b - Channel linking disabled for all channels 1b - Channel linking available and controlled by each channel's link settings
5 HALT	Halt DMA Operations This field stalls the start of any new channels. Executing channels are allowed to complete. Channel execution resumes when this field is cleared to 0. 0b - Normal operation 1b - Stall the start of any new channels
4 HAE	Halt After Error When this field is set to 1, any error causes the HALT field to be set to 1. Then all service requests are ignored until the HALT field is cleared to 0. 0b - Normal operation 1b - Any error causes the HALT field to be set to 1
3 —	Reserved
2 ERCA	Enable Round Robin Channel Arbitration 0b - Fixed priority arbitration used for channel selection within each group 1b - Round-robin arbitration used for channel selection within each group
1 EDBG	Enable Debug When in debug mode, the DMA stalls the start of a new channel. Executing channels are allowed to complete. DMA resumes channel execution when the system exits debug mode or clears the EDBG field to 0. 0b - When in debug mode, the DMA continues to operate

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - When in debug mode, the DMA stalls the start of a new channel
0 —	Reserved

12.4.1.3 Management Page Error Status (ES)

Offset

Register	Offset
ES	4h

Function

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- An illegal setting in the transfer control descriptor
- An error termination to a bus master read or write cycle
- An uncorrectable error that occurred when the device was accessing the TCD SRAM
- A "cancel transfer with error" request was made via the corresponding cancel transfer field or input signal

Upon any error condition, the software must initialize the TCD of the channel that contains the error, as it is in an incomplete state after an error. See Fault reporting and handling for more details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	Reserved		ERRCHN					0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						UCE	ECX	SAE	SOE	DAE	DOE	NCE	SGE	SBE	DBE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Valid Logical OR of all ERR status fields.

Table continues on the next page...

Table continued from the previous page...

Field	Function
VLD	0b - No ERR fields are set to 1 1b - At least one ERR field is set to 1, indicating a valid error exists that software has not cleared
30-29 —	Reserved
28-24 ERRCHN	Error Channel Number or Canceled Channel Number The channel number of the last recorded error or last recorded error-canceled transfer.
23-10 —	Reserved
9 UCE	Uncorrectable TCD Error During Channel Execution UCE is set to 1 only when an uncorrectable ECC error occurs on an access generated by the DMA. If a CPU access to the TCD causes an uncorrectable ECC error, then that access receives a bus error response. **NOTE** When the eDMA sees a RAM error on an IPS access (when you are accessing a TCD), it reports the error as a bus abort. When the DMA engine receives a RAM error (the execution engine is accessing a TCD) it is recorded in the Error Status register, ES[UCE], along with the channel number. 0b - No uncorrectable ECC error 1b - Last recorded error was an uncorrectable TCD RAM error
8 ECX	Transfer Canceled The ECX operation is a management page function. When employed, the targeted channel's CHn_ES register reports an unspecified error; that is, only the ERR field is set to 1. The management page has full view of the error condition. 0b - No canceled transfers 1b - Last recorded entry was a canceled transfer by the error cancel transfer input
7 SAE	Source Address Error When this field is 1, it indicates that TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SADDR field
6 SOE	Source Offset Error When this field is 1, it indicates that TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SOFF field
5	Destination Address Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
DAE	When this field is 1, it indicates that TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DADDR field
4 DOE	Destination Offset Error When this field is 1, it indicates that TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DOFF field
3 NCE	NBYTES/CITER Configuration Error This error indicates that one of the following has occurred: • TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE] • TCDn_CITER[CITER] is equal to zero • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK] 0b - No NBYTES/CITER configuration error 1b - Last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields
2 SGE	Scatter/Gather Configuration Error When this field is 1, it indicates that TCDn_DLAST_SGA is not on a 32-byte boundary. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG] is enabled. 0b - No scatter/gather configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DLAST_SGA field
1 SBE	Source Bus Error 0b - No source bus error 1b - Last recorded error was a bus error on a source read
0 DBE	Destination Bus Error 0b - No destination bus error 1b - Last recorded error was a bus error on a destination write

12.4.1.4 Management Page Interrupt Request Status (INT)

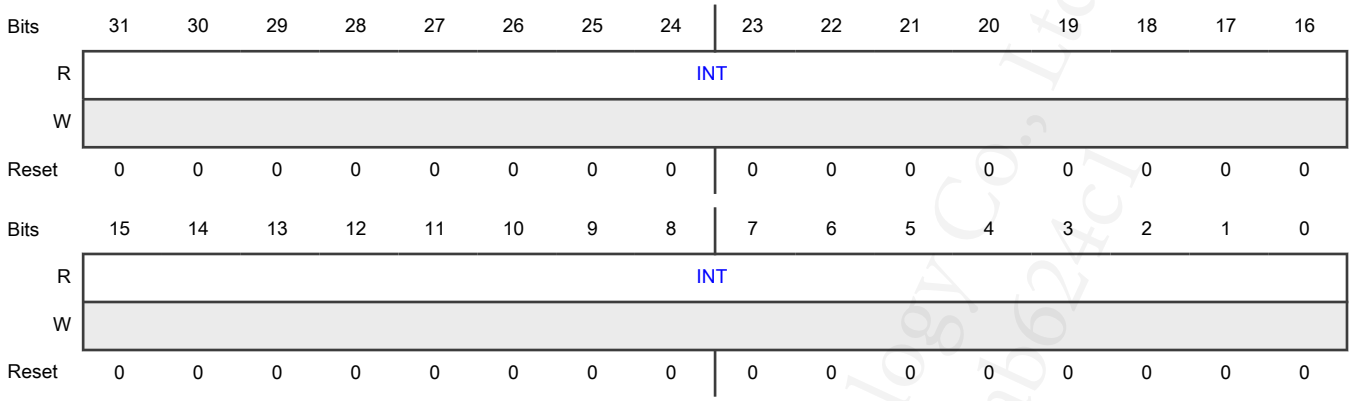
Offset

Register	Offset
INT	8h

Function

This register shows the current state of the interrupt service requests for all eDMA channels.

Diagram



Fields

Field	Function
31-0 INT	Interrupt Request Status The INT register presents the interrupt request status for each eDMA channel. Depending on the appropriate field setting in the transfer control descriptors, the eDMA engine generates an interrupt on data transfer completion. The eDMA routes channel interrupt requests to the interrupt controller. During the interrupt service routine associated with any given channel, it is the software's responsibility to clear the appropriate field in the channel's interrupt request register, CHn_INT, thus negating the interrupt request. 0b - Interrupt request for corresponding channel not present 1b - Interrupt request for corresponding channel present

12.4.1.5 Management Page Hardware Request Status (HRS)

Offset

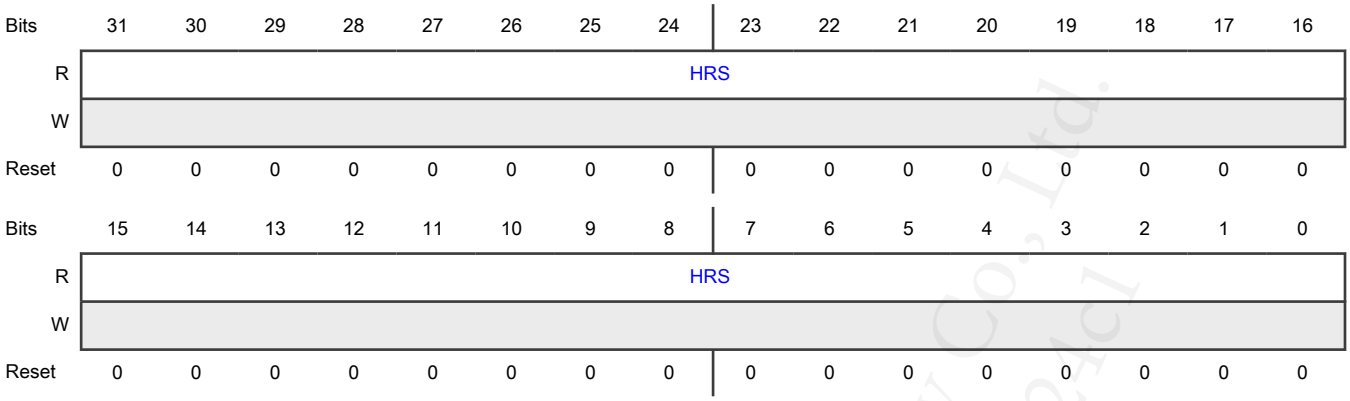
Register	Offset
HRS	Ch

Function

The hardware request status register (HRS) shows the current state of the hardware service request signaling as seen by eDMA's arbitration logic.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b2f1-390d3ab624c1

Diagram



Fields

Field	Function
31-0 HRS	Hardware Request Status The HRS bit for its respective channel remains asserted for the period when a hardware request is present on the channel. After the request is completed and the channel is free, the hardware automatically clears the corresponding HRS bit to 0. 0b - Hardware service request for corresponding channel is not present 1b - Hardware service request for corresponding channel is present

12.4.1.6 Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI)

Offset

For n = 0 to 31:

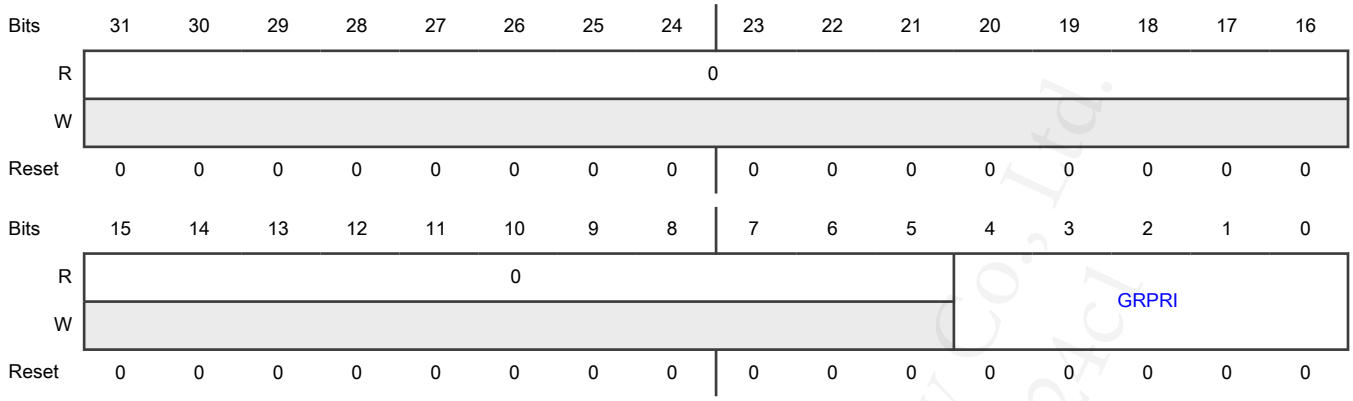
Register	Offset
CHn_GRPRI	100h + (n × 4h)

Function

The contents of this register define the arbitration group associated with each channel. Using a fixed-priority group arbitration scheme, eDMA evaluates the arbitration group priorities by numeric value from highest group number to lowest; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, and so on. The range of the group priority values is limited to the values of 0 through 31. Within each arbitration group, the channel priority assignment CH*n*_PRI determines the highest-priority channel.

Provided under NDA
COMPANY PROPRIETARY
Beijing ECU Automotive Technology Co., Ltd.
c8c462e3-821-48a8-b25-39d8ab624c1

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 GRPRI	Arbitration Group For Channel n Fixed-priority arbitration group number.

12.4.2 DMA TCD register descriptions

12.4.2.1 TCD memory map

eDMA_0_TCD base address: 4014_8000h

eDMA_1_TCD base address: 4024_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 1_F000h	Channel Control and Status (CH0_CSR - CH31_CSR)	32	RW	0000_0000h
4h - 1_F004h	Channel Error Status (CH0_ES - CH31_ES)	32	W1C	0000_0000h
8h - 1_F008h	Channel Interrupt Status (CH0_INT - CH31_INT)	32	W1C	0000_0000h
Ch - 1_F00Ch	Channel System Bus (CH0_SBR - CH31_SBR)	32	RW	See description
10h - 1_F010h	Channel Priority (CH0_PRI - CH31_PRI)	32	RW	0000_0000h
20h - 1_F020h	TCD Source Address (TCD0_SADDR - TCD31_SADDR)	32	RW	See description
24h - 1_F024h	TCD Signed Source Address Offset (TCD0_SOFF - TCD31_SOFF)	16	RW	See description

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
26h - 1_F026h	TCD Transfer Attributes (TCD0_ATTR - TCD31_ATTR)	16	RW	See description
28h - 1_F028h	TCD Transfer Size Without Minor Loop Offsets (TCD0_NBYTES_MLOFFNO - TCD31_NBYTES_MLOFFNO)	32	RW	See description
28h - 1_F028h	TCD Transfer Size with Minor Loop Offsets (TCD0_NBYTES_MLOFFYES - TCD31_NBYTES_MLOFFYES)	32	RW	See description
2Ch - 1_F02Ch	TCD Last Source Address Adjustment / Store DADDR Address (TCD0_SLAST_SDA - TCD31_SLAST_SDA)	32	RW	See description
30h - 1_F030h	TCD Destination Address (TCD0_DADDR - TCD31_DADDR)	32	RW	See description
34h - 1_F034h	TCD Signed Destination Address Offset (TCD0_DOFF - TCD31_DOFF)	16	RW	See description
36h - 1_F036h	TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD31_CITER_ELINKNO)	16	RW	See description
36h - 1_F036h	TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD31_CITER_ELINKYES)	16	RW	See description
38h - 1_F038h	TCD Last Destination Address Adjustment / Scatter Gather Address (TCD0_DLAST_SGA - TCD31_DLAST_SGA)	32	RW	See description
3Ch - 1_F03Ch	TCD Control and Status (TCD0_CSR - TCD31_CSR)	16	RW	See description
3Eh - 1_F03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD31_BITER_ELINKNO)	16	RW	See description
3Eh - 1_F03Eh	TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD31_BITER_ELINKYES)	16	RW	See description

12.4.2.2 Channel Control and Status (CH0_CSR - CH31_CSR)

Offset

For n = 0 to 31:

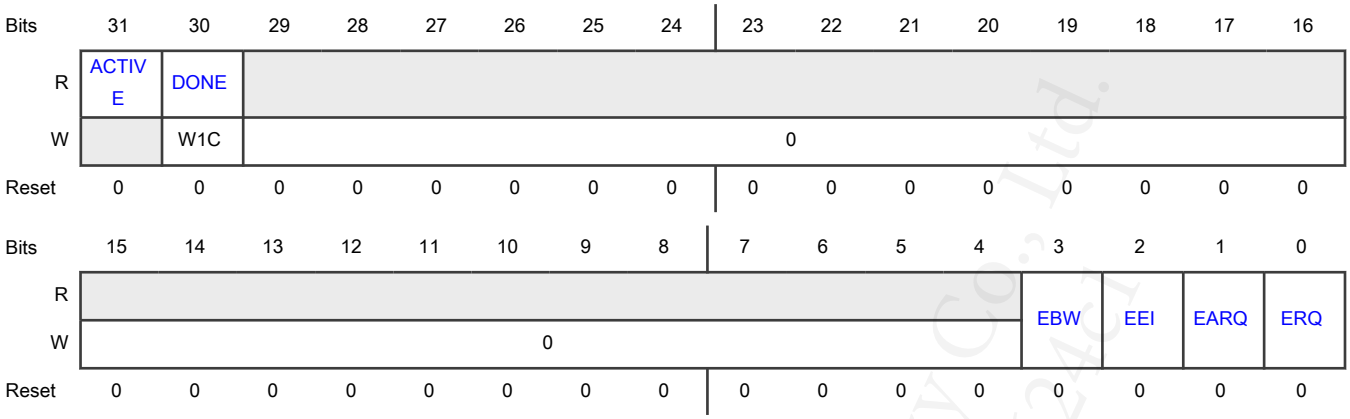
Register	Offset
CHn_CSR	0h + (n × 1000h)

Function

This register contains several fields related to hardware and interrupt requests, configuration, and status for the given channel.

Provided under NXP COMPANY CONFIDENTIAL Beijing CHJ Automotive Technology Co., Ltd. c84625-482-4a-b251-290da6741

Diagram



Fields

Field	Function
31 ACTIVE	Channel Active The ACTIVE field indicates the channel was selected by arbitration and is executing the prescribed transfers. The eDMA sets it to 1 when channel service begins, and clears it to 0 as the minor loop completes or when any error condition is detected. Except for dynamic scatter/gather or dynamic channel linking, you must not modify the transfer control descriptor when a channel is active.
30 DONE	Channel Done The DONE field indicates the eDMA has completed the major loop. The eDMA engine sets this field as the CITER count reaches zero. If enabled, the eDMA generates an interrupt request corresponding to this completed channel. The software clears it, or the hardware clears it when the channel is activated. **NOTE** This field must be cleared to 0 before writing the MAJORELINK or ESG fields.
29-4 —	Reserved
3 EBW	Enable Buffered Writes When buffered writes are enabled, all writes except for the last write sequence of the minor loop are signaled by the eDMA as bufferable. 0b - Buffered writes on system bus disabled 1b - Bufferable write signal asserted on all system bus writes except during last write sequence
2 EEI	Enable Error Interrupt The EEI field enables the error interrupt signal for the channel. The DMA error indicator and the error interrupt enable flag must be asserted before an error interrupt request for a given channel is asserted to the interrupt controller. 0b - Error signal for corresponding channel does not generate error interrupt 1b - Assertion of error signal for corresponding channel generates error interrupt request

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 EARQ	Enable Asynchronous DMA Request In Stop Mode For Channel The enable asynchronous DMA request field (EARQ) does not affect DMA operations. When set to 1, this field allows the hardware service request enable field (ERQ) to propagate out of the DMA to the power controller. When cleared to 0, this field masks the hardware service request enable field to the power controller. 0b - Disable asynchronous DMA request for the channel 1b - Enable asynchronous DMA request for the channel
0 ERQ	Enable DMA Request Disable a channel's hardware service request at the source before clearing the channel's ERQ field. The DMA hardware request input signal and the enable request field (ERQ) must be asserted before a channel's hardware service request is accepted. The state of the eDMA enable request field does not affect a channel service request made explicitly through software or channel linking. The state of the ERQ field does not affect the channel's START field. 0b - DMA hardware request signal for corresponding channel disabled 1b - DMA hardware request signal for corresponding channel enabled

12.4.2.3 Channel Error Status (CH0_ES - CH31_ES)

Offset

For n = 0 to 31:

Register	Offset
CHn_ES	4h + (n × 1000h)

Function

The ES provides information concerning the last recorded channel error. Channel errors can be caused by:

- An illegal setting in the transfer control descriptor
- An error termination to a bus master read or write cycle

The ERR field signals the presence of an error for the channel. The eDMA engine signals the occurrence of an error condition by setting the appropriate field in this register. The outputs of this register are enabled by the contents of the CHn_CSR[EEI] field, then logically summed across all channels to form an error interrupt request, which may be routed to the interrupt controller.

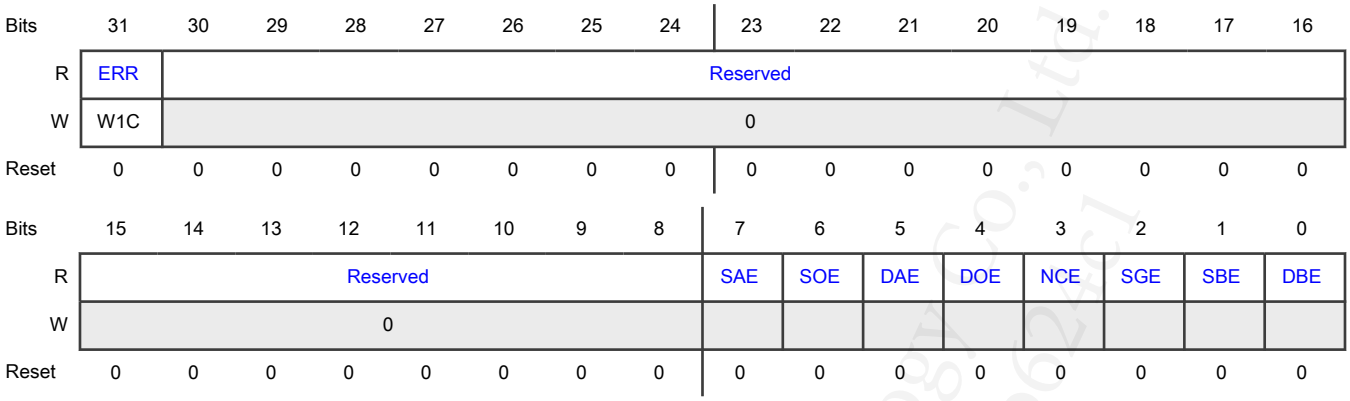
During the execution of the interrupt service routine associated with any DMA errors, it is software's responsibility to clear the appropriate bit, negating the error-interrupt request. The normal DMA channel completion indicators (setting the transfer control descriptor DONE flag and the possible assertion of an interrupt request) are not affected when eDMA detects an error. The contents of this ERR register field can also be polled because a non-zero value indicates the presence of a channel error, regardless of the state of the EEI mask.

The state of any given channel's error indicators is affected by writes to this register. Writing a 1 to the ERR field clears the channel's error status, and writing a 0 has no effect.

An unspecified error, where only the ERR field is set to 1, indicates that either a transfer was cancelled with an error or else an uncorrectable TCD error occurred. The Management Page Error Status register has full view of the error condition.

See Fault reporting and handling for more details.

Diagram



Fields

Field	Function
31 ERR	Error In Channel 0b - An error in this channel has not occurred 1b - An error in this channel has occurred
30-8 —	Reserved
7 SAE	Source Address Error TCDn_SADDR is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SADDR field
6 SOE	Source Offset Error TCDn_SOFF is inconsistent with TCDn_ATTR[SSIZE]. 0b - No source offset configuration error 1b - Last recorded error was a configuration error detected in the TCDn_SOFF field
5 DAE	Destination Address Error TCDn_DADDR is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination address configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DADDR field
4 DOE	Destination Offset Error TCDn_DOFF is inconsistent with TCDn_ATTR[DSIZE]. 0b - No destination offset configuration error

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Last recorded error was a configuration error detected in the TCDn_DOFF field
3 NCE	NBYTES/CITER Configuration Error This error indicates that one of the following has occurred: • TCDn_NBYTES is not a multiple of TCDn_ATTR[SSIZE] and TCDn_ATTR[DSIZE] • TCDn_CITER[CITER] is equal to zero • TCDn_CITER[ELINK] is not equal to TCDn_BITER[ELINK] 0b - No NBYTES/CITER configuration error 1b - Last recorded error was a configuration error detected in the TCDn_NBYTES or TCDn_CITER fields
2 SGE	Scatter/Gather Configuration Error When this field is 1, it indicates that TCDn_DLAST_SGA is not on a 32-byte boundary. This field is checked at the beginning of a scatter/gather operation after major loop completion if TCDn_CSR[ESG]is enabled. 0b - No scatter/gather configuration error 1b - Last recorded error was a configuration error detected in the TCDn_DLAST_SGA field
1 SBE	Source Bus Error 0b - No source bus error 1b - Last recorded error was bus error on source read
0 DBE	Destination Bus Error 0b - No destination bus error 1b - Last recorded error was bus error on destination write

12.4.2.4 Channel Interrupt Status (CH0_INT - CH31_INT)

Offset

For n = 0 to 31:

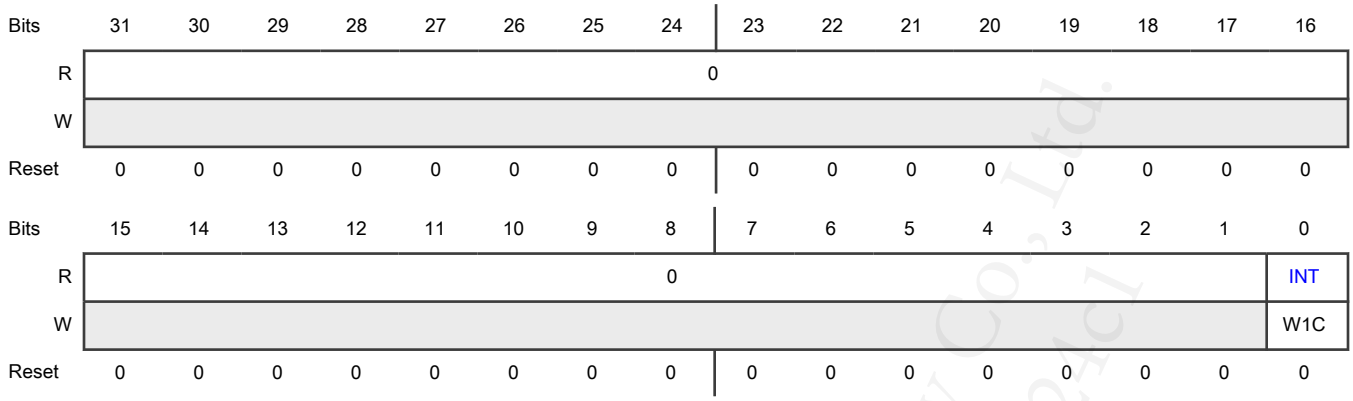
Register	Offset
CHn_INT	8h + (n × 1000h)

Function

The INT field signals the presence of an interrupt request for the channel. Depending on the appropriate bit setting in the transfe control descriptors, the eDMA engine generates an interrupt on data transfer completion.

The outputs of this register are directly routed to the interrupt controller. During the interrupt service routine associated with any given channel, it is the software's responsibility to clear the appropriate bit, negating the interrupt request. On writes to INT, a 1 clears the channel's interrupt request. A zero has no effect on the channel's current interrupt status.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 INT	Interrupt Request 0b - Interrupt request for corresponding channel cleared 1b - Interrupt request for corresponding channel active

12.4.2.5 Channel System Bus (CH0_SBR - CH31_SBR)

Offset

For n = 0 to 31:

Register	Offset
CHn_SBR	Ch + (n × 1000h)

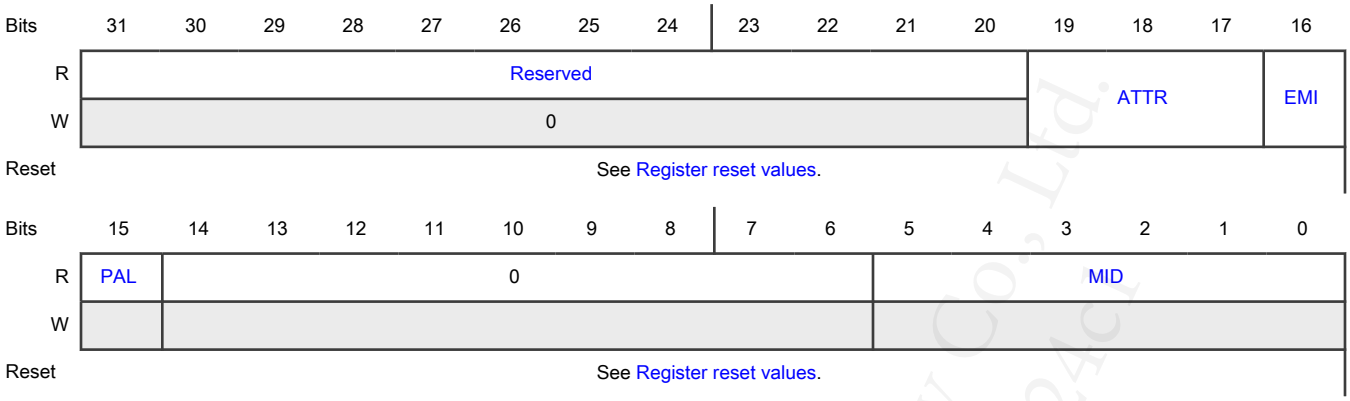
Function

The Channel System Bus register places identification and attribute information on the system bus interface for the eDMA.

The ATTR register outputs the register values onto the system bus interface for further decoding by the security system.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4c21-48a8-b251-390d3ab624c1

Diagram



Register reset values

Register	Reset value
CH0_SBR–CH31_SBR	eDMA_0_TCD: 0000_8006h eDMA_1_TCD: 0000_8007h

Fields

Field	Function
31-20 —	Reserved
19-17 ATTR	Attribute Output DMA's system bus attribute output value.
16 EMI	Enable Master ID Replication The eDMA master ID replication field allows the eDMA to use the same protection level and system bus ID of the master programming the eDMA's TCD. When enabled, the eDMA uses the master ID and protection level stored in the CHn_SBR registers, instead of the eDMA's default values. When a master (for example a core) programs a TCD, its master ID and protection level are captured when the TCD*n*_CSR control attributes are written. A scatter/gather operation does not affect the CHn_SBR registers. You can write the EMI only if CSR[GMRC] = 1, which means Global Master ID Replication Control is enabled; otherwise, the EMI is forced to zero. **NOTE** If master ID replication is disabled, the privileged protection level (Supervisor mode) for DMA transfers is used. 0b - Master ID replication is disabled 1b - Master ID replication is enabled
15	Privileged Access Level

Table continues on the next page...

Table continued from the previous page...

Field	Function
PAL	This field controls DMA's protection level on the system bus when the channel is active. **NOTE** When you enable master ID replication, the value captured in this register is the privilege level of the core or other master writing the channel's transfer control descriptor, which is the lower byte of TCD*n*_CSR. 0b - User protection level for DMA transfers 1b - Privileged protection level for DMA transfers
14-6 —	Reserved
5-0 MID	Master ID This field controls the DMA's master ID on the system bus when the channel is active. **NOTE** The ID captured in this register reflects the master ID of the core or other master writing the channel's control attributes, which are in the lower byte of TCD*n*_CSR.

12.4.2.6 Channel Priority (CH0_PRI - CH31_PRI)

Offset

For n = 0 to 31:

Register	Offset
CHn_PRI	10h + (n × 1000h)

Function

The contents of these registers define unique priorities associated with each channel within the same channel group. Channel grouping is programmed via Channel Arbitration Group (CH0_GRPRI - CH31_GRPRI).

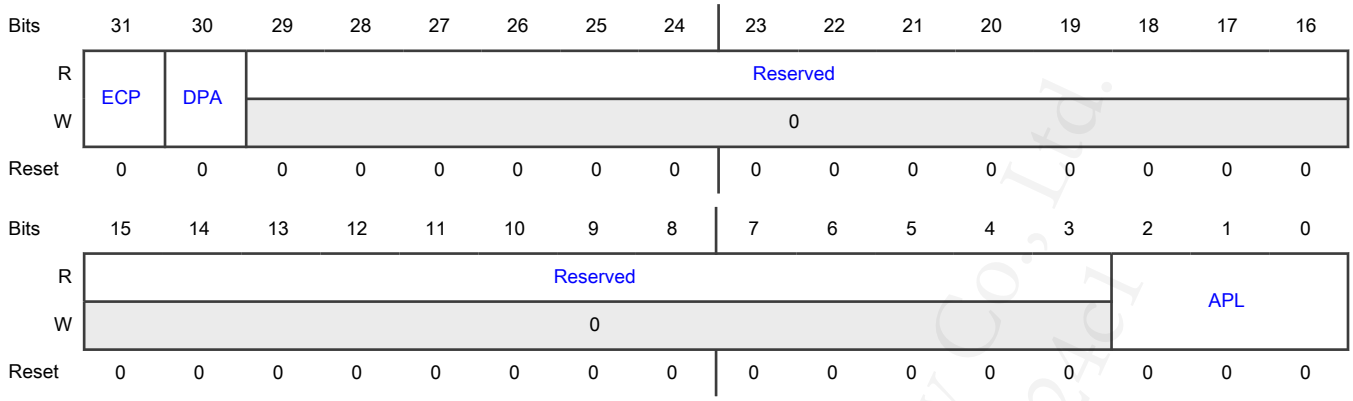
The channel priorities within a group are evaluated by numeric value; for example, 0 is the lowest priority, 1 is the next higher priority, then 2, 3, and so on. Software must program the channel priorities with unique values; otherwise, channel numbers with the same, non-zero value, will be selected based on channel number with the higher channel number having higher priority.

If more than one channel in a group has an arbitration priority level value of zero, then the arbitration mode field CSR[ERCA] is used to determine the arbitration scheme for all channels with APL=0 within a group.

When you enable round-robin channel arbitration (CSR[ERCA] = 1), all channels with APL=0 within a group will use a round-robin arbitration scheme, which rotates among these channels requesting service without regard to priority. Round-robin provides a fairness mechanism within an arbitration group.

When you enable fixed-priority channel arbitration (CSR[ERCA] = 0), eDMA selects channels with APL=0 based on channel number, with the higher channel number having higher priority.

Diagram



Fields

Field	Function
31 ECP	Enable Channel Preemption 0b - Channel cannot be suspended by a higher-priority channel's service request 1b - Channel can be temporarily suspended by a higher-priority channel's service request
30 DPA	Disable Preempt Ability 0b - Channel can suspend a lower-priority channel 1b - Channel cannot suspend any other channel, regardless of channel priority
29-3 —	Reserved
2-0 APL	Arbitration Priority Level Channel priority level for arbitration within the assigned arbitration group.

12.4.2.7 TCD Source Address (TCD0_SADDR - TCD31_SADDR)

Offset

For n = 0 to 31:

Register	Offset
TCDn_SADDR	20h + (n × 1000h)

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4e8-b251-30d3ab62461

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SADDR															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SADDR															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-0 SADDR	Source Address Memory address pointing to the source data.

12.4.2.8 TCD Signed Source Address Offset (TCD0_SOFF - TCD31_SOFF)

Offset

For n = 0 to 31:

Register	Offset
TCDn_SOFF	24h + (n × 1000h)

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SOFF															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
15-0 SOFF	Source Address Signed Offset Sign-extended offset applied to the current source address to form the next-state value as each source read is completed.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c62e5-482-18a8-b251-590d3ab624c1

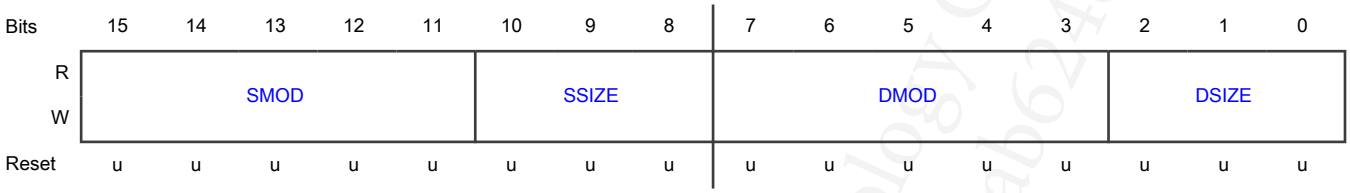
12.4.2.9 TCD Transfer Attributes (TCD0_ATTR - TCD31_ATTR)

Offset

For n = 0 to 31:

Register	Offset
TCDn_ATTR	26h + (n × 1000h)

Diagram



Fields

Field	Function
15-11 SMOD	Source Address Modulo This field defines a specific address range, which is the value after the SADDR + SOFF calculation is performed on the original register value. Setting this field makes it easy to implement a circular data queue. For data queues requiring power-of-2-sized bytes, the queue must start at a 0-modulo-size address and the SMOD field must be set to the appropriate value for the queue, freezing the required number of upper address bits. The value programmed into this field specifies the number of lower address bits that are allowed to change. For a circular queue application, you typically set TCDn_SOFF[SOFF] to the transfer size to implement post-increment addressing, with the SMOD function constraining the addresses to a 0-modulo-size range. 00000b - Source address modulo feature disabled 00001b - Source address modulo feature enabled for any non-zero value [1-31]
10-8 SSIZE	Source Data Transfer Size 000b - 8-bit 001b - 16-bit 010b - 32-bit 011b - 64-bit 100b - 16-byte 101b - 32-byte 110b - 64-byte 111b - Reserved
7-3	Destination Address Modulo

Table continues on the next page...

Table continued from the previous page...

Field	Function
DMOD	See the SMOD definition.
2-0 DSIZE	Destination Data Transfer Size See the SSIZE definition.

12.4.2.10 TCD Transfer Size Without Minor Loop Offsets (TCD0_NBYTES_MLOFFNO - TCD31_NBYTES_MLOFFNO)

Offset

For n = 0 to 31:

Register	Offset
TCDn_NBYTES_MLOFFNO	28h + (n × 1000h)

Function

The TCDn_NBYTES field defines the number of bytes to transfer per service request.

Minor loop offsets are address offset values added to the final source address (TCDn_SADDR), or destination address (TCDn_DADDR), upon minor loop completion. Minor loop completion is when the channel has finished the service request and has transferred NBYTES. When minor loop offsets are enabled, the minor loop offset value (TCDn_NBYTES_MLOFFYES[MLOFF]) is added to the final source address (TCDn_SADDR), to the final destination address (TCDn_DADDR), or to both, prior to the addresses being written back to the TCD. If the major loop is complete, the minor loop offset is ignored and the major loop address offsets (TCDn_SLAST_SDA and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

When minor loop mapping is enabled (SMLOE or DMLOE is 1), TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is redefined. A portion of TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is used to specify multiple fields:

- A source enable bit (SMLOE) to specify the minor loop offset must be applied to the source address (TCDn_SADDR) upon minor loop completion
- A destination enable bit (DMLOE) to specify the minor loop offset must be applied to the destination address (TCDn_DADDR) upon minor loop completion
- The sign extended minor loop offset value (MLOFF)

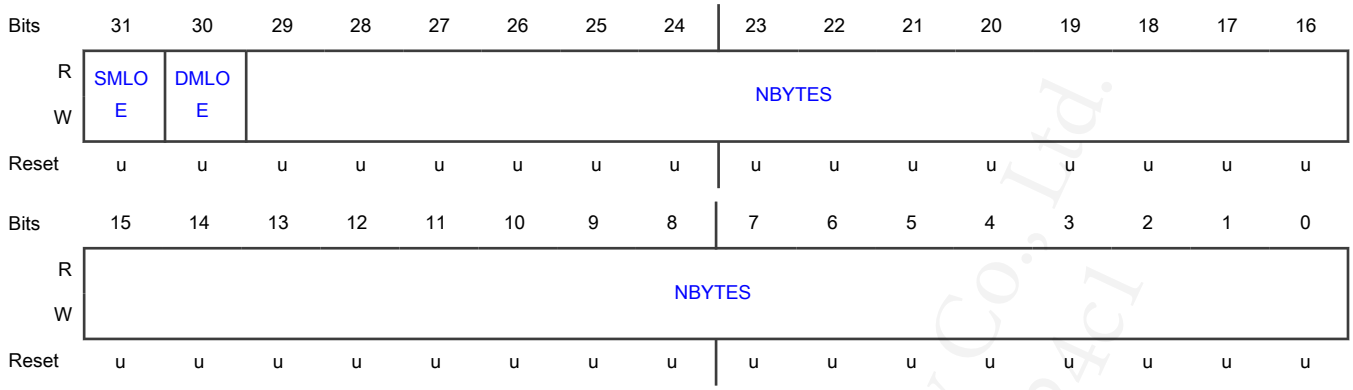
The same offset value (MLOFF) is used for both source and destination minor loop offsets. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. If both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

One of two register profiles (this register or TCDn_NBYTES_MLOFFYES), defines the number of bytes to transfer per request. Which register to use depends on whether source or destination minor loop mapping is enabled.

TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES is defined as follows:

- If SMLOE = 0 and DMLOE = 0, then see the TCDn_NBYTES_MLOFFNO register description.
- If either SMLOE or DMLOE is 1, then see the TCDn_NBYTES_MLOFFYES register description.

Diagram



Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - Minor loop offset not applied to SADDR 1b - Minor loop offset applied to SADDR
30 DMLOE	Destination Minor Loop Offset Enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - Minor loop offset not applied to DADDR 1b - Minor loop offset applied to DADDR
29-0 NBYTES	Number of Bytes To Transfer Per Service Request Number of bytes to be transferred for each service request of the channel. When a channel activates, the module loads the appropriate TCD contents into the eDMA engine and performs the appropriate reads and writes until the byte transfer count has been reached. This process is normally an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the byte count is exhausted, the SADDR and DADDR values are written back into the TCD memory, and the major loop iteration count (CITER) is decremented by one and written back to the TCD memory. If the major iteration count is complete, additional processing is performed.

12.4.2.11 TCD Transfer Size with Minor Loop Offsets (TCD0_NBYTES_MLOFFYES - TCD31_NBYTES_MLOFFYES)

Offset

For n = 0 to 31:

Register	Offset
TCDn_NBYTES_MLOFFYES	28h + (n × 1000h)

Function

The TCDn_NBYTES field defines the number of bytes to transfer per service request.

Minor loop offset is an address offset value added to the final source address (TCDn_SADDR) or destination address (TCDn_DADDR) upon minor loop completion. Minor loop completion occurs when the channel has finished the service request and has transferred NBYTES. Minor loop offsets are enabled by setting either the source enable bit (SMLOE) or the destination enable bit (DMLOE).

The source enable bit (SMLOE) specifies the minor loop offset value (MLOFF) that is to be applied to the source address (TCDn_SADDR) upon minor loop completion. The destination enable bit (DMLOE) specifies the minor loop offset (MLOFF) that is to be applied to the destination address (TCDn_DADDR) upon minor loop completion.

If the major loop is complete, the minor loop offsets are ignored and the major loop address offsets (TCDn_SLAST_SDA and TCDn_DLAST_SGA) are used to compute the next TCDn_SADDR and TCDn_DADDR values.

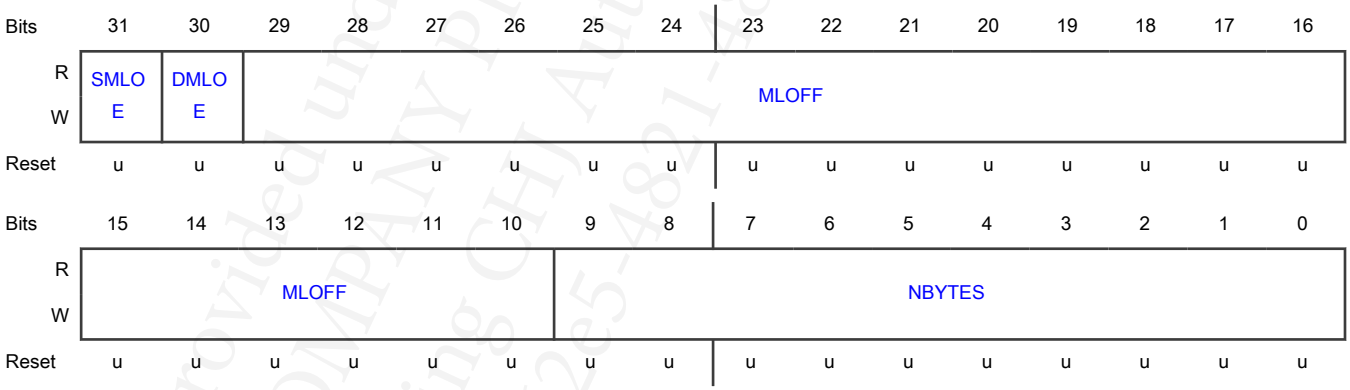
When you enable the minor loop offset overlay (either SMLOE or DMLOE is 1), eDMA redefines TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES. A portion of TCDn_NBYTES_MLOFFNO/TCDn_NBYTES_MLOFFYES specifies the sign-extended minor loop offset value (MLOFF). The same offset value (MLOFF) applies to both source and destination minor loop offsets. When the minor loop offset is enabled, you must align it to the transfer size of the source or destination it is associated with. When either minor loop offset is enabled (SMLOE set or DMLOE set), the NBYTES field is reduced to 10 bits. If both minor loop offsets are disabled (SMLOE cleared and DMLOE cleared), the NBYTES field is a 30-bit vector.

One of two register profiles (this register or TCDn_NBYTES_MLOFFNO) defines the number of bytes to transfer per request. Which register to use depends on whether source or destination minor loop mapping is enabled.

TCDn_NBYTES_MLOFFYES is defined as follows:

- If either minor loop offset is enabled (SMLOE or DMLOE = 1), then see the TCDn_NBYTES_MLOFFYES register description.
- If SMLOE and DMLOE are both 0, then see the TCDn_NBYTES_MLOFFNO register description.

Diagram



Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-b23f-01b3b2f4

Fields

Field	Function
31 SMLOE	Source Minor Loop Offset Enable Selects whether the minor loop offset is applied to the source address upon minor loop completion. 0b - Minor loop offset not applied to SADDR 1b - Minor loop offset applied to SADDR
30 DMLOE	Destination Minor Loop Offset Enable Selects whether the minor loop offset is applied to the destination address upon minor loop completion. 0b - Minor loop offset not applied to DADDR 1b - Minor loop offset applied to DADDR
29-10 MLOFF	Minor Loop Offset If SMLOE or DMLOE is 1, this field represents a sign-extended offset applied to the source or destination address to form the next-state value after the minor loop completes.
9-0 NBYTES	Number of Bytes To Transfer Per Service Request The number of bytes to be transferred in each service request of the channel. As a channel activates, the module loads the appropriate TCD contents into the eDMA engine and performs the appropriate reads and writes until the minor byte transfer count has been reached. This is an indivisible operation and cannot be halted. It can, however, be stalled by using the bandwidth control field, or via preemption. After the minor count is exhausted, the SADDR and DADDR values are written back into the TCD memory, and the major iteration count is decremented and restored to the TCD memory. If the major iteration count is complete, additional processing is performed.

12.4.2.12 TCD Last Source Address Adjustment / Store DADDR Address (TCD0_SLAST_SDA - TCD31_SLAST_SDA)

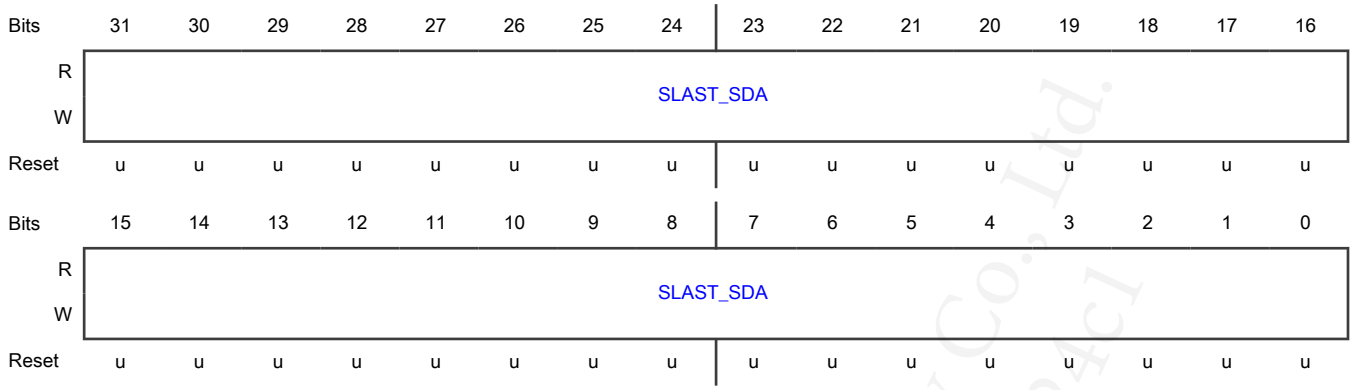
Offset

For n = 0 to 31:

Register	Offset
TCDn_SLAST_SDA	2Ch + (n × 1000h)

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4888-a25f-390d3ab624c1

Diagram



Fields

Field	Function
31-0 SLAST_SDA	Last Source Address Adjustment / Store DADDR Address Source last address adjustment or the system memory address for destination address (DADDR) storage. If (TCDn_CSR[ESDA] = 0), then: • Adjustment value is added to the source address at the completion of the major iteration count. This value can be used to restore the source address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final source address adjustment. Otherwise: • This address points to the 32-bit-aligned memory location where the destination address (DADDR) is to be stored in system memory. By saving the final destination address in system memory via the ESDA feature, you are able to compute the size of a variable destination data buffer by simply subtracting the beginning DADDR from the final, saved DADDR. This feature is used together with the scatter/gather operation to prevent the loss of the final DADDR, which is overwritten during the scatter/gather operation. The "Store Destination Address" (SDA) value must be a 32-bit-aligned location because the eDMA forces the lower two address bits of the SLAST_SDA field to zero when ESDA is enabled. The module performs this write operation when the major loop is done; that is, when the major iteration count (CITER) decrements to zero.

12.4.2.13 TCD Destination Address (TCD0_DADDR - TCD31_DADDR)

Offset

For n = 0 to 31:

Register	Offset
TCDn_DADDR	30h + (n × 1000h)

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DADDR															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-0 DADDR	Destination Address Memory address pointing to the destination data.

12.4.2.14 TCD Signed Destination Address Offset (TCD0_DOFF - TCD31_DOFF)

Offset

For n = 0 to 31:

Register	Offset
TCDn_DOFF	34h + (n × 1000h)

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DOFF															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
15-0 DOFF	Destination Address Signed Offset Sign-extended offset that is applied to the current destination address to form the next-state value as each destination write is completed.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c62e5-482-48a8-b251-90d3ab624c1

12.4.2.15 TCD Current Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_CITER_ELINKNO - TCD31_CITER_ELINKNO)

Offset

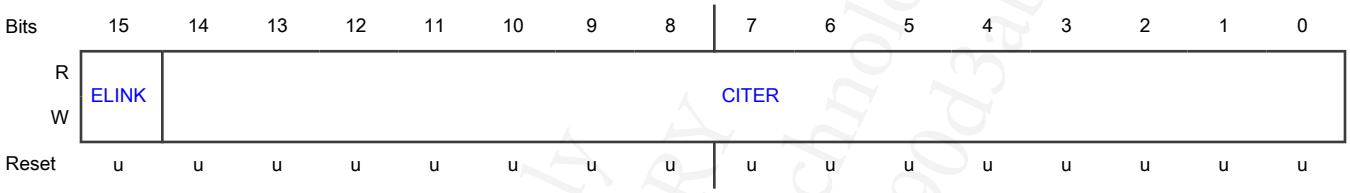
For n = 0 to 31:

Register	Offset
TCDn_CITER_ELINKNO	36h + (n × 1000h)

Function

If TCDn_CITER[ELINK] is 0, the TCDn_CITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by the relevant LINKCH field. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] bit of the specified channel to 1. If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of MAJORELINK channel linking. **NOTE** This field must be equal to the BITER[ELINK] field; otherwise, a configuration error is reported. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14-0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the channel finishes a service request and is written back to TCD memory. After the major iteration count is exhausted, the channel performs a number of operations — for example, final source and destination address calculations — and optionally generates an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

12.4.2.16 TCD Current Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_CITER_ELINKYES - TCD31_CITER_ELINKYES)

Offset

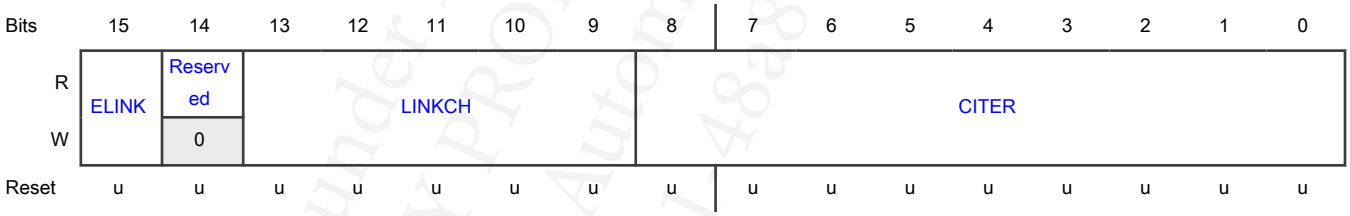
For n = 0 to 31:

Register	Offset
TCDn_CITER_ELINKYES	36h + (n × 1000h)

Function

If TCDn_CITER[ELINK] is 1, the TCDn_CITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by the relevant LINKCH field. When enabled, an internal mechanism sets the TCDn_CSR[START] field of the specified channel (LINKCH) upon minor loop completion.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If channel linking is disabled, the CITER value is extended to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of MAJORELINK channel linking. **NOTE** This field must be equal to the BITER[ELINK] field; otherwise, a configuration error is reported. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14 —	Reserved
13-9 LINKCH	Minor Loop Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted the eDMA engine initiates a channel service request to the channel defined by this field by writing that channel's TCDn_CSR[START] field to 1.
8-0 CITER	Current Major Iteration Count This 9-bit (ELINK = 1) or 15-bit (ELINK = 0) count represents the current major loop count for the channel. It is decremented each time the channel finishes a service request and is written back to the TCD memory. After the major iteration count is exhausted, the channel performs a number of operations — for example, final source and destination address calculations — and optionally generates an interrupt to signal channel completion before reloading the CITER field from the Beginning Iteration Count (BITER) field. **NOTE** When the CITER field is initially loaded by software, it must be set to the same value as that contained in the BITER field. **NOTE** If the channel is configured to execute a single service request, the initial values of BITER and CITER should be 0x0001.

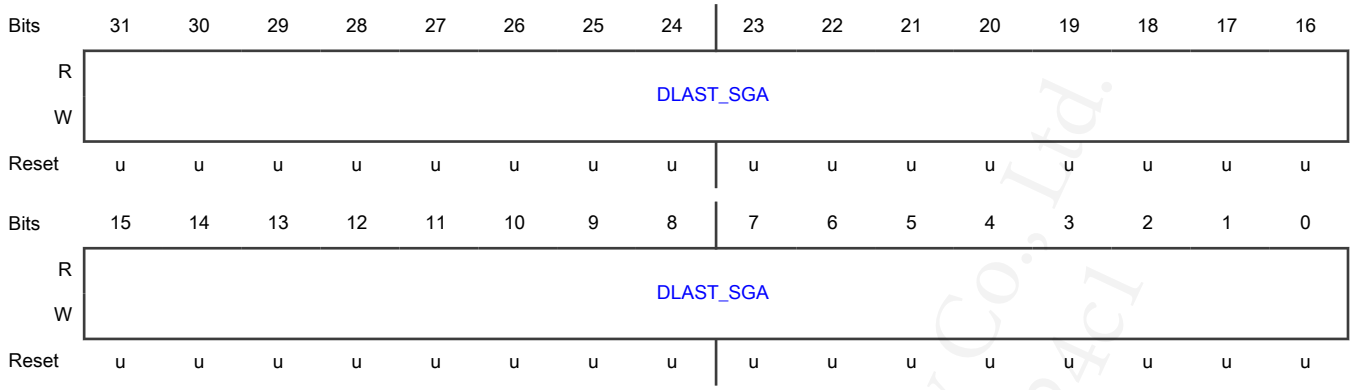
12.4.2.17 TCD Last Destination Address Adjustment / Scatter Gather Address (TCD0_DLAST_SGA - TCD31_DLAST_SGA)

Offset

For n = 0 to 31:

Register	Offset
TCDn_DLAST_SGA	38h + (n × 1000h)

Diagram



Fields

Field	Function
31-0 DLAST_SGA	Last Destination Address Adjustment / Scatter Gather Address Adjustment of the last destination address or the memory address for the next transfer control descriptor to be loaded into this channel (scatter/gather). If (TCDn_CSR[ESG] = 0) then: • Adjustment value is added to the destination address at the completion of the major iteration count. This value can apply to restore the destination address to the initial value or adjust the address to reference the next data structure. • This field uses two's complement notation for the final destination address adjustment. Otherwise: • This address points to the beginning of a 0-modulo 32-byte region containing the next transfer control descriptor to be loaded into this channel. This channel reload is performed as the major iteration count completes. The scatter/gather address must be 0-modulo 32-byte, or else a configuration error is reported.

12.4.2.18 TCD Control and Status (TCD0_CSR - TCD31_CSR)

Offset

For n = 0 to 31:

Register	Offset
TCDn_CSR	3Ch + (n × 1000h)

Provided under NDA
COMPANY PRIVATE
Beijing CHJ Automotive Technology Co.,Ltd.
c8c4625-4821-48a-b25-1390d3ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BWC			MAJORLINKCH					ESDA	EEOP	MAJOREL...	ESG	DREQ	INTHALF	INTMAJOR	START
W			0													
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	0

Fields

Field	Function
15-14 BWC	Bandwidth Control Throttles the amount of bus bandwidth consumed by the eDMA. Generally, as the eDMA processes the minor loop, it continuously generates read/write sequences until the minor count is exhausted. This field forces eDMA to stall after the completion of each read/write access, to control the bus request bandwidth seen by the system bus interconnect. **NOTE** If the source and destination sizes are equal, this field is ignored between the first and second transfers and after the last write of each minor loop. This behavior is a side effect of reducing start-up latency. 00b - No eDMA engine stalls 01b - Enable eDMA master high-priority elevation (HPE) mode. No eDMA engine stalls. 10b - eDMA engine stalls for 4 cycles after each R/W 11b - eDMA engine stalls for 8 cycles after each R/W
13 —	Reserved
12-8 MAJORLINKCH	Major Loop Link Channel Number If (MAJORELINK = 0) then: • No channel-to-channel linking, or chaining, is performed after the major loop counter is exhausted. Otherwise: • After the major loop counter is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] field to 1.
7 ESDA	Enable Store Destination Address As the channel completes the major loop by either the current iteration counter (CITER) decrementing to 0, or by receiving an enabled end-of-packet signal, this field enables writing the destination address (DADDR) to the address stored in the SLAST_SDA field. The value written to system memory is the last DADDR value prior to the DLAST_SGA offset being applied, or overwritten by an enabled scatter/gather operation. When the SDA bit is 1, SLAST_SDA contains the write pointer instead of the final source address offset. Because this is a pointer and not a final offset, a last source address offset of zero is applied to SADDR instead of the SLAST_SGA value. 0b - Ability to store destination address to system memory disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Ability to store destination address to system memory enabled
6 EEOP	Enable End-Of-Packet Processing When enabled by the EEOP field, an end-of-packet hardware input signal directs eDMA to discontinue executing the active channel, and to treat the shutdown as the major-loop-completed event. If the EEOP field is 1, the end-of-packet signal from supported peripherals is accepted. If the EEOP field is 0, the end-of-packet input is ignored. With an end-of-packet retirement, the current TCD destination address (or ESDA-saved destination address), minus the software-saved initial address (DADDR), reflects the total amount of data transferred. 0b - End-of-packet operation disabled 1b - End-of-packet hardware input signal enabled
5 MAJORELINK	Enable Link When Major Loop Complete As the channel completes the major loop, this flag enables linking to another channel defined by MAJORLINKCH. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. **NOTE** To support the dynamic linking coherency model, this field is forced to 0 if written when TCDn_CSR[DONE] is 1. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
4 ESG	Enable Scatter/Gather Processing As the channel completes the major loop, this flag enables scatter/gather processing in the current channel. If enabled, the eDMA engine uses TCDn_DLAST_SGA as a memory pointer to a 0-modulo 32-bit address containing a 32-byte data structure, which is loaded as the transfer control descriptor into local memory. **NOTE** To support the dynamic scatter/gather coherency model, this field is forced to 0 if written when TCDn_CSR[DONE] is 1. 0b - Current channel's TCD is normal format 1b - Current channel's TCD specifies scatter/gather format.
3 DREQ	Disable Request If this flag is 1, the eDMA hardware automatically clears the corresponding ERQ bit when the current major iteration count reaches 0. 0b - Channel's ERQ field not affected 1b - Channel's ERQ field cleared to 0 when major loop complete
2 INTHALF	Enable Interrupt If Major Counter Half-complete

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this flag is 1, the channel generates an interrupt request by setting the appropriate field in the INT register to 1 when the current major iteration count reaches the halfway point. Specifically, the comparison performed by the eDMA engine is (CITER = (BITER/2)). This halfway point interrupt request is provided to support double-buffered, also known as ping-pong, schemes, or other types of data movement where the processor needs an early indication of the transfer's progress. **NOTE** If BITER = 1, do not use INTHALF; use INTMAJOR instead. 0b - Halfway point interrupt disabled 1b - Halfway point interrupt enabled
1 INTMAJOR	Enable Interrupt If Major count complete If this flag is 1, the channel generates an interrupt request by setting the appropriate field in the INT register to 1 when the current major iteration count (CITER) reaches 0. 0b - End-of-major loop interrupt disabled 1b - End-of-major loop interrupt enabled
0 START	Channel Start If this flag is 1, the channel is requesting service. The eDMA hardware automatically clears this flag to 0 after the channel begins execution. 0b - Channel not explicitly started 1b - Channel explicitly started via a software-initiated service request

12.4.2.19 TCD Beginning Major Loop Count (Minor Loop Channel Linking Disabled) (TCD0_BITER_ELINKNO - TCD31_BITER_ELINKNO)

Offset

For n = 0 to 31:

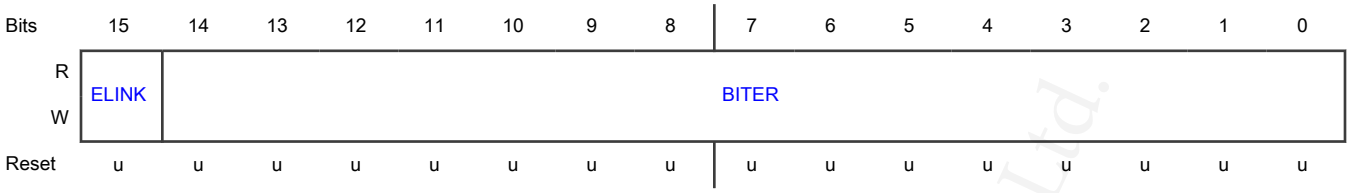
Register	Offset
TCDn_BITER_ELINKNO	3Eh + (n × 1000h)

Function

If the TCDn_BITER[ELINK] field is 0, the TCDn_BITER register is defined as follows.

Provided under NDA to Sony
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-4821-48a8-9251-390d3ab624c1

Diagram



Fields

Field	Function
15 ELINK	Enables Link As the channel completes the minor loop, this flag enables linking to another channel as defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. If channel linking is disabled, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14-0 BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be set equal to the value in the CITER field. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER must be 0x0001.

12.4.2.20 TCD Beginning Major Loop Count (Minor Loop Channel Linking Enabled) (TCD0_BITER_ELINKYES - TCD31_BITER_ELINKYES)

Offset

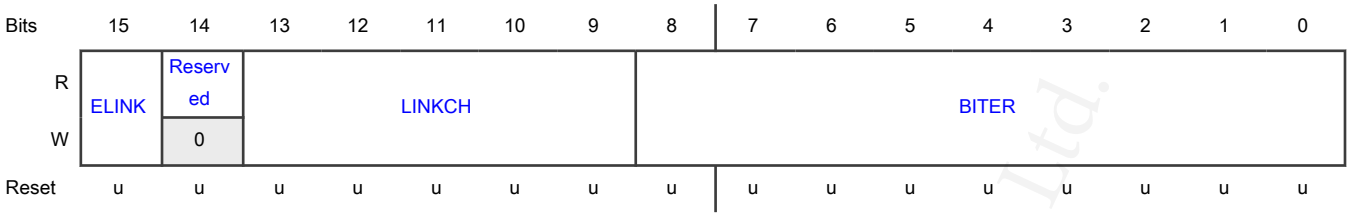
For n = 0 to 31:

Register	Offset
TCDn_BITER_ELINKYES	3Eh + (n × 1000h)

Function

If the TCDn_BITER[ELINK] field is set, the TCDn_BITER register is defined as follows.

Diagram



Fields

Field	Function
15 ELINK	Enable Link As the channel completes the minor loop, this flag enables linking to another channel as defined by BITER[LINKCH]. The link target channel initiates a channel service request via an internal mechanism that sets the TCDn_CSR[START] field of the specified channel. If channel linking disables, the BITER value extends to 15 bits in place of a link channel number. If the major loop is exhausted, this link mechanism is suppressed in favor of the MAJORELINK channel linking. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. 0b - Channel-to-channel linking disabled 1b - Channel-to-channel linking enabled
14 —	Reserved
13-9 LINKCH	Link Channel Number If channel-to-channel linking is enabled (ELINK = 1), then after the minor loop is exhausted, the eDMA engine initiates a channel service request at the channel defined by this field by setting that channel's TCDn_CSR[START] field. **NOTE** When the software loads the TCD, this field must be set equal to the corresponding CITER field; otherwise, a configuration error is reported. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field.
8-0 BITER	Starting Major Iteration Count As the transfer control descriptor is first loaded by software, this 9-bit (ELINK = 1) or 15-bit (ELINK = 0) field must be set equal to the value in the CITER field. As the major iteration count is exhausted, eDMA reloads the contents of this field into the CITER field. If the channel is configured to execute a single service request, the initial values of BITER and CITER must be 0x0001.

12.5 Functional description

The operation of eDMA is described in the following subsections.

12.5.1 eDMA basic data flow

The basic flow of a data transfer can be partitioned into three segments.

As shown in the following diagram, the first segment involves the channel activation:

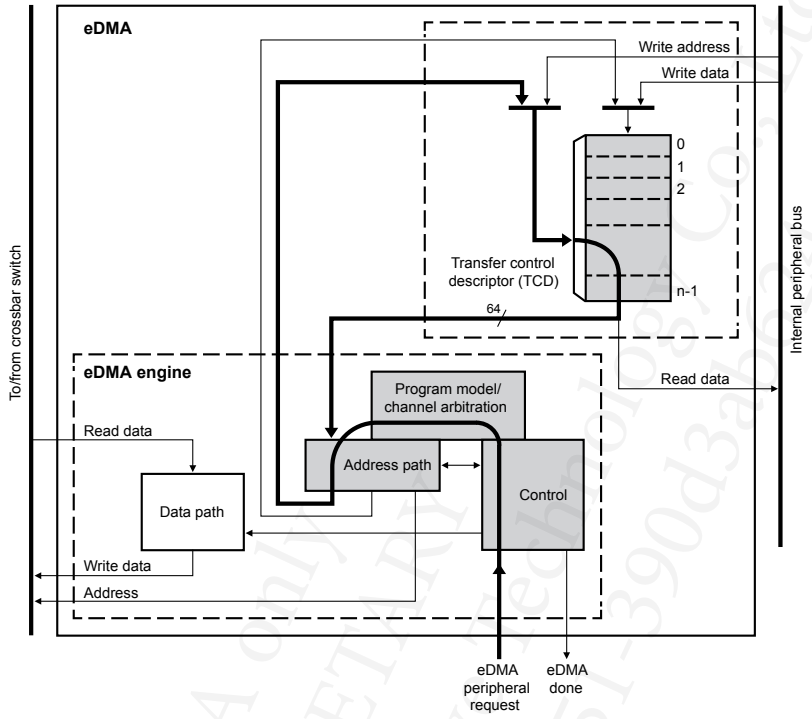


Figure 23. eDMA operation, part 1

This example uses the assertion of the eDMA peripheral request signal to request service for channel *n*. Channel activation via software and the TCD*n*_CSR[START] field follows the same basic flow as peripheral requests. The eDMA request input signal is registered internally and then routed through the eDMA engine: first through the control module, then into the program model and channel arbitration.

In the next cycle, the channel arbitration begins using fixed-priority plus the optional round-robin algorithm. After arbitration is complete, the activated channel number is sent through the address path and converted into the required address to access the local memory for TCD*n*. Next, the TCD memory is accessed and the required descriptor is read from the local memory and then loaded into the eDMA engine address path's primary or secondary channel execution registers. The TCD memory is 64 bits wide to minimize the time needed to fetch the activated channel descriptor and load it into the address path registers.

The following diagram illustrates the second part of the basic data flow:

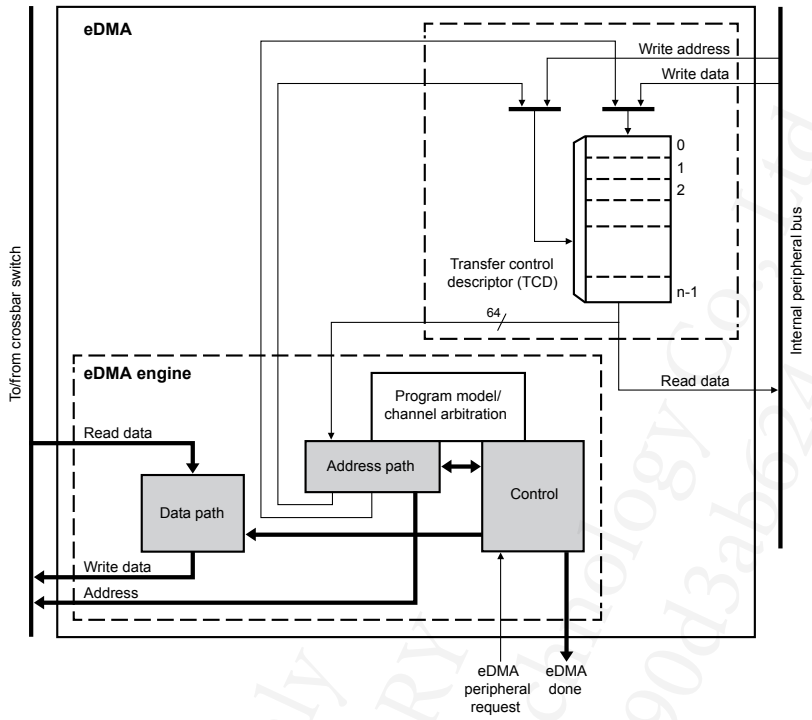


Figure 24. eDMA operation, part 2

The modules associated with the data transfer (address path, data path, and control) go through the required sequence of source reads and destination writes to perform the actual data movement. The source reads are initiated, and the fetched data is temporarily stored in the data path block until it is gated onto the internal bus during the destination write. This source read/destination write processing continues until the byte count, NBYTES, has been transferred.

After NBYTES of data has been moved, the final phase of the basic data flow is performed. In this segment, the address path logic performs the required updates to certain fields in the appropriate TCD (for example, SADDR, DADDR, CITER). If the major iteration count is exhausted, additional operations are performed. These include the final address adjustments and reloading of the BITER field into the CITER field. Assertion of an optional interrupt request also occurs at this time, as does a possible fetch of a new TCD from memory using the scatter/gather address pointer included in the descriptor (if scatter/gather is enabled). The updates to the TCD memory and the assertion of an interrupt request are shown in the following diagram.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4e5b-b7...390d3ab924c1

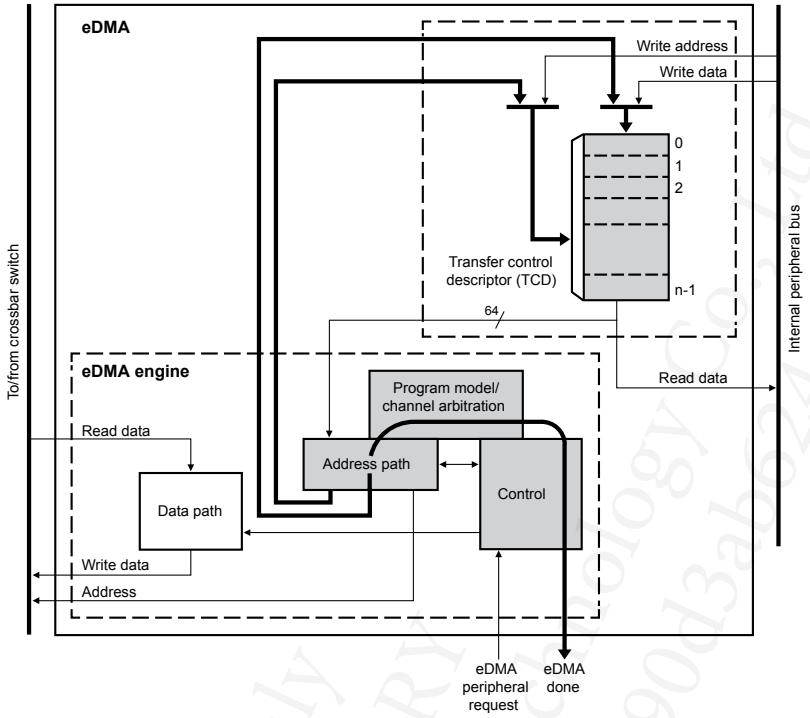


Figure 25. eDMA operation, part 3

12.5.2 Fault reporting and handling

Channel errors are reported in the Error Status register (CHn_CSR and TCDn_CSR) and can be caused by any of the following:

- A configuration error, which is an illegal setting in the transfer control descriptor
- An active channel canceled via a "cancel transfer with error" hardware or software request
- A TCD memory error
- An error termination to a bus master read or write cycle

A configuration error is reported when an inconsistent state is represented by one of these factors:

- Starting source or destination address
- Source or destination offsets
- Minor loop byte count
- Transfer size

Each of these possible causes is detailed below:

- The addresses and offsets must be aligned on zero-modulo-transfer-sized boundaries.
- The minor loop byte count must be a multiple of the source and destination transfer sizes.
- All source reads and destination writes must be configured to the natural boundary of the programmed transfer size.

NOTE

To aid in debugging, set the Halt After Error field in the DMA's Control Status register, CSR[HAE]. Upon any error condition, the DMA is halted after the error is recorded. The DMA remains halted and does not process any channel service requests. After the error is fixed, the DMA may be enabled again by clearing the Halt field, CSR[HALT].

Provided under NDA only
COMPANY PROPRIETARY
Copyright CHJ Automotive Technology Co., Ltd.
e0462a-4821-48a8-b251-390d3ab924c1

- If a scatter/gather operation is enabled upon channel completion, a configuration error is reported if the scatter/gather address (DLAST_SGA) is not aligned on a 32-byte boundary.
- If minor loop channel linking is enabled upon channel completion, a configuration error is reported when the link is attempted if the TCDn_CITER[ELINK] field does not equal the TCDn_BITER[ELINK] field.

If enabled, all configuration error conditions, except the scatter/gather and minor-loop link errors, are reported as the channel activates and asserts an error interrupt request. A scatter/gather configuration error is reported when the scatter/gather operation begins at major loop completion if properly enabled. A minor loop channel link configuration error is reported when the link operation is serviced at minor loop completion.

The occurrence of any error causes the eDMA engine to stop normal processing of the active channel immediately (it goes to its error processing states and the transaction to the system bus still has pipeline effect), and the appropriate channel field in the eDMA error register is set to 1. At the same time, the details of the error condition are loaded into the Error Status register (CHn_CSR and TCDn_CSR). The major loop complete indicators, setting the transfer control descriptor DONE flag, and the possible assertion of an interrupt request are not affected when an error is detected.

After the error status has been updated, the eDMA engine continues operating by servicing the next appropriate channel. A channel that experiences an error condition is not automatically disabled. If a channel is terminated by an error and then issues another service request before the error is fixed, that channel executes and terminates with the same error condition.

The error status fields are read-only. These error indicators are sticky and cannot be cleared. They show the last recorded error until the DMA is reset. The valid field (VLD) is used to determine if a new error condition exists. This field is the logical OR of each channel's error interrupt field (ERR).

After the software has resolved any errors and cleared all of the error interrupt fields, the valid field is cleared to 0 but the cause of the last error is still indicated.

12.5.3 Channel preemption

Channel preemption is enabled on a per-channel basis by setting the CHn_PRI[ECP] field. Channel preemption allows the executing channel’s data transfers to temporarily suspend in favor of starting a higher-priority channel. After the preempting channel has completed all of its minor loop data transfers, the preempted channel is restored and resumes execution.

After the restored channel completes one read/write sequence, it is again eligible for preemption. If any higher priority channel is requesting service, the restored channel is suspended, and the higher-priority channel is serviced. Nested preemption, that is, attempting to preempt a preempting channel, is not supported. After a preempting channel begins execution, it cannot be preempted.

A channel’s ability to preempt another channel can be disabled by setting CHn_PRI[DPA] to 1. When a channel’s preempt ability is disabled, that channel cannot suspend a lower-priority channel’s data transfer, regardless of the lower-priority channel’s ECP setting. This allows for a pool of low-priority, large-data-moving channels to be defined.

You can configure these low-priority channels to not preempt each other, thus preventing a low-priority channel from consuming the preempt slot normally available to a true high-priority channel. When you enable round-robin channel arbitration mode (CSR[ERCA] is set to 1), any channel with a priority level equal to 0 (CHn_PRI[APL] = 0) has preemption disabled and cannot preempt another channel.

12.6 Initialization/application information

The following sections discuss initialization of the eDMA and programming considerations.

12.6.1 eDMA initialization

To initialize the eDMA:

1. Write to the CSR if a configuration other than the default is wanted.
2. Write the channel priority levels to the CHn_PRI registers and group priority levels to the CHn_GRPRI registers if a configuration other than the default is wanted.
3. Enable error interrupts in the CHn_CSR[EEI] registers if they are wanted.

4. Write the 32-byte TCD for each channel that may request service.
5. Enable any hardware service requests via the CHn_CSR[ERQ] registers.
6. Request channel service via either:
   - Software: setting TCDn_CSR[START]
   - Hardware: slave device asserting its eDMA peripheral request signal

After any channel requests service, a channel is selected for execution based on the arbitration and priority levels written into the programmer's model. The eDMA engine reads the entire TCD, including the TCD control and status fields, as shown in Table 42, for the selected channel into its internal address path module.

As the TCD is read, the first transfer is initiated on the internal bus, unless a configuration error is detected. Transfers from the source, defined by TCDn_SADDR, to the destination, defined by TCD_DADDR, continue until the number of bytes specified by TCDn_NBYTES are transferred.

When the transfer is complete, the eDMA engine's local TCDn_SADDR, TCDn_DADDR, and TCDn_CITER are written back to the main TCD memory and any minor loop channel linking is performed, if enabled. If the major loop is exhausted, then eDMA executes further post-processing, such as interrupts, major loop channel linking, and scatter/gather operations, if enabled.

Table 42. TCD control and status (TCDn_CSR) fields

TCDn_CSR field name	Description
START	Control field to start the channel explicitly when using a software-initiated DMA service (automatically cleared by hardware)
EEOP	Control field to enable end-of-packet processing
ESDA	Control field to enable storing of the destination address to system memory after the major loop completes
DREQ	Control field to disable hardware-initiated DMA service requests after major loop completion
BWC	Control field for throttling the bandwidth control of a channel
ESG	Control field to enable the scatter-gather feature
INTHALF	Control field to enable interrupt when major loop is half-complete
INTMAJOR	Control field to enable interrupt when major loop completes

Table 43. Channel control and status (CHn_CSR) fields

CHn_CSR field name	Description
ACTIVE	Status field indicating the channel is currently in execution
DONE	Status field indicating major loop completion (cleared by software when a channel begins execution)
EEI	Control field to enable error interrupts
EARQ	Control field to enable external, asynchronous wakeup event in conjunction with the ERQ field
ERQ	Control field to enable hardware service requests

The following figure shows how each DMA request initiates one minor-loop transfer, or iteration, without CPU intervention. DMA arbitration can occur after each minor loop, and one level of minor loop DMA preemption is allowed. The number of minor loops in a major loop is specified by the beginning iteration count (BITER).

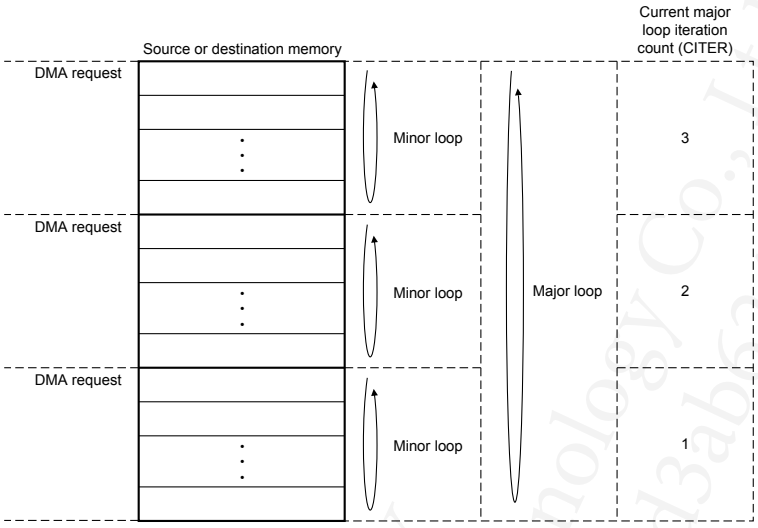


Figure 26. Example of multiple loop iterations

The following figure lists the memory array terms and how the TCD settings are related.

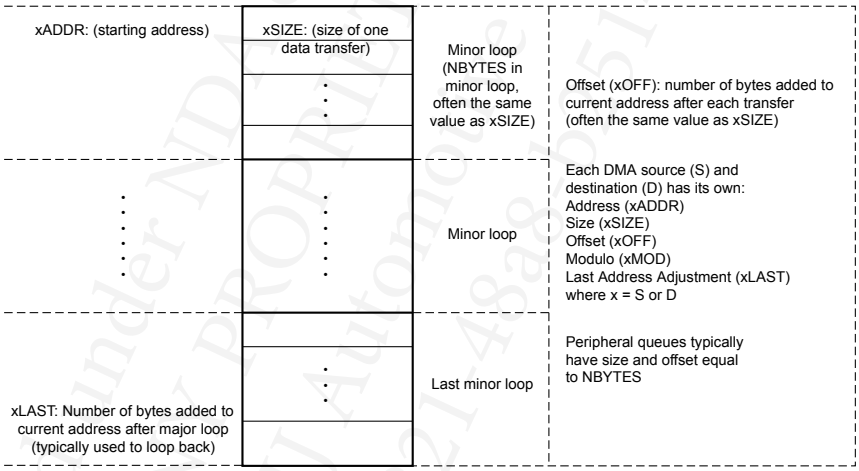


Figure 27. Memory array terms

12.6.2 eDMA arbitration

The eDMA uses a layered arbitration scheme composed of multiple priority levels. The eDMA uses a fixed-priority arbitration scheme with optional round-robin arbitration under specific conditions. The priorities are evaluated in the following order:

Table 44. eDMA arbitration priorities

Priority	Scheme	Description
1 (Highest)	Arbitration group priority	Each channel is assigned an arbitration group via the CHn_GRPRI registers.

Table continues on the next page...

Table 44. eDMA arbitration priorities (continued)

Priority	Scheme	Description
		Priority is given to the highest value (31 being the highest possible value) down to the lowest value (zero, the default).
2	Channel priority	Each channel is assigned a channel priority level via the CHn_PRI registers. The channel priority is a relative priority level within an arbitration group. Priority is given to the highest value (seven being the highest possible value) down to the lowest value (zero, the default). Channel priorities within each arbitration group need not be unique. If multiple channels have the same channel priority level, the channel number will be used to determine priority as defined in row three.
3	Channel number	When two or more channels have the same arbitration group priority and channel priority, the channel number (CHn_NUM) is used to determine the highest priority. Priority is giver to the highest channel number. Lowest priority is channel 0. The channel numbers are static and cannot be changed in the programmer's model.
4 (Lowest)	Round-robin	When round-robin is enabled, any channel configured for round-robin operation has lowest priority within an arbitration group. Round-robin is enabled by setting the CSR[ERCA] field to 1. After being enabled, channels with a channel priority of zero (CHn_PRI=0) will use round-robin arbitration. Round-robin arbitration will rotate the channel selection among the channels requesting service with CHn_PRI=0 within the arbitration group. Any non-zero channel within the arbitration group will continue to use fixed-priority arbitration, and if requesting service will be selected over any round-robin channels.

For fixed arbitration, the overall priority can be considered a number composed of three concatenated priority levels: CHn_GRPRI:CHn_PRI:CH_NUM. The largest number has the highest priority and the lowest number has the lowest priority.

For round-robin arbitration, the priority number is CHn_GRPRI:0:X. The module rotates through the CHn_PRI=0 channels requesting service without regard to priority among these channels. Any channel within the arbitration group for which CHn_PRI is greater than 0 will be serviced before the round-robin channels.

Provided under NDA only
COMPANY PROPRIETARY
Continental CHJ Automotive Technology (Beijing)
ca240e5-4821-48a8-b251-

12.6.3 Programming errors

The eDMA performs various tests on the transfer control descriptor to verify consistency in the descriptor data.

The channel number causing the error is recorded in the Error Status register (CHn_CSR and TCDn_CSR). If the error source is not removed before the next activation of the problematic channel, the error is detected and recorded again. Setting the halt after error field, CSR[HAE], will halt the DMA and prevent reoccurrence of the error.

12.6.4 Arbitration mode considerations

This section discusses arbitration considerations for eDMA.

12.6.4.1 Fixed group arbitration, fixed channel arbitration

In this mode, eDMA selects for execution the channel service request from the highest-priority channel in the highest-priority group. If eDMA is programmed so that the channels within a high-priority group have a high number of requests or large data transfers, that group may consume all the bandwidth of the eDMA controller. That is, no lower-priority groups are serviced if there is always at least one DMA request pending on a channel in the highest-priority group when the controller arbitrates the next DMA request. The advantage of this scenario is that latency can be small for channels that need to be serviced quickly.

12.6.4.2 Fixed group arbitration, round-robin channel arbitration

The highest-priority group with a request is serviced. Lower-priority groups are serviced if no pending requests exist in the higher-priority groups.

Within each group, channels are serviced starting with the highest non-zero channel priority. For all channels with a channel priority programmed to 0, selection begins with the highest channel number requesting service and then rotates through to the lowest channel number requesting service. The round-robin channel arbitration can provide a fairness mechanism to lower-priority channels.

This scenario could cause the same bandwidth consumption problem as indicated in Fixed group arbitration, fixed channel arbitration, but all the channels in the highest-priority group will be serviced. Service latency is short on the highest-priority group, but could potentially be very much longer as the group priority decreases.

12.6.5 Performing DMA transfers

This section presents examples on how to perform DMA transfers with the eDMA.

12.6.5.1 Single request

To perform a simple transfer of n bytes of data with one activation, set the major loop to one (TCD*n*_CITER = TCD*n*_BITER = 1). The data transfer begins after the channel service request is acknowledged and the channel is selected to execute. After the transfer is complete, the CH*n*_CSR[DONE] field is set to 1 and an interrupt is generated if properly enabled.

For example, the following TCD entry is configured to transfer 16 bytes of data. The eDMA is programmed for one iteration of the major loop transferring 16 bytes per iteration. The source memory has a byte-wide memory port located at 0x1000. The destination memory has a 32-bit port located at 0x2000. The address offsets are programmed in increments to match the transfer size: one byte for the source, and four bytes for the destination. The final source and destination addresses are adjusted to return to their beginning values.

```
TCDn_CITER = TCDn_BITER = 1
TCDn_NBYTES = 16
TCDn_SADDR = 0x1000
TCDn_SOFF = 1
TCDn_ATTR[SSIZE] = 0
TCDn_SLAST = -16
TCDn_DADDR = 0x2000
TCDn_DOFF = 4
TCDn_ATTR[DSIZE] = 2
```

```
TCDn_DLAST_SGA= -16
TCDn_CSR[INTMAJ] = 1
TCDn_CSR[START] = 1 (should be written last after all other fields have been initialized)
All other TCDn fields = 0
```

This generates the following event sequence:

1. User write to the TCD*n*_CSR[START] field requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes:
   - CH*n*_CSR[DONE] = 0
   - TCD*n*_CSR[START] = 0
   - CH*n*_CSR[ACTIVE] = 1
4. eDMA engine reads: channel TCD data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:
   a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
   b. Write 32 bits to location 0x2000 → first iteration of the minor loop.
   c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
   d. Write 32 bits to location 0x2004 → second iteration of the minor loop.
   e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
   f. Write 32 bits to location 0x2008 → third iteration of the minor loop.
   g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
   h. Write 32 bits to location 0x200C → last iteration of the minor loop → major loop complete.
6. The eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 1 (TCD*n*_BITER).
7. The eDMA engine writes: CH*n*_CSR[ACTIVE] = 0, CH*n*_CSR[DONE] = 1, CH*n*_INT[INT] = 1.
8. The channel retires and the eDMA goes idle or services the next channel.

12.6.5.2 Multiple requests

The following example transfers 32 bytes via two hardware requests, but is otherwise the same as the previous example. The only fields that change are the major loop iteration count and the final address offsets. The eDMA is programmed for two iterations of the major loop, transferring 16 bytes per iteration. After the channel's hardware requests are enabled via the CHn_CSR[ERQ] register field, the slave device initiates channel service requests.

```
TCDn_CITER = TCDn_BITER = 2
TCDn_SLAST = -32
TCDn_DLAST_SGA = -32
```

This would generate the following sequence of events:

1. First hardware (eDMA peripheral) requests channel service.
2. The channel is selected by arbitration for servicing.
3. eDMA engine writes: CH*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, CH*n*_CSR[ACTIVE] = 1.
4. eDMA engine reads: channel TCD*n* data from local memory to internal register file.
5. The source-to-destination transfers are executed as follows:

a. Read byte from location 0x1000, read byte from location 0x1001, read byte from 0x1002, read byte from 0x1003.
b. Write 32 bits to location 0x2000 → first iteration of the minor loop.
c. Read byte from location 0x1004, read byte from location 0x1005, read byte from 0x1006, read byte from 0x1007.
d. Write 32 bits to location 0x2004 → second iteration of the minor loop.
e. Read byte from location 0x1008, read byte from location 0x1009, read byte from 0x100A, read byte from 0x100B.
f. Write 32 bits to location 0x2008 → third iteration of the minor loop.
g. Read byte from location 0x100C, read byte from location 0x100D, read byte from 0x100E, read byte from 0x100F.
h. Write 32 bits to location 0x200C → last iteration of the minor loop.

6. eDMA engine writes: TCD*n*_SADDR = 0x1010, TCD*n*_DADDR = 0x2010, TCD*n*_CITER = 1.
7. eDMA engine writes: CH*n*_CSR[ACTIVE] = 0.
8. The channel retires, which concludes one iteration of the major loop. The eDMA goes idle or services the next channel.
9. Second hardware (eDMA peripheral) requests channel service.
10. The channel is selected by arbitration for servicing.
11. eDMA engine writes: CH*n*_CSR[DONE] = 0, TCD*n*_CSR[START] = 0, CH*n*_CSR[ACTIVE] = 1.
12. eDMA engine reads: Channel TCD data from local memory to internal register file.
13. The source-to-destination transfers are executed as follows:
    a. Read byte from location 0x1010, read byte from location 0x1011, read byte from 0x1012, read byte from 0x1013.
    b. Write 32 bits to location 0x2010 → first iteration of the minor loop.
    c. Read byte from location 0x1014, read byte from location 0x1015, read byte from 0x1016, read byte from 0x1017.
    d. Write 32 bits to location 0x2014 → second iteration of the minor loop.
    e. Read byte from location 0x1018, read byte from location 0x1019, read byte from 0x101A, read byte from 0x101B.
    f. Write 32 bits to location 0x2018 → third iteration of the minor loop.
    g. Read byte from location 0x101C, read byte from location 0x101D, read byte from 0x101E, read byte from 0x101F.
    h. Write 32 bits to location 0x201C → last iteration of the minor loop → major loop complete.
14. eDMA engine writes: TCD*n*_SADDR = 0x1000, TCD*n*_DADDR = 0x2000, TCD*n*_CITER = 2 (TCD*n*_BITER).
15. eDMA engine writes: CH*n*_CSR[ACTIVE] = 0, CH*n*_CSR[DONE] = 1, CH*n*_INT[INT] = 1.
16. The channel retires, which concludes with the major loop complete. The eDMA goes idle or services the next channel.

12.6.5.3 Using the modulo feature

The modulo feature of the eDMA allows implementation of a circular data queue in which the size of the queue is a power of 2. xMOD is a 5-bit field for the source and destination in the TCD, and it specifies which lower address bits increment from their original value after the address+offset calculation. All upper address bits remain the same as in the original value. A setting of 0 for this field disables the modulo feature. Modulo addressing applies to cases where the minor loop offset is enabled; that is, the upper address bits remain the same after the minor loop offset is added to the source or destination address.

The following table shows how the transfer addresses are specified based on the setting of the MOD field. Here a circular buffer is created where the address wraps to the original value but the 28 upper address bits (0x1234567*x*) retain their original value. In this example, the source address is set to 0x12345670, the offset is set to four bytes, and the MOD field is set to four, which allows for a 2^4 byte (16 byte) queue size.

Table 45. Modulo example

Transfer number	Address
1	0x12345670
2	0x12345674
3	0x12345678
4	0x1234567C
5	0x12345670
6	0x12345674

12.6.6 Monitoring transfer descriptor status

This section discusses how to monitor eDMA status.

12.6.6.1 Testing for minor loop completion

There are two methods to test for minor loop completion when using software-initiated service requests.

1. The first method is to read the TCDn_CITER field and test for a change.
2. The second method, extracted from the sequence shown below, is to test the TCDn_CSR[START] field and the CHn_CSR[ACTIVE] field. The minor-loop-complete condition is indicated by both fields reading 0 after TCDn_CSR[START] is set to 1. Polling the CHn_CSR[ACTIVE] field only may be inconclusive because the active status may be missed if the channel execution is short in duration.

The CHn_CSR and TCDn_CSR status fields execute the following sequence for a software-activated channel:

Stage	TCDn_CSR field	CHn_CSR fields		State
	START	ACTIVE	DONE	
1	1	0	0	Initiate channel service request via software.
2	0	1	0	Channel is executing.
3a	0	0	0	Channel has completed the minor loop and is idle.
3b	0	0	1	Channel has completed the major loop and is idle.

The best method to test for minor-loop completion when using hardware-initiated (that is, peripheral-initiated) service requests is to read the TCDn_CITER field and test for a change. The hardware request and acknowledge handshake signals are not visible in the programmer's model.

The TCD status fields execute the following sequence for a hardware-activated channel:

Beijing CHJ Automotive Technology Co., Ltd. COMPANY PROPRIETARY Provided under NDA only c8c42e-4821-4a8-b25-390d3ab624c1

Stage	TCDn_CSR field	CHn_CSR fields		State
	START	ACTIVE	DONE	
1	0	0	0	Initiate channel service request via hardware (peripheral request asserted).
2	0	1	0	Channel is executing.
3a	0	0	0	Channel has completed the minor loop and is idle.
3b	0	0	1	Channel has completed the major loop and is idle.

For both activation types, the major-loop-complete status is explicitly indicated via the CHn_CSR[DONE] field.

The TCDn_CSR[START] field is cleared to 0 automatically when the channel begins execution, regardless of how the channel activates.

12.6.6.2 Reading the transfer descriptors of active channels

The eDMA reads back the true TCDn_SADDR, TCDn_DADDR, and TCDn_NBYTES values if they are read when a channel executes. The true values of SADDR, DADDR, and NBYTES are the values the eDMA engine currently uses in its internal register file, and not the values in the TCD local memory for that channel. The addresses, SADDR and DADDR, and NBYTES (which decrements to zero as the transfer progresses), can give an indication of the progress of the transfer. All other values are read back from the TCD local memory.

12.6.6.3 Checking channel preemption status

A preemptive situation is one in which a preempt-enabled channel is executing and a higher-priority request becomes active. When round-robin channel arbitration mode is enabled, all channels with their channel priority set to 0 lose their preempt ability. Channel priorities of 0 are treated as equal, that is, they are constantly rotating, when round-robin arbitration mode is enabled.

The CHn_CSR[ACTIVE] field for the preempted channel remains asserted throughout the preemption. The preempted channel is temporarily suspended when the preempting channel executes one major loop iteration. If two CHn_CSR[ACTIVE] fields are set simultaneously in the global TCD map, a higher-priority channel is actively preempting a lower-priority channel.

12.6.7 Channel linking

Channel linking (or chaining) is a mechanism in which one channel sets the TCDn_CSR[START] field of another channel (or itself), thus initiating a service request for that channel. When properly enabled, the eDMA engine automatically performs this operation at the major or minor loop completion.

The minor loop channel linking occurs at the completion of the minor loop (or one iteration of the major loop). The TCDn_CITER[ELINK] field determines whether a minor loop link is requested. When enabled, the channel link is made after each iteration of the major loop except for the last. When the major loop is exhausted, only the major loop channel link fields are used to determine if a channel link should be made. For example, using an initial field setting of:

```
TCDn_CITER[ELINK] = 1
TCDn_CITER[LINKCH] = 0xC
TCDn_CITER[CITER] value = 0x4
TCDn_CSR[MAJORELINK] = 1
TCDn_CSR[MAJORLINKCH] = 0x7
```

executes as:

1. Minor loop done → set TCD12_CSR[START] field

2. Minor loop done → set TCD12_CSR[START] field
3. Minor loop done → set TCD12_CSR[START] field
4. Minor loop done, major loop done→ set TCD7_CSR[START] field

When minor loop linking is enabled (TCDn_CITER[ELINK] = 1), the TCDn_CITER[CITER] field uses a nine-bit vector to form the current iteration count. When minor loop linking is disabled (TCDn_CITER[ELINK] = 0), the TCDn_CITER[CITER] field uses a 15-bit vector to form the current iteration count. The bits associated with the TCDn_CITER[LINKCH] field are concatenated onto the CITER value to increase the range of the CITER.

NOTE

The TCDn_CITER[ELINK] field and the TCDn_BITER[ELINK] field must be equal — if they are not, a configuration error is reported. The CITER and BITER vector widths must be equal to calculate the major loop halfway done interrupt point.

The following table summarizes how a DMA channel can link to another DMA channel, that is, use another channel's TCD, at the end of a loop.

Table 46. Channel linking parameters

Wanted link behavior	TCD control field name	Description
Link at end of minor loop	TCDn_CITER[ELINK]	Enable channel-to-channel linking on minor loop completion (current iteration)
	TCDn_CITER[LINKCH]	Link channel number when linking at end of minor loop (current iteration)
Link at end of major loop	TCDn_CSR[MAJORELINK]	Enable channel-to-channel linking on major loop completion
	TCDn_CSR[MAJORLINKCH]	Link channel number when linking at end of major loop

12.6.8 Dynamic programming

This section provides recommended methods to change the programming model during channel execution.

12.6.8.1 Dynamically changing the channel priority

To change group or channel priority levels:

1. Halt the DMA by writing 1 to the CSR[HALT] field.
2. Change the group or channel priorities as wanted.
3. Enable normal DMA operations by writing 0 to the CSR[HALT] field.

12.6.8.2 Dynamic channel linking

Dynamic channel linking is the process of setting the TCDn_CSR[MAJORELINK] field during channel execution (see the diagram in TCD structure). This field is read from the TCD local memory at the end of channel execution, thus allowing you to enable the feature during channel execution.

Because you are allowed to change the configuration during execution, you need a coherency model. Consider the scenario where you attempt to execute a dynamic channel link by enabling the TCDn_CSR[MAJORELINK] field at the same time the eDMA engine is retiring the channel. TCDn_CSR[MAJORELINK] would be set in the programmer's model, but it would be unclear whether the actual link was made before the channel retired.

We recommend that you use the following coherency model when executing a dynamic channel link request.

1. Write 1 to the TCDn_CSR[MAJORELINK] field.

2. Read back the TCDn_CSR[MAJORELINK] field.
3. Test the TCDn_CSR[MAJORELINK] request status:
   - If TCDn_CSR[MAJORELINK] = 1, the dynamic link attempt was successful.
   - If TCDn_CSR[MAJORELINK] = 0, the attempted dynamic link did not succeed (the channel was already retiring).

For this request, the TCD local memory controller forces the TCDn_CSR[MAJORELINK] field to 0 on any writes to a channel's TCDn_CSR[7:0] after that channel's CHn_CSR[DONE] field is set to 1, indicating the major loop is complete.

NOTE

You must clear the CHn_CSR[DONE] field to 0 before writing to the TCDn_CSR[MAJORELINK] field. The CHn_CSR[DONE] field is cleared to 0 automatically by the eDMA engine after a channel begins execution.

12.6.8.3 Dynamic scatter/gather

Scatter/gather is the process of automatically loading a new TCD into a channel. It allows a DMA channel to use multiple TCDs; this enables a DMA channel to scatter the DMA data to multiple destinations or gather it from multiple sources. When scatter/gather is enabled and the channel has finished its major loop, a new TCD is fetched from system memory and loaded into that channel's descriptor location in the eDMA programmer's model, thus replacing the current descriptor.

Because you are allowed to change the configuration during execution, you need a coherency model. Consider the scenario where you attempt to execute a dynamic scatter/gather operation by enabling the TCDn_CSR[ESG] field at the same time the eDMA engine is retiring the channel. The TCDn_CSR[ESG] field would be set in the programmer's model, but it would be unclear whether the actual scatter/gather request was honored before the channel retired.

Two methods are recommended for executing a dynamic scatter/gather request. Whenever the TCDn_CSR is written, the TCD local memory controller forces the TCDn_CSR[ESG] field to 0 on any writes to a channel's TCDn_CSR[7:0] after that channel's CHn_CSR[DONE] field has been set to 1, indicating the major loop is complete. If attempting to set the ESG, ensure the DONE field is cleared to 0.

NOTE

You must clear the CHn_CSR[DONE] field to 0 before writing the TCDn_CSR[MAJORELINK] or TCDn_CSR[ESG] fields. The CHn_CSR[DONE] field is cleared to 0 automatically by the eDMA engine after a channel begins execution and is set to 1 upon major loop completion.

12.6.8.3.1 Method 1 (channel not using major loop channel linking)

For a channel not using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request.

When the TCDn_CSR[MAJORELINK] field is 0, the TCDn_CSR[MAJORLINKCH] field is not used by the eDMA. In this case, the TCDn_CSR[MAJORLINKCH] bits may be used for other purposes. This method uses the TCDn_CSR[MAJORLINKCH] field as a TCDn_CSR identification (ID).

When the descriptors are built, write a unique TCDn_CSR ID in the TCDn_CSR[MAJORLINKCH] field for each TCDn_CSR associated with a channel using dynamic scatter/gather.

1. Write a 1 to the TCDn_CSR[DREQ] field. Should a dynamic scatter/gather attempt fail, setting the TCDn_CSR[DREQ] field to 1 will prevent future hardware activation of this channel. This stops the channel from executing with a destination address (daddr) that was calculated using a scatter/gather address (written in the next step) instead of a DLAST final offset value.
2. Write the TCDn_DLAST_SGA field with the scatter/gather address.
3. Write a 1 to the TCDn_CSR[ESG] field.
4. Read back the 16-bit TCDn_CSR control/status field.
5. Test the TCDn_CSR[ESG] request status and TCDn_CSR[MAJORLINKCH] value:
   - If ESG = 1, the dynamic scatter/gather attempt was successful.

- If ESG = 0 and the MAJORLINKCH (ID) did not change, the dynamic scatter/gather attempt was not successful (the channel was already retiring).
- If ESG = 0 and the MAJORLINKCH (ID) changed, the dynamic scatter/gather attempt was successful (the new TCDn_CSR's ESG value cleared the ESG field to 0).

12.6.8.3.2 Method 2 (channel using major loop channel linking)

For a channel using major loop channel linking, the coherency model described here may be used for a dynamic scatter/gather request. This method uses the TCDn_DLAST_SGA field as a TCD identification (ID).

1. Write a 1 to the TCDn_CSR[DREQ] field. Should a dynamic scatter/gather attempt fail, setting the DREQ field to 1 will prevent a future hardware activation of this channel. This stops the channel from executing with a destination address (DADDR) that was calculated using a scatter/gather address (written in the next step) instead of a DLAST final offset value.
2. Write the TCDn_DLAST_SGA field with the scatter/gather address.
3. Write a 1 to the TCDn_CSR[ESG] field.
4. Read back the TCDn_CSR[ESG] field.
5. Test the TCDn_CSR[ESG] request status:
   - If ESG = 1, the dynamic scatter/gather attempt was successful.
   - If ESG = 0, read the 32-bit TCDn_DLAST_SGA field.
   - If ESG = 0 and the TCDn_DLAST_SGA did not change, the dynamic scatter/gather attempt was not successful (the channel was already retiring).
   - If ESG = 0 and the TCDn_DLAST_SGA changed, the dynamic scatter/gather attempt was successful (the new TCDn_CSR's ESG value cleared the ESG field to 0).

12.6.9 Lockstep

This chip contains an eDMA Checker which, when enabled, operates in Lockstep mode with the primary eDMA, executing the exact same transfer profiles. The eDMA Checker checks to ensure the primary eDMA is operating correctly. A block diagram is shown below.

Provided under IDA only, COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-4821-48a8-b251-390d3ab624

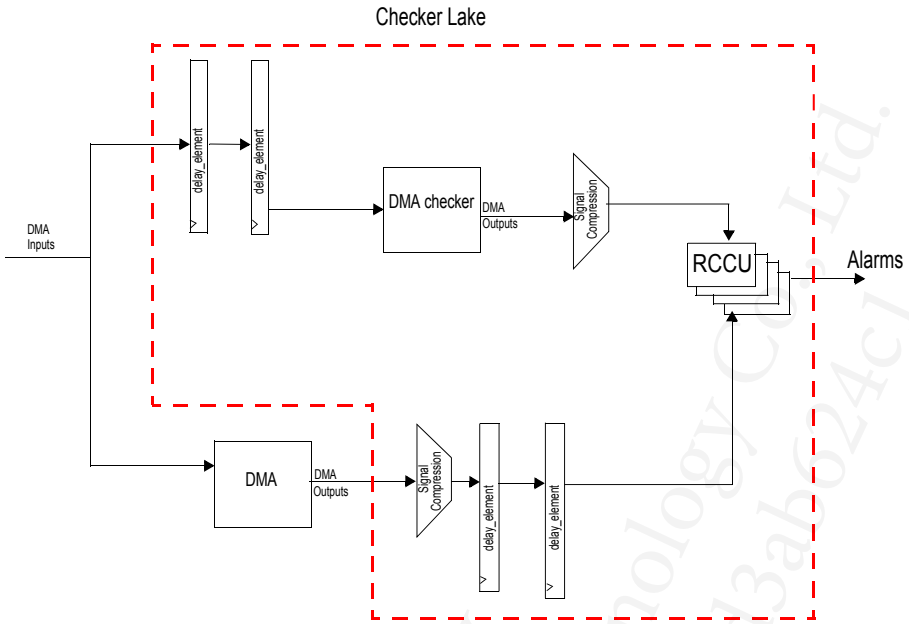


Figure 28. Lockstep block diagram

Initialization

To prevent the eDMA Checker from generating a false comparison error after power-on-reset, the eDMA's data output buffer must be initialized prior to enabling the Checker. To initialize the eDMA data output buffer, the eDMA must perform one or more 64-bit data transfers using any data value.

Errors

To handle source-address errors (SAE), perform the following steps.

1. Fix the addressing error.
2. Clear the ERR bit in the appropriate Channel Error Status (CH0_ES - CH31_ES). Management Page Error Status (ES) indicates the presence, location, and type of the last recorded channel error.
3. Start the channel in a normal fashion.

12.6.10 Suspend/resume a DMA channel with active hardware service requests

The DMA allows you to move data from memory or peripheral registers to another location in memory or to peripheral registers without CPU interaction. After the DMA and peripherals are configured and active, it is rare but supported to suspend a peripheral's service request dynamically. In this scenario, there are certain restrictions to disabling a DMA hardware service request. For coherency, you must follow a specific procedure. This section provides guidance on how to coherently suspend and resume a Direct Memory Access (DMA) channel when the DMA is triggered by a slave module such as the Serial Peripheral Interface (SPI), Sigma Delta Analog to Digital Convertor (SDADC), or other module.

12.6.10.1 Suspend an active DMA channel

To suspend an active DMA channel:

1. Stop the DMA service request at the peripheral first. Confirm it has been disabled by reading back the appropriate register in the peripheral.
2. Check the DMA's Hardware Request Status (HRS) to ensure there is no service request to the DMA channel being suspended. Then disable the hardware service request by clearing the ERQ field to 0 on the appropriate DMA channel.

For example, assume the SPI is set as a master for transmitting data via a DMA service request when the TXFIFO has an empty slot. The DMA will transfer the next command and data to the TXFIFO upon the request. If you need to suspend the DMA/SPI transfer loop, perform the following steps:

1. Disable the DMA service request at the source by writing 0 to DSPI_RSER[TFFF_RE]. Confirm that DSPI_RSER[TFFF_RE] is 0.
2. Ensure there is no DMA service request from the SPI by verifying that HRS[HRS] is 0 for the appropriate channel. If no service request is present, disable the DMA channel by clearing the channel's ERQ field to 0. If a service request is present, wait until the request has been processed and the HRS field reads 0.

12.6.10.2 Resume a DMA channel

To resume a DMA channel:

1. Enable the DMA service request on the appropriate channel by setting its ERQ field to 1.
2. Enable the DMA service request at the peripheral.

12.7 Glossary

TCD	Transfer control descriptor

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 13
DMA CRC Module (DMA_CRC)

13.1 Introduction

This section provides an overview of the CRC unit and DMA Controller. The CRC unit is capable of calculating the CRC checksum in parallel with the DMA transfers. The CRC unit supports up to eight out of 64 DMA channels that are selectable through a programmable register. The CRC calculation is performed on the read data transfers. It is cumulatively calculated with each transfer until the DMA transfer is completed. The DMA CRC module supports aligned 32-bits and 64-bits data transfers. The unaligned transfer produces unpredictable checksum. All control/status registers inside the unit support the 32-bit peripheral interface protocol.

13.1.1 Block diagram

Here is the block diagram of the DMA_CRC module:

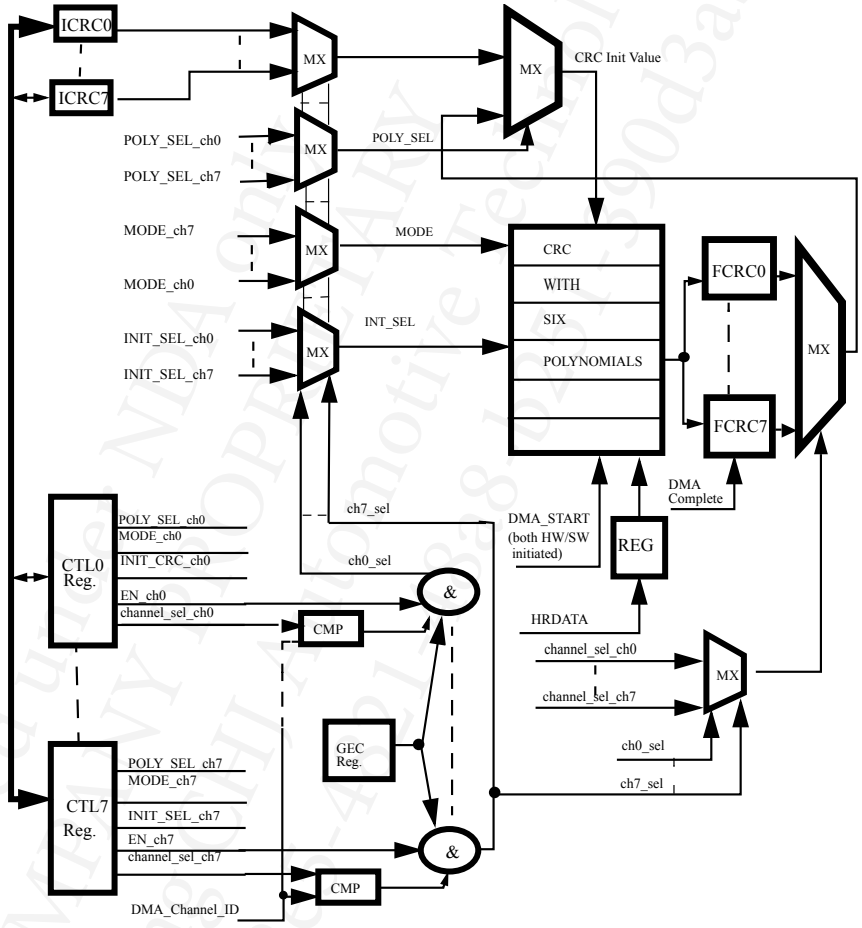


Figure 29. Block diagram

13.1.2 Features

The following list summarizes the key features of the DMA_CRC module:

- Contains peripheral interface for accessing CRC's control and status registers
- Supports up to eight channels

- Supports up to six different polynomials (8-bit, 16-bit, and 32-bit). Each channel can have its own polynomial or share the same polynomial. The supported polynomials are:
  - Ethernet (IEEE 802,3) CCITT32 CRC-32 0x04C11DB7
  - CASTAGNOLI, iSCSI CRC-32C 0x1EDC6F41
  - E2E Profile 4 CRC-32 P4 0xF4ACFB13
  - x.25. CCITT16 CRC-16 0x1021
  - H2F, AUTOSAR 4.0 CRC-8 0x2F
  - VDA CAN, SAE-J1850 CRC-8 0x1D
- Can enable/disable CRC functionality per channel
- Can use global CRC enable to safeguard against the accidental enablement
- Can preload CRC register at the start of the DMA transfers with an initial value (ICRC register) or continue with the accumulated CRC value from previous read transactions. The software can preload the ICRC register with a pre-calculated checksum value and check for a zero-checksum value at the end of the DMA transfers
- Makes the intermediate and final result of the CRC accessible through the Final CRC Value Register (FCRC). Each channel has its own FCRC register
- Supports all capabilities of the current eDMA, such as TCDs, channel linking, scatter-gather, and so on

13.1.3 Mode of operation

The DMA_CRC module supports only the normal mode. It uses CTL[MODE] to select the normal mode. In the normal mode, the DMA controller performs its normal operation which is reading from its source address and writing to its destination address. The CRC checksum is calculated on the read data (HRDATA[63:0] or HRDATA[31:0]).

NOTE

For this device, the data used in the CRC computation is in little endian.

13.2 DMA_CRC register descriptions

The DMA_CRC module provides a programming model mapped to a standard 16 KB on-platform peripheral slot.

The programming model can be referenced using a 32-, 16-, or 8-bit accesses. The attempted references to undefined (reserved) addresses generate an error termination. Each DMA channel with CRC provides three architected registers: Initial CRC Value Register (ICRC), Final CRC Value Register (FCRC), and CRC Control Register (CTL). In addition, an architected register, Global Enable CRC Register (GEC), is present to enable the CRC logic globally on all the eight channels.

13.2.1 DMA_CRC memory map

DMA_CRC_0 base address: 4013_C000h

DMA_CRC_1 base address: 4023_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Global Enable CRC Register (GEC)	32	RW	0000_0000h
10h	CRC Control Register (CTL0)	32	RW	0000_0000h
14h	Initial CRC Value Register (ICRC0)	32	RW	FFFF_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
18h	Final CRC Value Register (FCRC0)	32	RO	0000_0000h
20h	CRC Control Register (CTL1)	32	RW	0000_0000h
24h	Initial CRC Value Register (ICRC1)	32	RW	FFFF_FFFFh
28h	Final CRC Value Register (FCRC1)	32	RO	0000_0000h
30h	CRC Control Register (CTL2)	32	RW	0000_0000h
34h	Initial CRC Value Register (ICRC2)	32	RW	FFFF_FFFFh
38h	Final CRC Value Register (FCRC2)	32	RO	0000_0000h
40h	CRC Control Register (CTL3)	32	RW	0000_0000h
44h	Initial CRC Value Register (ICRC3)	32	RW	FFFF_FFFFh
48h	Final CRC Value Register (FCRC3)	32	RO	0000_0000h
50h	CRC Control Register (CTL4)	32	RW	0000_0000h
54h	Initial CRC Value Register (ICRC4)	32	RW	FFFF_FFFFh
58h	Final CRC Value Register (FCRC4)	32	RO	0000_0000h
60h	CRC Control Register (CTL5)	32	RW	0000_0000h
64h	Initial CRC Value Register (ICRC5)	32	RW	FFFF_FFFFh
68h	Final CRC Value Register (FCRC5)	32	RO	0000_0000h
70h	CRC Control Register (CTL6)	32	RW	0000_0000h
74h	Initial CRC Value Register (ICRC6)	32	RW	FFFF_FFFFh
78h	Final CRC Value Register (FCRC6)	32	RO	0000_0000h
80h	CRC Control Register (CTL7)	32	RW	0000_0000h
84h	Initial CRC Value Register (ICRC7)	32	RW	FFFF_FFFFh
88h	Final CRC Value Register (FCRC7)	32	RO	0000_0000h

13.2.2 Global Enable CRC Register (GEC)

Offset

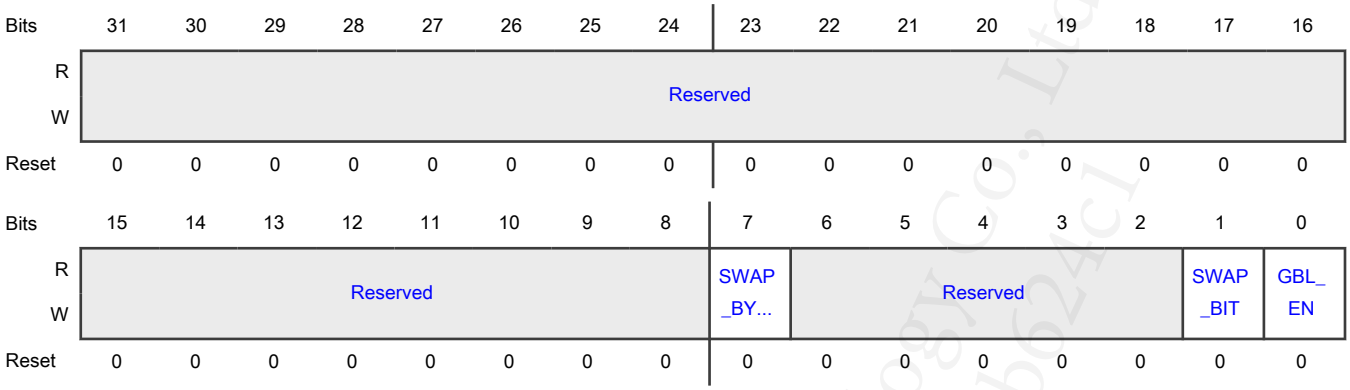
Register	Offset
GEC	0h

Function

The bit 0 of the Global Enable CRC Register (GEC) enables the CRC logic in all the eight channels. A write to any of the CRC Control Register (CTLn) clears this bit and disables the CRC logic. This bit should be set as the last step in enabling the CRC logic.

This register safeguards against the accidental enablement of the CRC function or accidental configuration update of the CTLn in the middle of a DMA transfer.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 SWAP_BYTE	Swap Byte This field swap the DMA data byte-wise. 0b - Do not swap. 1b - Byte-wise swap on the input data.
6-2 —	Reserved
1 SWAP_BIT	Swap Bit This field swap the DMA data bit-wise. 0b - Do not swap. 1b - Bit-wise swap on the input data.
0 GBL_EN	Global Enable bit This bit globally enables/disables CRC logic in all channels. Writing to any of the CTLn register clears this bit. 0b - Disable CRC in all channels. 1b - Enable CRC in all channels.

13.2.3 CRC Control Register (CTL0 - CTL7)

Offset

For a = 0 to 7:

Register	Offset
CTLa	10h + (a × 10h)

Function

The CRC Control Register (CTLn) allows the user to set the following functions (n indicates channel 0 through 7):

- Selecting one of the 64 channels to monitor
- Selecting one of the six available polynomials
- Selecting an initial seed value of the CRC
- Enabling/disabling the CRC logic
- Selecting the CRC mode

To initiate the CRC calculation on the read data, this control register must be set up prior to any DMA transfer. Updating the CTL register in the middle of the DMA transfers results in non-deterministic behavior and corrupts the CRC checksum. This register can be accessed in byte, half-word, or word. The CRC control register must be set prior to selecting the functionalities of the CRC, such as enabling/disabling it, selecting the polynomial, preloading of the CRC, and so on.

NOTE

Writing to the reserved bits does not generate error and is ignored. Reading the reserved bits returns zeroes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EN	Reserved												MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	INIT_SEL	Reserved			POLY_SEL				Reserved		CH_SEL					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EN	CRC Logic This bit enables/disables CRC logic. 0b - Disable CRC. 1b - Enable CRC.
30-19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18-16 MODE	CRC Mode These bits are used to select the CRC mode. The normal CRC mode where the DMA does a read transfer on the source address, a write transfer on the destination address, and a CRC checksum is calculated on the read data. 000b - Normal CRC Mode. All other combinations are invalid.
15 INIT_SEL	Initial values of the CRC This bit is used to select the initial values of the CRC at the start of a DMA transfer. 0b - Initialize CRC with the content of the Initial CRC Value Register (ICRC). 1b - Continue accumulating previous CRC values stored in Final CRC Value Register (FCRC).
14-12 —	Reserved
11-8 POLY_SEL	Polynomial Select Values written to this field determine which polynomial to use for CRC calculation. Setting these bits to 3'b110 or 3'b111 gives a non-deterministic checksum result. 0000b - Select Ethernet (IEEE 802,3) CCITT32 CRC-32 0x04C11DB7. 0001b - Select CASTAGNOLI, iSCSICRC-32C 0x1EDC6F41. 0010b - Select E2E Profile 4 CRC-32 P4 0xF4ACFB13. 0011b - Select x.25. CCITT16 CRC-16 0x1021. 0100b - Select H2F, AUTOSAR 4.0 CRC-8 0x2F. 0101b - Select VDA CAN, SAE-J1850 CRC-8. 0110b - Reserved 0111b - Reserved
7-6 —	Reserved
5-0 CH_SEL	Channel Select These bits are compared with the DMA_CHANNEL_ID signals. A match along with assertion of EN bit enables the CRC logic on this channel. **NOTE** The CH_SEL[5:0] bits for CTL0 through CTL7 must be unique when the CRC is enabled; otherwise, the checksum value will be non-deterministic. 000000b - Select Channel 0. 000001b - Select Channel 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	111110b - Select Channel 62. 111111b - Select Channel 63.

13.2.4 Initial CRC Value Register (ICRC0 - ICRC7)

Offset

For a = 0 to 7:

Register	Offset
ICRCa	14h + (a × 10h)

Function

The Initial CRC Value Register (ICRC) is one of the options to initialize the CRC at the start of a DMA transfer. When CTL[INIT_SEL] is '1', the content of ICRC is used as the starting seed of the CRC. Software can use this feature to check for a zero-checksum value at the end of the DMA transfer by preloading the CRC with a certain value. The CTL[POLY_SEL] determines the values to be loaded in this register. Example, for a "16 bits polynomial x.25. CCITT16 CRC-16 0x1021", bits 15:0 are used to preload the CRC seed. The rest of the bits[31:16] are treated as don't care.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INI_CRC_VAL															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INI_CRC_VAL															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 INI_CRC_VAL	Initial CRC Value Contains the initial value of the CRC register. When CTL[INIT_SEL] is '1', the content of this register is preloaded into the CRC at the start of the DMA transfer.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c2e54-8214-8ad-b251-d3ab624c1

13.2.5 Final CRC Value Register (FCRC0 - FCRC7)

Offset

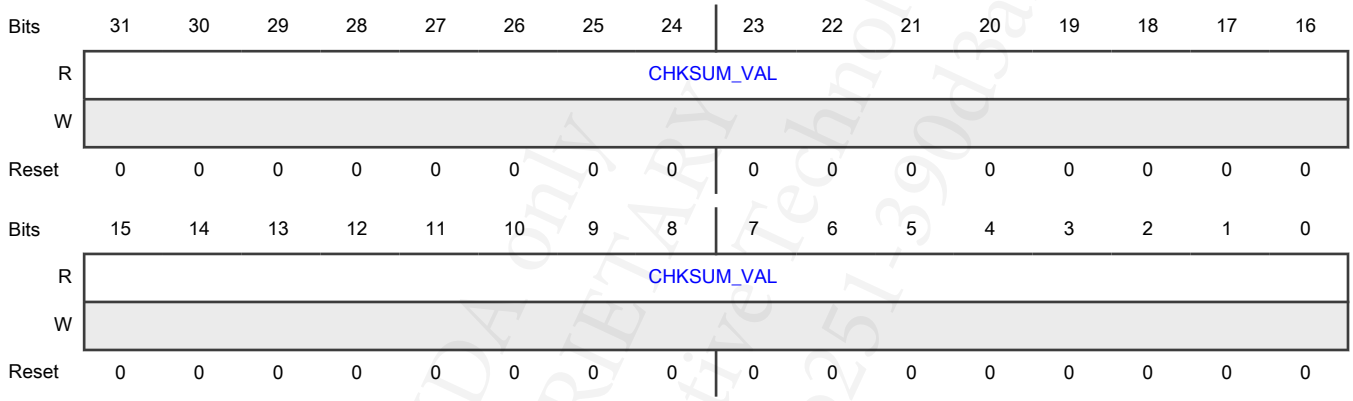
For a = 0 to 7:

Register	Offset
FCRCa	18h + (a × 10h)

Function

The Final CRC Value register holds the checksum at the end of the DMA transfer. In a channel linking, it is updated on the completion of every minor loop and major loop. The valid checksum bits depend on the size of the polynomial. Example, for an 8-bit polynomial H2F, AUTOSAR 4.0 CRC-8 0x2F, the resulting checksum resides in FCRC[7:0].

Diagram



Fields

Field	Function
31-0 CHKSUM_VAL	Final CRC Value Contains the final value of the CRC (Checksum)

13.3 Functional description

The CH_SEL[5:0] bits of CTL0 through CTL7 are compared to the incoming DMA_CHANNEL_ID[5:0] signal. When a match is detected and both the CTL[EN] and GEC[GBL_EN] bits for the matched channels are set, a CRC calculation is initiated on the incoming read data (HRDATA[63:0] or HRDATA[31:0]) depending on the size of data that the DMA transfers. The CRC calculation is performed on all the read data until the DMA engine completes its transfer, but excluding any transfer for TCD updates. The content of CH_SEL[5:0] bits for CTL0 through CTL7 must be unique if their EN bit is set or a non-deterministic checksum is generated. The match signal along with the assertion of the EN and CRC Global Enable signals selects other attributes corresponding to each channel, such as CTL[POLY_SEL], CTL[MODE], and CTL[INIT_SEL].

The CTL[POLY_SEL] selects one of the six polynomials. The non-selected polynomials should be kept static to minimize power. The CTL[POLY_SEL] bits should never be set to '110' and '111'; otherwise, a non-deterministic checksum is generated. The CRC is calculated by XORing the registered values of HRDATA[63:0] or HRDATA[31:0] with the seed value of CRC selected by INIT_SEL.

The checksum value is stored in the Final CRC Value Register (FCRC). For a 16-bit polynomials, the checksum is stored in FCRC[15:0], and for 8-bit polynomials, it is stored in FCRC[7:0]. The software must verify the correctness of the checksum

value in the Final CRC Value register. The checksum calculation for 64-bit or 32-bit data using any polynomial completes in one clock cycle.

Examples of input data and expected output checksum:

Initial value ICRC [31:0] = FFFFFFFFh

SWAP_BIT = 0

SWAP_BYTE = 1 or littleEndian

Final XOR value: 32'h0

Table 47. Example of input data and expected output checksum

DMA read data		A1B2C3D4h	11223344h
Polynomial (CTL)		Checksum (FCRC)	
CRC-32 Ethernet	04C11DB7h	2DA0A80Ch	B14257CCh
CRC-32 CASTAGNOLI	1EDC6F41h	ADFBCF70h	495AD03Ah
CRC-32 E2E	F4ACFB13h	B8E7F41Eh	9BFAE80Bh
CRC-16 x.25	1021h	4063h	59F3h
CRC-8 H2F	2Fh	3Bh	FCh
CRC-8 VDA	1Dh	D4h	5Fh

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8a1e2a51

Chapter 14
Direct Memory Access Multiplexer (DMAMUX)

14.1 Chip-specific DMAMUX information

14.1.1 DMAMUX-to-eDMA channel mapping

The following table shows the channel mapping between DMAMUX instances and eDMA instances.

DMAMUX instance channels	eDMA instance (lockstep pair) channels
DMAMUX_0 channels 0–15	eDMA_0 channels 0–15
DMAMUX_1 channels 0–15	eDMA_0 channels 16–31
DMAMUX_2 channels 0–15	eDMA_1 channels 0–15
DMAMUX_3 channels 0–15	eDMA_1 channels 16–31

NOTE

See DMA Map file attached to this document for details on the channel mapping.

14.2 Introduction

14.2.1 Overview

The Direct Memory Access Multiplexer (DMAMUX) routes DMA sources, called slots, to any of the 16 DMA channels. See the chip-specific information to know the detailed source numbers. This process is illustrated in the following figure.

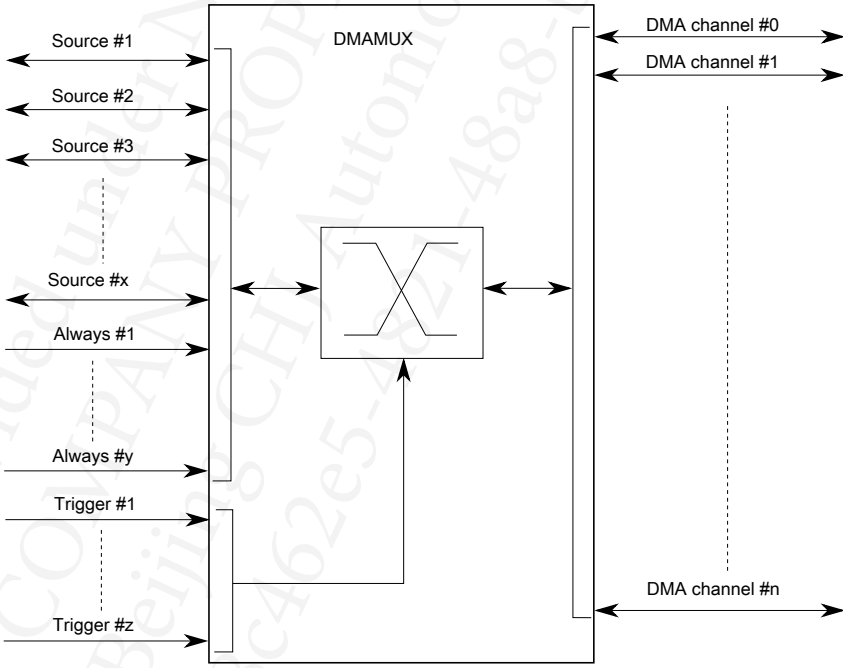


Figure 30. DMAMUX block diagram

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c462e5-4f21-48a8-b251-390d3b0c4af1

14.2.2 Features

Following are the features of DMAMUX module.

- Up to 59 peripheral slots and up to 4 always-on slots can be routed to 16 channels.
- 16 independently selectable DMA channel routers.
  - The first 5 channels additionally provide a trigger functionality.
- Each channel router can be assigned to one of the possible peripheral DMA slots or to one of the always-on slots.

14.2.3 Modes of operation

The DMAMUX module supports the following operating modes.

- Disabled mode: In this mode, the DMA channel is disabled. Because disabling and enabling of DMA channels is done primarily via the DMA configuration registers, this mode is used mainly as the reset state for a DMA channel in the DMA channel MUX. It may also be used to temporarily suspend a DMA channel while reconfiguration of the system takes place, for example, changing the period of a DMA trigger.
- Normal mode: In this mode, a DMA source is routed directly to the specified DMA channel. The operation of the DMAMUX in this mode is completely transparent to the system.
- Periodic Trigger mode: In this mode, a DMA source may only request a DMA transfer, such as when a transmit buffer becomes empty or a receive buffer becomes full, periodically.

  Configuration of the period is done by an external periodic interrupt timer (PIT). This mode is available only for channels 0–4.

14.3 Memory map/register definition

This section provides a detailed description of all memory-mapped registers in the DMAMUX.

14.3.1 DMAMUX register descriptions

14.3.1.1 DMAMUX memory map

DMAMUX_0 base address: 4012_C000h

DMAMUX_1 base address: 4013_0000h

DMAMUX_2 base address: 4022_C000h

DMAMUX_3 base address: 4023_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Channel Configuration register (CHCFG3)	8	RW	00h
1h	Channel Configuration register (CHCFG2)	8	RW	00h
2h	Channel Configuration register (CHCFG1)	8	RW	00h
3h	Channel Configuration register (CHCFG0)	8	RW	00h
4h	Channel Configuration register (CHCFG7)	8	RW	00h
5h	Channel Configuration register (CHCFG6)	8	RW	00h
6h	Channel Configuration register (CHCFG5)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
7h	Channel Configuration register (CHCFG4)	8	RW	00h
8h	Channel Configuration register (CHCFG11)	8	RW	00h
9h	Channel Configuration register (CHCFG10)	8	RW	00h
Ah	Channel Configuration register (CHCFG9)	8	RW	00h
Bh	Channel Configuration register (CHCFG8)	8	RW	00h
Ch	Channel Configuration register (CHCFG15)	8	RW	00h
Dh	Channel Configuration register (CHCFG14)	8	RW	00h
Eh	Channel Configuration register (CHCFG13)	8	RW	00h
Fh	Channel Configuration register (CHCFG12)	8	RW	00h

14.3.1.2 Channel Configuration register (CHCFG0 - CHCFG15)

Offset

Register	Offset
CHCFG3	0h
CHCFG2	1h
CHCFG1	2h
CHCFG0	3h
CHCFG7	4h
CHCFG6	5h
CHCFG5	6h
CHCFG4	7h
CHCFG11	8h
CHCFG10	9h
CHCFG9	Ah
CHCFG8	Bh
CHCFG15	Ch
CHCFG14	Dh
CHCFG13	Eh
CHCFG12	Fh

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
8c4625-4821-48a8-b251-390dabfa61

Function

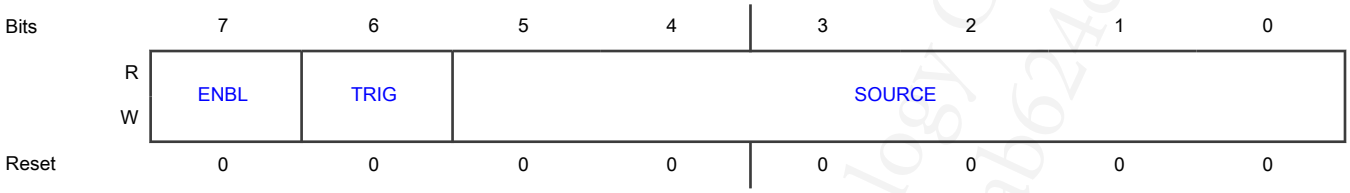
Each of the DMA channels can be independently enabled/disabled and associated with one of the DMA slots (peripheral slots or always-on slots) in the system.

NOTE

Setting multiple CHCFG registers with the same source value will result in unpredictable behavior. This is true, even if a channel is disabled (ENBL==0).

Before changing the trigger or source settings, a DMA channel must be disabled via CHCFGn[ENBL].

Diagram



Fields

Field	Function
7 ENBL	DMA Channel Enable Enables the DMA channel. 0b - DMA channel is disabled. This mode is primarily used during configuration of the DMAMUX. The DMA has separate channel enables/disables, which should be used to disable or reconfigure a DMA channel. 1b - DMA channel is enabled
6 TRIG	DMA Channel Trigger Enable Enables the periodic trigger capability for the triggered DMA channel. 0b - Triggering is disabled. If triggering is disabled and ENBL is set, the DMA Channel will simply route the specified source to the DMA channel. (Normal mode) 1b - Triggering is enabled. If triggering is enabled and ENBL is set, the DMAMUX is in Periodic Trigger mode.
5-0 SOURCE	DMA Channel Source (Slot) Specifies which DMA source, if any, is routed to a particular DMA channel. See the chip-specific DMAMUX information for details about the peripherals and their slot numbers.

14.4 Functional description

The primary purpose of the DMAMUX is to provide flexibility in the system's use of the available DMA channels. As such, configuration of the DMAMUX is intended to be a static procedure done during execution of the system boot code. However, if the procedure outlined in Enabling and configuring sources is followed, the configuration of the DMAMUX may be changed during the normal operation of the system.

Functionally, the DMAMUX channels may be divided into two classes:

- Channels that implement the normal routing functionality plus periodic triggering capability
- Channels that implement only the normal routing functionality

14.4.1 DMA channels with periodic triggering capability

Besides the normal routing functionality, the first five channels of the DMAMUX provide a special periodic triggering capability that can be used to provide an automatic mechanism to transmit bytes, frames, or packets at fixed intervals without the need for processor intervention.

The trigger is generated by an external periodic interrupt timer (PIT); as such, the configuration of the periodic triggering interval is done by configuring the external periodic timer.

NOTE

Because of the dynamic nature of the system (due to DMA channel priorities, bus arbitration, interrupt service routine lengths, etc.), the number of clock cycles between a trigger and the actual DMA transfer cannot be guaranteed.

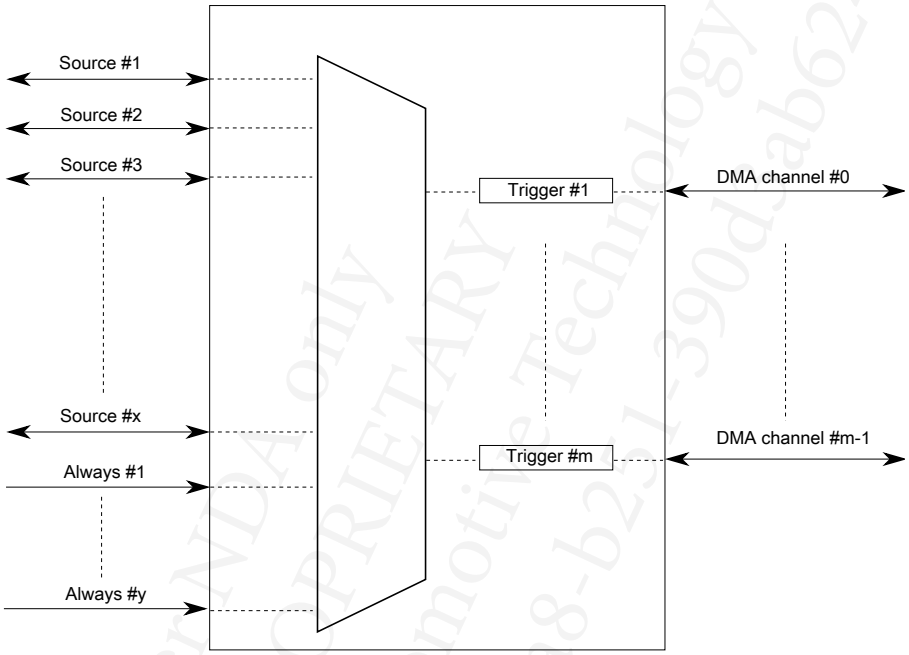


Figure 31. DMAMUX triggered channels

The DMA channel triggering capability allows the system to schedule regular DMA transfers, usually on the transmit side of certain peripherals, without the intervention of the processor. This trigger works by gating the request from the peripheral to the DMA until a trigger event has been seen. This is illustrated in the following figure.

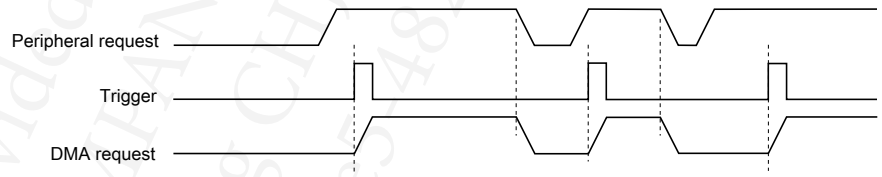


Figure 32. DMAMUX channel triggering: normal operation

After the DMA request has been serviced, the peripheral negates its request, effectively resetting the gating mechanism until the peripheral reasserts its request and the next trigger event is seen. This means that if a trigger is seen, but the peripheral is not requesting a transfer, then that trigger will be ignored. This situation is illustrated in the following figure.

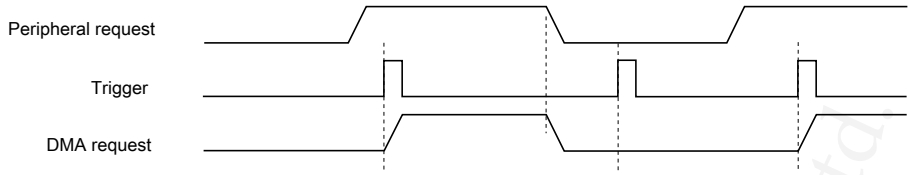


Figure 33. DMAMUX channel triggering: ignored trigger

This triggering capability may be used with any peripheral that supports DMA transfers, and is most useful in the following two situations.

- Periodically polling external devices on a particular bus: As an example, the transmit side of an SPI is assigned to a DMA channel with a trigger, as described above. After it has been set up, the SPI requests DMA transfers, presumably from memory, as long as its transmit buffer is empty. By using a trigger on this channel, the SPI transfers can be automatically performed every 5 µs (as an example). On the receive side of the SPI, the SPI and DMA can be configured to transfer receive data into memory, effectively implementing a method to periodically read data from external devices and transfer the results into memory without processor intervention.
- Using the GPIO ports to drive or sample waveforms: By configuring the DMA to transfer data to one or more GPIO ports, it is possible to create complex waveforms using tabular data stored in on-chip memory. Conversely, using the DMA to periodically transfer data from one or more GPIO ports, it is possible to sample complex waveforms and store the results in tabular form in on-chip memory.

14.4.2 DMA channels with no triggering capability

The other channels of the DMAMUX provide the normal routing functionality as described in Modes of operation.

14.4.3 Always-enabled DMA sources

In addition to the peripherals that can be used as DMA sources, there are 4 additional DMA sources that are always enabled. Unlike the peripheral DMA sources, where the peripheral controls the flow of data during DMA transfers, the sources that are always enabled provide no such "throttling" of the data transfers. These sources are most useful in the following cases:

- Performing DMA transfers to/from GPIO—Moving data from/to one or more GPIO pins, either unthrottled (that is, as fast as possible), or periodically (using the DMA triggering capability).
- Performing DMA transfers from memory to memory: Moving data from memory to memory, typically as fast as possible, sometimes with software activation.
- Performing DMA transfers from memory to the external bus, or vice-versa: Similar to memory to memory transfers, this is typically done as quickly as possible.
- Any DMA transfer that requires software activation: Any DMA transfer that should be explicitly started by software.

In cases where software should initiate the start of a DMA transfer, an always-enabled DMA source can be used to provide maximum flexibility. When activating a DMA channel via software, subsequent executions of the minor loop require that a new start event be sent. This can either be a new software activation, or a transfer request from the DMA channel MUX. The options for doing this are as follows.

- Transfer all data in a single minor loop: By configuring the DMA to transfer all of the data in a single minor loop (that is, major loop counter = 1), no reactivation of the channel is necessary. The disadvantage to this option is the reduced granularity in determining the load that the DMA transfer will impose on the system. For this option, the DMA channel must be disabled in the DMA channel MUX.
- Use explicit software reactivation: In this option, the DMA is configured to transfer the data using both minor and major loops, but the processor is required to reactivate the channel by writing to the DMA registers after every minor loop. For this option, the DMA channel must be disabled in the DMA channel MUX.
- Use an always-enabled DMA source: In this option, the DMA is configured to transfer the data using both minor and major loops, and the DMA channel MUX does the channel reactivation. For this option, the DMA channel should be enabled and

pointing to an "always enabled" source. Note that the reactivation of the channel can be continuous (DMA triggering is disabled) or can use the DMA triggering capability. In this manner, it is possible to execute periodic transfers of packets of data from one source to another, without processor intervention.

14.5 Initialization/application information

This section provides instructions for initializing the DMA channel MUX.

14.5.1 Reset

The reset state of each individual bit is shown in Memory map/register definition. In summary, after reset, all channels are disabled and must be explicitly enabled before use.

14.5.2 Enabling and configuring sources

To enable a source with periodic triggering:

1. Determine the DMA channel with which the source will be associated. Note that only the first five DMA channels have periodic triggering capability.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] fields of the DMA channel.
3. Ensure that the DMA channel is properly configured in the DMA. The DMA channel may be enabled at this point.
4. Configure the corresponding timer.
5. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that the CHCFG[ENBL] and CHCFG[TRIG] fields are set.

NOTE

The following is an example. See the chip-specific information for the number of this device's DMA channels that have triggering capability.

To configure source #5 transmit for use with DMA channel 1, with periodic triggering capability:

1. Write 0x00 to CHCFG1.
2. Configure channel 1 in the DMA, including enabling the channel.
3. Configure a timer for the desired trigger interval.
4. Write 0xC5 to CHCFG1.

The following code example illustrates steps 1 and 4 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR     0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
```

```
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);

In File main.c:
#include "registers.h"
:
:
*CHCFG1 = 0x00;                                 /* Clear all the fields of CHCFG1 register */
*CHCFG1 = 0xC5;                                 /* ENBL = 1  DMA Channel is enabled */
                                                /* TRIG = 1  Triggering is enabled */
                                                /* SOURCE = 5  DMA Source 5 is selected */
```

To enable a source, without periodic triggering:

1. Determine the DMA channel with which the source will be associated. Note that only the first five DMA channels have periodic triggering capability.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] fields of the DMA channel.
3. Ensure that the DMA channel is properly configured in the DMA. The DMA channel may be enabled at this point.
4. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that CHCFG[ENBL] is setwhile CHCFG[TRIG] is cleared.

NOTE

The following is an example. See the chip configuration details for the number of this device's DMA channels that have triggering capability.

To configure source #5 transmit for use with DMA channel 1, with no periodic triggering capability:

1. Write 0x00 to CHCFG1.
2. Configure channel 1 in the DMA, including enabling the channel.
3. Write 0x85 to CHCFG1.

The following code example illustrates steps 1 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR    0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);
```

```
In File main.c:
#include "registers.h"
:
:
*CHCFG1 = 0x00;                                 /* Clear all the fields of CHCFG1 register */
*CHCFG1 = 0x85;                                 /* ENBL = 1  DMA Channel is enabled */
                                                /* TRIG = 0  Triggering is disabled */
                                                /* SOURCE = 5  DMA Source 5 is selected */
```

To disable a source:

A particular DMA source may be disabled by not writing the corresponding source value into any of the CHCFG registers. Additionally, some module-specific configuration may be necessary. See the eDMA chapter for more details.

To switch the source of a DMA channel:

1. Disable the DMA channel in the DMA and reconfigure the channel for the new source.
2. Clear the CHCFG[ENBL] and CHCFG[TRIG] bits of the DMA channel.
3. Select the source to be routed to the DMA channel. Write to the corresponding CHCFG register, ensuring that the CHCFG[ENBL] and CHCFG[TRIG] fields are set.

To switch DMA channel 8 from source #5 transmit to source #7 transmit:

1. In the DMA configuration registers, disable DMA channel 8 and reconfigure it to handle the transfers to peripheral slot 7. This example assumes channel 8 doesn't have triggering capability.
2. Write 0x00 to CHCFG8.
3. Write 0x87 to CHCFG8. (In this example, setting CHCFG[TRIG] would have no effect due to the assumption that channel 8 does not support the periodic triggering functionality.)

The following code example illustrates steps 2 and 3 above:

```
In File registers.h:
#define DMAMUX_BASE_ADDR    0x40021000/* Example only ! */
/* Following example assumes char is 8-bits */
volatile unsigned char *CHCFG0 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0003);
volatile unsigned char *CHCFG1 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0002);
volatile unsigned char *CHCFG2 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0001);
volatile unsigned char *CHCFG3 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0000);
volatile unsigned char *CHCFG4 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0007);
volatile unsigned char *CHCFG5 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0006);
volatile unsigned char *CHCFG6 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0005);
volatile unsigned char *CHCFG7 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0004);
volatile unsigned char *CHCFG8 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000B);
volatile unsigned char *CHCFG9 = (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000A);
volatile unsigned char *CHCFG10= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0009);
volatile unsigned char *CHCFG11= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x0008);
volatile unsigned char *CHCFG12= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000F);
volatile unsigned char *CHCFG13= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000E);
volatile unsigned char *CHCFG14= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000D);
volatile unsigned char *CHCFG15= (volatile unsigned char *) (DMAMUX_BASE_ADDR+0x000C);

In File main.c:
#include "registers.h"
:
:
```

```
*CHCFG8 = 0x00;                                 /* Clear all the fields of CHCFG1 register */
*CHCFG8 = 0x87;                                 /* ENBL = 1  DMA Channel is enabled */
                                                /* TRIG = 0  Triggering is disabled */
                                                /* SOURCE = 7  DMA Source 7 is selected */
```

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 15
Extended Resource Domain Controller (XRDC)

15.1 Chip-specific XRDC_0 information

15.1.1 XRDC instances

This chip has two XRDC instances:

- XRDC_0 is the system instance.
- XRDC_1 is the Accelerator subsystem instance.

Features shared by both instances include:

- Access rights computed by MRCs and PACs can be determined by the SEMA42 module (hardware semaphore).
- The XRDCs have no interrupts of their own. An access violation triggers a Data Abort exception on the core performing the access.

NOTE

In this chapter:

- SDAC and MRC are interchangeable terms.
- PDAC and PAC are interchangeable terms.

15.1.2 XRDC_0 submodule configurations

XRDC consists of multiple submodules. See the "Block diagram" section for a list and descriptions of the submodules.

The following tables summarize the submodules and their configurations in this chip.

- MDAC configurations
- MRC configurations
- PAC configurations

The global XRDC configuration defines eight domains. Unimplemented domain numbers read as zero, provide no access, and cannot be written.

Additional global configuration includes:

- The width of the PIDs is defined as the low-order 6 bits.
- The implemented number of low-order address bits included in the memory region descriptor start and end address evaluations is 20.
- Upper address bits in the start address are read as zeros and cannot be written. In the end address, they read as ones and cannot be written.

15.1.2.1 MDAC configurations

Table 48. MDACn configurations

Submodule instance	Configuration	Number of MDA_W*x*_*y*_DFMT*z* registers	Bus master	Non-secure input[1]
XRDC_MDAC0	Processor	8	Cortex-A53 C0	Input from master

Table continues on the next page...

Table 48. MDACn configurations (continued)

Submodule instance	Configuration	Number of MDA_W*x*_*y*_DFMT*z* registers	Bus master	Non-secure input[1]
XRDC_MDAC1	Processor	8	Cortex-A53 C1	Input from master
XRDC_MDAC2	Nonprocessor	1	Debug ETR	Input from master
XRDC_MDAC3	Nonprocessor	1	GIC-500	Input from master
XRDC_MDAC4	Nonprocessor	1	Reserved	1
XRDC_MDAC5	Nonprocessor	1	SerDes_0	Input from master
XRDC_MDAC6	Nonprocessor	1	eDMA_0	1
XRDC_MDAC7	Nonprocessor	1	eDMA_1	1
XRDC_MDAC8	Processor	8	M7_0 AXI	Input from master(0)
XRDC_MDAC9	Processor	8	M7_1 AXI	Input from master(0)
XRDC_MDAC10	Processor	8	M7_2 AXI	Input from master(0)
XRDC_MDAC11	Nonprocessor	1	HSE_H	Input from master
XRDC_MDAC12	Nonprocessor	1	GMAC_0 (Ethernet)	Input from master
XRDC_MDAC15	Nonprocessor	1	FlexRay	1
XRDC_MDAC16	Processor	8	M7_0 AHB	0
XRDC_MDAC17	Processor	8	M7_1 AHB	0
XRDC_MDAC18	Processor	8	M7_2 AHB	0
XRDC_MDAC19	Nonprocessor	1	LLCE	1
XRDC_MDAC20	Nonprocessor	1	uSDHC	Input from master
XRDC_MDAC21	Nonprocessor	1	Debug trace	Input from master

1. The MDAC has a non-secure input for the bus master. Entries in this column mean:
   - 1: The bus master is always a non-secure input.
   - 0: The bus master is always a secure input.
   - Input from master: The bus master determines whether the input is 1 or 0.

NOTE

Input DID is 0 for all masters except DMA MDAC which takes input from DMA.

15.1.2.2 MRC configurations

Table 49. MRCn configurations

Submodule instance	Region format	Number of region descriptors	Slaves protected, module (port number)
XRDC_MRC0	Auto	16	DRAM0_A53(0), DRAM0_M7_EDMA(1), DRAM0_GENERAL(2)

Table continues on the next page...

Table 49. MRCn configurations (continued)

Submodule instance	Region format	Number of region descriptors	Slaves protected, module (port number)
XRDC_MRC1	Auto	16	unused
XRDC_MRC2	Auto	16	SRAM_1(0), SRAM_2(1), SRAM_3(2), SRAM_0(3)
XRDC_MRC3	Auto	16	SRAM_4(1), SRAM_5(0), SRAM_6(2), SRAM_7(3)
XRDC_MRC4	Auto	16	SRAM_9(0), SRAM_8(1), SRAM_11(2), SRAM_10(3)
XRDC_MRC5	Auto	16	SRAM_12(0), SRAM_13(1), SRAM_14(2), SRAM_15(3)
XRDC_MRC6	Auto	16	Ncore registers(1) and coherent accesses of SerDes_0, GMAC_0, and other masters with access to this slave port(0)
XRDC_MRC7	Auto	16	S_FLASH(0)(external)
XRDC_MRC8	Auto	16	S_LLCE(0)
XRDC_MRC9	Auto	16	S_GIC-500(0)
XRDC_MRC10	Auto	16	S_SerDes(0)
XRDC_MRC11	Auto	12	M7_TCM_0(0), M7_TCM_1(0), and M7_TCM_2(0)
XRDC_MRC12	Auto	4	NoC_0 registers(0)
XRDC_MRC13	Auto	16	S_DBG_APB(0)

NOTE

- For MRC2/3, SRAM{0-7} work together to support the interleaved SRAM scheme and are mapped to the 4MB address range 0x34000000 – 0x343FFFFF.
- For MRC4/5, SRAM{8-15} work together to support the interleaved SRAM scheme and are mapped to the 4MB address range 0x34400000 – 0x347FFFFF.

15.1.2.3 PAC configurations

Table 50. PACn configurations

Module name	Slaves protected
XRDC_PAC0	Peripheral group 0
XRDC_PAC1	Peripheral group 1
XRDC_PAC2	Peripheral group 2
XRDC_PAC3	Peripheral group 3

For details about the modules in each peripheral group, see the chip's memory map.

15.1.3 XRDC_0 MRCs and SRAM interleaving

Due to the granularity of SRAM interleaving, MRC2,MRC3 and MRC4,MRC5 for SRAM behave as a single MRC respectively. Hardware mirrors the two MRCs.

- Sixteen MRGDs for MRC2, MRC3 for first half of SRAM memory address space, and sixteen MRGDs for MRC4,MRC5 for the second half of SRAM are usable.
- Software can write to any of the two MRC instances and the write replicates to the other MRC automatically.
- Software can read from any of the two MRCs and get the same result.

15.1.4 XRDC_1 Introduction

This chip contains an additional XRDC manager, also known as XRDC_1, to support region protection for the off-CC NoC masters. The following table shows the ports that have XRDC_1 protection associated with them. Refer to the XRDC chapter for more details.

Table 51. XRDC_1 MDACn configurations

Submodule instance	Configuration	Number of MDA_W*x_y*_DFMT*z* registers	Bus master	Non-secure input[1]
XRDC_MDAC0	Nonprocessor	1	PCIe_1	input from master
XRDC_MDAC1	Nonprocessor	1	PFE_HIF0 (PFE_HIF BD fetch)	1
XRDC_MDAC2	Nonprocessor	1	PFE_HIF1 (PFE_HIF BD update)	1
XRDC_MDAC3	Nonprocessor	1	PFE_HIF2 (PFE_HIF data write)	1
>XRDC_MDAC4	Nonprocessor	1	PFE_HIF3 (PFE_HIF data read)	1
XRDC_MDAC5	Nonprocessor	1	PFE_DDR (PFE_PKT_MASTER)	1
XRDC_MDAC6	Nonprocessor	1	PFE_UTIL	1
XRDC_MDAC7	Nonprocessor	1	USB	1

1. The MDAC has a non-secure input for the bus master. Entries in this column mean:
   - 1: The bus master is always a non-secure input.
   - 0: The bus master is always a secure input.
   - Input from master: The bus master determines whether the input is 1 or 0.

Table 52. XRDC_1 MRCn configurations

Submodule instance	Region format	Number of descriptors	Slaves protected, module (port number)
XRDC_MRC0	Auto	4	STDBY_SRAM
XRDC_MRC1	Auto	4	LLCE
XRDC_MRC2	Auto	16	Concerto
XRDC_MRC3	Auto	4	PCIe_1
XRDC_MRC4	Auto	4	PFE
XRDC_MRC5	Auto	4	Service_Regs

Provided under NDA only
COMPANY PROPRIETARY
Beijing C&J Automotive Technology Co., Ltd.
c8c46e5-4821-4805-9e2-0d3ab62c1

Table 53. XRDC_1 PACn configurations

Module name	Slaves protected
XRDC_PAC0	Accelerator Peripheral group 0

15.2 Introduction

XRDC provides an integrated, scalable architectural framework for access control, system memory protection, and peripheral isolation. It allows software to assign chip resources including processor cores, non-core bus masters, memory regions, and slave peripherals for processing domains to support enforcement of robust operational environments. First, each bus mastering resource is assigned to a DID . For processors, there are additional fields that can optionally be used to assign each core to multiple domains. Next, the access control policies for the individual domains are programmed into slave memory region descriptors and slave peripheral domain access control registers. Finally, all accesses throughout the chip are monitored concurrently to determine the validity of each and every access. If a reference from a given domain has sufficient access rights, it is allowed to continue, else the access is aborted and error information captured.

The access control scheme defined by XRDC supports a 4-level model, combining the traditional privileged (also known as supervisor) and user modes with an additional signal defining the secure and nonsecure attributes of each memory reference. The result is a 4-level hierarchical access control mechanism, where:

SecurePriv(ileged) > SecureUser > NonsecurePriv(ileged) > NonsecureUser

with different access control policies that are based on read and write references. Combined with the user or privileged and secure or nonsecure attributes, a DID is associated with every system bus transaction and forms the hardware basis for implementation of the XRDC's access control mechanisms.

Access to shared memory regions and slave peripherals can be dynamically controlled with the optional inclusion of a hardware semaphore. This semaphore, if enabled for a given address space or peripheral, allows writes to the targeted address space only if the requesting domain owns the semaphore. This capability allows the access control policy for a given resource to be dynamically revised based on hardware semaphore ownership.

Each core contains its own local memory protection unit to control permissions and manage resources among the various tasks executing on that core. The XRDC adds a second layer of protection for the shared resources in a multicore system by providing an integrated, scalable architectural framework for access control, system memory protection, and peripheral isolation. It allows software to assign chip resources including processor cores, non-core bus masters, memory regions, and slave peripherals to processing domains to support enforcement of robust operational environments.

15.2.1 Features

The key features of XRDC include:

- Assignment of chip resources to processing domains
  - Resources are categorized into four groups:
    - Processor cores
    - Non-core bus masters
    - Slave memories
    - Slave peripherals
  - Each processing domain is assigned a unique DID
  - DID is a new attribute associated with every system bus transaction
  - Used in conjunction with user/privileged, secure/nonsecure attributes
- Access rights to slave targets defined in region descriptor registers for memories and access control registers for peripherals
- Support for sharing of memory and peripherals with optional inclusion of hardware semaphores to dynamically determine access rights

- Built upon a four-level hierarchical access control model
  - SecurePriv(ileged) > SecureUser > NonsecurePriv(ileged) > NonsecureUser
  - Encoded into a three-bit per-domain ACP used throughout the XRDC
  - Certain processors do not support the NonsecurePriv state. For these cores, the model simplifies to a three-state definition: SecurePriv > SecureUser > NonsecureUser
- Programming model and hardware implementation is distributed across multiple submodules
  - Support for a broad, highly-configurable architecture definition
  - Memory region descriptors support a format leveraged from earlier System Memory Protection Units (SMPUs)

15.2.2 Block diagram

As previously noted, the XRDC implementation is distributed across multiple submodules instantiated throughout the chip. The XRDC submodules include:

- MGR
  - The MGR submodule provides hardware management for all programming model reads and writes
- MDAC
  - The MDAC handles resource assignments and generation of the domain identifiers
- MRC
  - The MRC implements the access controls for slave memories based on the pre-programmed region descriptor registers
- PAC
  - The PAC implements the access controls for slave peripherals based on the pre-programmed domain access control registers

For chip-specific implementation details of this module's instances, see the chip configuration information.

See Figure 34 for a simplified XRDC block diagram focusing on topology and connections.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-290d3b624c

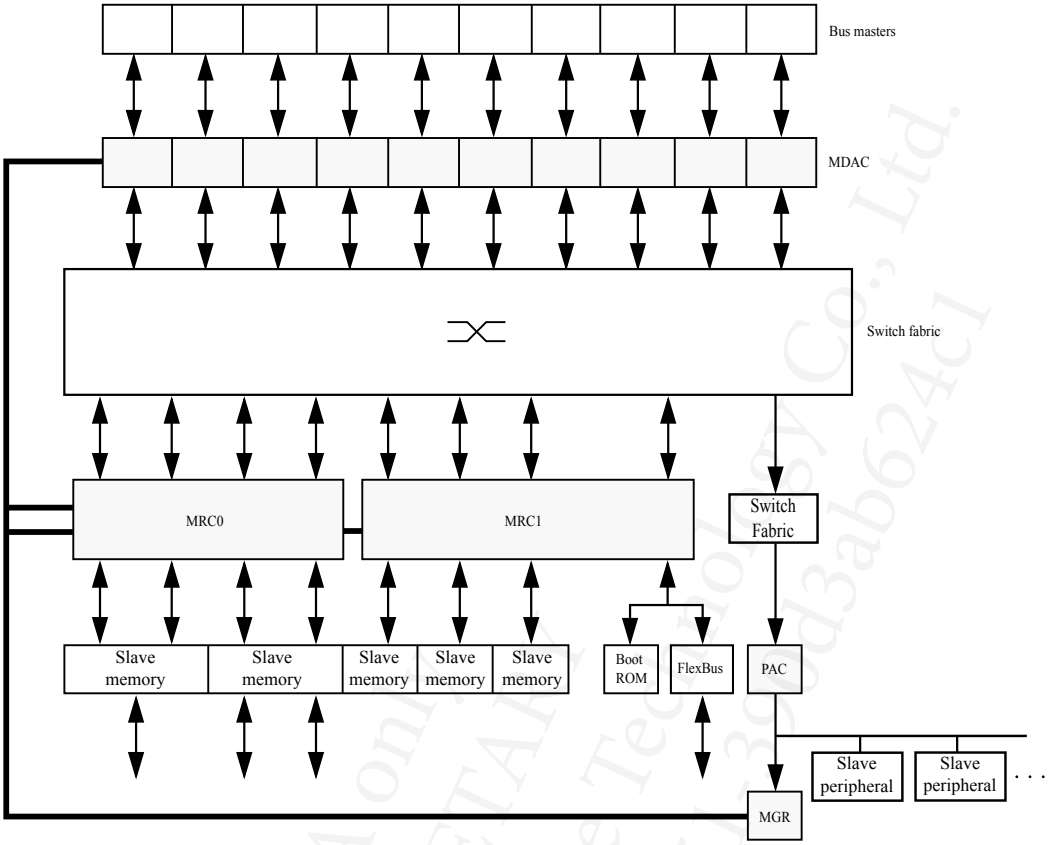


Figure 34. Simplified XRDC block diagram

15.2.3 Modes of operation

The XRDC module does not support any special modes of operation. As a memory-mapped device located in the core platform's clock domain, it strictly responds on the basis of the memory addresses of the connected system buses. The slave peripheral bus is used to access XRDC's distributed programming model; these references are only allowed if the requesting bus master is operating in the secure, privileged mode. Domain assignment and access control functions are evaluated on a reference-by-reference basis using the addresses and attributes connected to the system bus port(s). The implementation of the MRC submodule supports either AMBA-AHB or AMBA-AXI system bus protocols.

15.3 External signal description

The XRDC module does not directly support any external interfaces.

The internal interfaces include a standard 32-bit slave bus for all programming model accesses, connections to the address phase signals associated with AHB and/or AXI system buses, and connections to the slave peripheral buses as shown in Figure 34.

15.4 Memory map/register definition

The XRDC module supports a large number of architecturally defined, program-visible registers distributed across four 4 KB address slots. The actual registers are chip-specific.

The register resources of the XRDC programming model can only be accessed while in secure, privileged mode. Unless noted otherwise, the programming model registers can be accessed via 8-, 16- or 32-bit reads, and 32-bit write references. Attempted accesses in a different operating mode, using unsupported write data sizes, and writes to read-only resources or to reserved spaces are terminated with an error unless noted otherwise.

Accesses to these memory map holes return an error:

- Any access to an MDAC, PDAC, or MRC that does not exist
- Holes in the 0h-F0h and DERR to PID register space
- For MDAC, gaps in the registers
- For MRCs, RGD gaps; thus, if there are four RGDs, an attempted access to RGD5 fails

Accesses to these memory map holes do not return a bus error:

- For MRCs, current RGDs are four words but have eight words of address available; words 4-7
- For PDAC, reserved slots

The XRDC programming model is partitioned into five groups of registers:

- Basic hardware control and configuration
- Domain errors: location and details
- Master domain assignments
- Peripheral domain access controls
- Memory region descriptors

It should be noted that many of the programming model registers in the XRDC are organized as two-dimensional data structures. These generic arrays contain *m* words representing the columns and *n* instances of the structure representing the rows, and may be described as `structure[n][m]`. They appear in the address space of the programming model memory map in the standard C language row-major layout; that is, the *m* words representing the column appear sequentially with the entire row replicated *n* times. The XRDC register structure uses the naming convention of *<regname>_Wm_n*, where the column number *m* appears as a numeric suffix on the W (32-bit word) identifier, and the row identifier *n* appears as the final numerical suffix.

The architectural definition of the complete XRDC programming model is presented in this section. Each implementation is SoC-specific and highly dependent on the chip's requirements.

15.4.1 XRDC_0 register descriptions

15.4.1.1 XRDC_0 memory map

XRDC_0 base address: 401A_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	0000_008Ah
F0h	Hardware Configuration 0 (HWCFG0)	32	RO	130C_1307h
F4h	Hardware Configuration 1 (HWCFG1)	32	RO	See description
F8h	Hardware Configuration 2 (HWCFG2)	32	RO	0000_0000h
100h	Master Domain Assignment Configuration (MDACFG0)	8	RO	08h
101h	Master Domain Assignment Configuration (MDACFG1)	8	RO	08h
102h	Master Domain Assignment Configuration (MDACFG2)	8	RO	81h
103h	Master Domain Assignment Configuration (MDACFG3)	8	RO	81h
104h	Master Domain Assignment Configuration (MDACFG4)	8	RO	81h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
105h	Master Domain Assignment Configuration (MDACFG5)	8	RO	81h
106h	Master Domain Assignment Configuration (MDACFG6)	8	RO	81h
107h	Master Domain Assignment Configuration (MDACFG7)	8	RO	81h
108h	Master Domain Assignment Configuration (MDACFG8)	8	RO	08h
109h	Master Domain Assignment Configuration (MDACFG9)	8	RO	08h
10Ah	Master Domain Assignment Configuration (MDACFG10)	8	RO	08h
10Bh	Master Domain Assignment Configuration (MDACFG11)	8	RO	81h
10Ch	Master Domain Assignment Configuration (MDACFG12)	8	RO	81h
10Fh	Master Domain Assignment Configuration (MDACFG15)	8	RO	81h
110h	Master Domain Assignment Configuration (MDACFG16)	8	RO	08h
111h	Master Domain Assignment Configuration (MDACFG17)	8	RO	08h
112h	Master Domain Assignment Configuration (MDACFG18)	8	RO	08h
113h	Master Domain Assignment Configuration (MDACFG19)	8	RO	81h
114h	Master Domain Assignment Configuration (MDACFG20)	8	RO	81h
115h	Master Domain Assignment Configuration (MDACFG21)	8	RO	81h
140h	Memory Region Configuration (MRCFG0)	8	RO	10h
142h	Memory Region Configuration (MRCFG2)	8	RO	10h
143h	Memory Region Configuration (MRCFG3)	8	RO	10h
144h	Memory Region Configuration (MRCFG4)	8	RO	10h
145h	Memory Region Configuration (MRCFG5)	8	RO	10h
146h	Memory Region Configuration (MRCFG6)	8	RO	10h
147h	Memory Region Configuration (MRCFG7)	8	RO	10h
148h	Memory Region Configuration (MRCFG8)	8	RO	10h
149h	Memory Region Configuration (MRCFG9)	8	RO	10h
14Ah	Memory Region Configuration (MRCFG10)	8	RO	10h
14Bh	Memory Region Configuration (MRCFG11)	8	RO	0Ch
14Ch	Memory Region Configuration (MRCFG12)	8	RO	04h
14Dh	Memory Region Configuration (MRCFG13)	8	RO	10h
200h - 21Ch	Domain Error Location (DERRLOC0 - DERRLOC7)	32	RO	0000_0000h
400h	Domain Error Word0 (DERR_W0_0)	32	RO	0000_0000h
404h	Domain Error Word1 (DERR_W1_0)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
408h	Domain Error Word2 (DERR_W2_0)	32	RO	0000_0000h
40Ch	Domain Error Word3 (DERR_W3_0)	32	WORZ	0000_0000h
420h	Domain Error Word0 (DERR_W0_2)	32	RO	0000_0000h
424h	Domain Error Word1 (DERR_W1_2)	32	RO	0000_0000h
428h	Domain Error Word2 (DERR_W2_2)	32	RO	0000_0000h
42Ch	Domain Error Word3 (DERR_W3_2)	32	WORZ	0000_0000h
430h	Domain Error Word0 (DERR_W0_3)	32	RO	0000_0000h
434h	Domain Error Word1 (DERR_W1_3)	32	RO	0000_0000h
438h	Domain Error Word2 (DERR_W2_3)	32	RO	0000_0000h
43Ch	Domain Error Word3 (DERR_W3_3)	32	WORZ	0000_0000h
440h	Domain Error Word0 (DERR_W0_4)	32	RO	0000_0000h
444h	Domain Error Word1 (DERR_W1_4)	32	RO	0000_0000h
448h	Domain Error Word2 (DERR_W2_4)	32	RO	0000_0000h
44Ch	Domain Error Word3 (DERR_W3_4)	32	WORZ	0000_0000h
450h	Domain Error Word0 (DERR_W0_5)	32	RO	0000_0000h
454h	Domain Error Word1 (DERR_W1_5)	32	RO	0000_0000h
458h	Domain Error Word2 (DERR_W2_5)	32	RO	0000_0000h
45Ch	Domain Error Word3 (DERR_W3_5)	32	WORZ	0000_0000h
460h	Domain Error Word0 (DERR_W0_6)	32	RO	0000_0000h
464h	Domain Error Word1 (DERR_W1_6)	32	RO	0000_0000h
468h	Domain Error Word2 (DERR_W2_6)	32	RO	0000_0000h
46Ch	Domain Error Word3 (DERR_W3_6)	32	WORZ	0000_0000h
470h	Domain Error Word0 (DERR_W0_7)	32	RO	0000_0000h
474h	Domain Error Word1 (DERR_W1_7)	32	RO	0000_0000h
478h	Domain Error Word2 (DERR_W2_7)	32	RO	0000_0000h
47Ch	Domain Error Word3 (DERR_W3_7)	32	WORZ	0000_0000h
480h	Domain Error Word0 (DERR_W0_8)	32	RO	0000_0000h
484h	Domain Error Word1 (DERR_W1_8)	32	RO	0000_0000h
488h	Domain Error Word2 (DERR_W2_8)	32	RO	0000_0000h
48Ch	Domain Error Word3 (DERR_W3_8)	32	WORZ	0000_0000h
490h	Domain Error Word0 (DERR_W0_9)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
494h	Domain Error Word1 (DERR_W1_9)	32	RO	0000_0000h
498h	Domain Error Word2 (DERR_W2_9)	32	RO	0000_0000h
49Ch	Domain Error Word3 (DERR_W3_9)	32	WORZ	0000_0000h
4A0h	Domain Error Word0 (DERR_W0_10)	32	RO	0000_0000h
4A4h	Domain Error Word1 (DERR_W1_10)	32	RO	0000_0000h
4A8h	Domain Error Word2 (DERR_W2_10)	32	RO	0000_0000h
4ACh	Domain Error Word3 (DERR_W3_10)	32	WORZ	0000_0000h
4B0h	Domain Error Word0 (DERR_W0_11)	32	RO	0000_0000h
4B4h	Domain Error Word1 (DERR_W1_11)	32	RO	0000_0000h
4B8h	Domain Error Word2 (DERR_W2_11)	32	RO	0000_0000h
4BCh	Domain Error Word3 (DERR_W3_11)	32	WORZ	0000_0000h
4C0h	Domain Error Word0 (DERR_W0_12)	32	RO	0000_0000h
4C4h	Domain Error Word1 (DERR_W1_12)	32	RO	0000_0000h
4C8h	Domain Error Word2 (DERR_W2_12)	32	RO	0000_0000h
4CCh	Domain Error Word3 (DERR_W3_12)	32	WORZ	0000_0000h
4D0h	Domain Error Word0 (DERR_W0_13)	32	RO	0000_0000h
4D4h	Domain Error Word1 (DERR_W1_13)	32	RO	0000_0000h
4D8h	Domain Error Word2 (DERR_W2_13)	32	RO	0000_0000h
4DCh	Domain Error Word3 (DERR_W3_13)	32	WORZ	0000_0000h
500h	Domain Error Word0 (DERR_W0_16)	32	RO	0000_0000h
504h	Domain Error Word1 (DERR_W1_16)	32	RO	0000_0000h
508h	Domain Error Word2 (DERR_W2_16)	32	RO	0000_0000h
50Ch	Domain Error Word3 (DERR_W3_16)	32	WORZ	0000_0000h
510h	Domain Error Word0 (DERR_W0_17)	32	RO	0000_0000h
514h	Domain Error Word1 (DERR_W1_17)	32	RO	0000_0000h
518h	Domain Error Word2 (DERR_W2_17)	32	RO	0000_0000h
51Ch	Domain Error Word3 (DERR_W3_17)	32	WORZ	0000_0000h
520h	Domain Error Word0 (DERR_W0_18)	32	RO	0000_0000h
524h	Domain Error Word1 (DERR_W1_18)	32	RO	0000_0000h
528h	Domain Error Word2 (DERR_W2_18)	32	RO	0000_0000h
52Ch	Domain Error Word3 (DERR_W3_18)	32	WORZ	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
530h	Domain Error Word0 (DERR_W0_19)	32	RO	0000_0000h
534h	Domain Error Word1 (DERR_W1_19)	32	RO	0000_0000h
538h	Domain Error Word2 (DERR_W2_19)	32	RO	0000_0000h
53Ch	Domain Error Word3 (DERR_W3_19)	32	WORZ	0000_0000h
700h	Process Identifier (PID0)	32	RW	0100_0000h
704h	Process Identifier (PID1)	32	RW	0100_0000h
720h	Process Identifier (PID8)	32	RW	0000_0000h
724h	Process Identifier (PID9)	32	RW	0000_0000h
728h	Process Identifier (PID10)	32	RW	0000_0000h
740h	Process Identifier (PID16)	32	RW	0000_0000h
744h	Process Identifier (PID17)	32	RW	0000_0000h
748h	Process Identifier (PID18)	32	RW	0000_0000h
800h	Master Domain Assignment (MDA_W0_0_DFMT0)	32	RW	0000_0000h
804h	Master Domain Assignment (MDA_W1_0_DFMT0)	32	RW	0000_0000h
808h	Master Domain Assignment (MDA_W2_0_DFMT0)	32	RW	0000_0000h
80Ch	Master Domain Assignment (MDA_W3_0_DFMT0)	32	RW	0000_0000h
810h	Master Domain Assignment (MDA_W4_0_DFMT0)	32	RW	0000_0000h
814h	Master Domain Assignment (MDA_W5_0_DFMT0)	32	RW	0000_0000h
818h	Master Domain Assignment (MDA_W6_0_DFMT0)	32	RW	0000_0000h
81Ch	Master Domain Assignment (MDA_W7_0_DFMT0)	32	RW	0000_0000h
820h	Master Domain Assignment (MDA_W0_1_DFMT0)	32	RW	0000_0000h
824h	Master Domain Assignment (MDA_W1_1_DFMT0)	32	RW	0000_0000h
828h	Master Domain Assignment (MDA_W2_1_DFMT0)	32	RW	0000_0000h
82Ch	Master Domain Assignment (MDA_W3_1_DFMT0)	32	RW	0000_0000h
830h	Master Domain Assignment (MDA_W4_1_DFMT0)	32	RW	0000_0000h
834h	Master Domain Assignment (MDA_W5_1_DFMT0)	32	RW	0000_0000h
838h	Master Domain Assignment (MDA_W6_1_DFMT0)	32	RW	0000_0000h
83Ch	Master Domain Assignment (MDA_W7_1_DFMT0)	32	RW	0000_0000h
840h	Master Domain Assignment (MDA_W0_2_DFMT1)	32	RW	2000_0000h
860h	Master Domain Assignment (MDA_W0_3_DFMT1)	32	RW	2000_0000h
880h	Master Domain Assignment (MDA_W0_4_DFMT1)	32	RW	2000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8A0h	Master Domain Assignment (MDA_W0_5_DFMT1)	32	RW	2000_0000h
8C0h	Master Domain Assignment (MDA_W0_6_DFMT1)	32	RW	2000_0000h
8E0h	Master Domain Assignment (MDA_W0_7_DFMT1)	32	RW	2000_0000h
900h	Master Domain Assignment (MDA_W0_8_DFMT0)	32	RW	0000_0000h
904h	Master Domain Assignment (MDA_W1_8_DFMT0)	32	RW	0000_0000h
908h	Master Domain Assignment (MDA_W2_8_DFMT0)	32	RW	0000_0000h
90Ch	Master Domain Assignment (MDA_W3_8_DFMT0)	32	RW	0000_0000h
910h	Master Domain Assignment (MDA_W4_8_DFMT0)	32	RW	0000_0000h
914h	Master Domain Assignment (MDA_W5_8_DFMT0)	32	RW	0000_0000h
918h	Master Domain Assignment (MDA_W6_8_DFMT0)	32	RW	0000_0000h
91Ch	Master Domain Assignment (MDA_W7_8_DFMT0)	32	RW	0000_0000h
920h	Master Domain Assignment (MDA_W0_9_DFMT0)	32	RW	0000_0000h
924h	Master Domain Assignment (MDA_W1_9_DFMT0)	32	RW	0000_0000h
928h	Master Domain Assignment (MDA_W2_9_DFMT0)	32	RW	0000_0000h
92Ch	Master Domain Assignment (MDA_W3_9_DFMT0)	32	RW	0000_0000h
930h	Master Domain Assignment (MDA_W4_9_DFMT0)	32	RW	0000_0000h
934h	Master Domain Assignment (MDA_W5_9_DFMT0)	32	RW	0000_0000h
938h	Master Domain Assignment (MDA_W6_9_DFMT0)	32	RW	0000_0000h
93Ch	Master Domain Assignment (MDA_W7_9_DFMT0)	32	RW	0000_0000h
940h	Master Domain Assignment (MDA_W0_10_DFMT0)	32	RW	0000_0000h
944h	Master Domain Assignment (MDA_W1_10_DFMT0)	32	RW	0000_0000h
948h	Master Domain Assignment (MDA_W2_10_DFMT0)	32	RW	0000_0000h
94Ch	Master Domain Assignment (MDA_W3_10_DFMT0)	32	RW	0000_0000h
950h	Master Domain Assignment (MDA_W4_10_DFMT0)	32	RW	0000_0000h
954h	Master Domain Assignment (MDA_W5_10_DFMT0)	32	RW	0000_0000h
958h	Master Domain Assignment (MDA_W6_10_DFMT0)	32	RW	0000_0000h
95Ch	Master Domain Assignment (MDA_W7_10_DFMT0)	32	RW	0000_0000h
960h	Master Domain Assignment (MDA_W0_11_DFMT1)	32	RW	2000_0000h
980h	Master Domain Assignment (MDA_W0_12_DFMT1)	32	RW	2000_0000h
9E0h	Master Domain Assignment (MDA_W0_15_DFMT1)	32	RW	2000_0000h
A00h	Master Domain Assignment (MDA_W0_16_DFMT0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A04h	Master Domain Assignment (MDA_W1_16_DFMT0)	32	RW	0000_0000h
A08h	Master Domain Assignment (MDA_W2_16_DFMT0)	32	RW	0000_0000h
A0Ch	Master Domain Assignment (MDA_W3_16_DFMT0)	32	RW	0000_0000h
A10h	Master Domain Assignment (MDA_W4_16_DFMT0)	32	RW	0000_0000h
A14h	Master Domain Assignment (MDA_W5_16_DFMT0)	32	RW	0000_0000h
A18h	Master Domain Assignment (MDA_W6_16_DFMT0)	32	RW	0000_0000h
A1Ch	Master Domain Assignment (MDA_W7_16_DFMT0)	32	RW	0000_0000h
A20h	Master Domain Assignment (MDA_W0_17_DFMT0)	32	RW	0000_0000h
A24h	Master Domain Assignment (MDA_W1_17_DFMT0)	32	RW	0000_0000h
A28h	Master Domain Assignment (MDA_W2_17_DFMT0)	32	RW	0000_0000h
A2Ch	Master Domain Assignment (MDA_W3_17_DFMT0)	32	RW	0000_0000h
A30h	Master Domain Assignment (MDA_W4_17_DFMT0)	32	RW	0000_0000h
A34h	Master Domain Assignment (MDA_W5_17_DFMT0)	32	RW	0000_0000h
A38h	Master Domain Assignment (MDA_W6_17_DFMT0)	32	RW	0000_0000h
A3Ch	Master Domain Assignment (MDA_W7_17_DFMT0)	32	RW	0000_0000h
A40h	Master Domain Assignment (MDA_W0_18_DFMT0)	32	RW	0000_0000h
A44h	Master Domain Assignment (MDA_W1_18_DFMT0)	32	RW	0000_0000h
A48h	Master Domain Assignment (MDA_W2_18_DFMT0)	32	RW	0000_0000h
A4Ch	Master Domain Assignment (MDA_W3_18_DFMT0)	32	RW	0000_0000h
A50h	Master Domain Assignment (MDA_W4_18_DFMT0)	32	RW	0000_0000h
A54h	Master Domain Assignment (MDA_W5_18_DFMT0)	32	RW	0000_0000h
A58h	Master Domain Assignment (MDA_W6_18_DFMT0)	32	RW	0000_0000h
A5Ch	Master Domain Assignment (MDA_W7_18_DFMT0)	32	RW	0000_0000h
A60h	Master Domain Assignment (MDA_W0_19_DFMT1)	32	RW	2000_0000h
A80h	Master Domain Assignment (MDA_W0_20_DFMT1)	32	RW	2000_0000h
AA0h	Master Domain Assignment (MDA_W0_21_DFMT1)	32	RW	2000_0000h
1000h	Peripheral Domain Access Control (PDAC_W0_0)	32	RW	0000_0000h
1004h	Peripheral Domain Access Control (PDAC_W1_0)	32	RW	0000_0000h
1008h	Peripheral Domain Access Control (PDAC_W0_1)	32	RW	0000_0000h
100Ch	Peripheral Domain Access Control (PDAC_W1_1)	32	RW	0000_0000h
1010h	Peripheral Domain Access Control (PDAC_W0_2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1014h	Peripheral Domain Access Control (PDAC_W1_2)	32	RW	0000_0000h
1018h	Peripheral Domain Access Control (PDAC_W0_3)	32	RW	0000_0000h
101Ch	Peripheral Domain Access Control (PDAC_W1_3)	32	RW	0000_0000h
1020h	Peripheral Domain Access Control (PDAC_W0_4)	32	RW	0000_0000h
1024h	Peripheral Domain Access Control (PDAC_W1_4)	32	RW	0000_0000h
1028h	Peripheral Domain Access Control (PDAC_W0_5)	32	RW	0000_0000h
102Ch	Peripheral Domain Access Control (PDAC_W1_5)	32	RW	0000_0000h
1030h	Peripheral Domain Access Control (PDAC_W0_6)	32	RW	0000_0000h
1034h	Peripheral Domain Access Control (PDAC_W1_6)	32	RW	0000_0000h
1038h	Peripheral Domain Access Control (PDAC_W0_7)	32	RW	0000_0000h
103Ch	Peripheral Domain Access Control (PDAC_W1_7)	32	RW	0000_0000h
1040h	Peripheral Domain Access Control (PDAC_W0_8)	32	RW	0000_0000h
1044h	Peripheral Domain Access Control (PDAC_W1_8)	32	RW	0000_0000h
1048h	Peripheral Domain Access Control (PDAC_W0_9)	32	RW	0000_0000h
104Ch	Peripheral Domain Access Control (PDAC_W1_9)	32	RW	0000_0000h
1050h	Peripheral Domain Access Control (PDAC_W0_10)	32	RW	0000_0000h
1054h	Peripheral Domain Access Control (PDAC_W1_10)	32	RW	0000_0000h
1058h	Peripheral Domain Access Control (PDAC_W0_11)	32	RW	0000_0000h
105Ch	Peripheral Domain Access Control (PDAC_W1_11)	32	RW	0000_0000h
1060h	Peripheral Domain Access Control (PDAC_W0_12)	32	RW	0000_0000h
1064h	Peripheral Domain Access Control (PDAC_W1_12)	32	RW	0000_0000h
1068h	Peripheral Domain Access Control (PDAC_W0_13)	32	RW	0000_0000h
106Ch	Peripheral Domain Access Control (PDAC_W1_13)	32	RW	0000_0000h
1070h	Peripheral Domain Access Control (PDAC_W0_14)	32	RW	0000_0000h
1074h	Peripheral Domain Access Control (PDAC_W1_14)	32	RW	0000_0000h
1078h	Peripheral Domain Access Control (PDAC_W0_15)	32	RW	0000_0000h
107Ch	Peripheral Domain Access Control (PDAC_W1_15)	32	RW	0000_0000h
1080h	Peripheral Domain Access Control (PDAC_W0_16)	32	RW	0000_0000h
1084h	Peripheral Domain Access Control (PDAC_W1_16)	32	RW	0000_0000h
1088h	Peripheral Domain Access Control (PDAC_W0_17)	32	RW	0000_0000h
108Ch	Peripheral Domain Access Control (PDAC_W1_17)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1090h	Peripheral Domain Access Control (PDAC_W0_18)	32	RW	0000_0000h
1094h	Peripheral Domain Access Control (PDAC_W1_18)	32	RW	0000_0000h
1098h	Peripheral Domain Access Control (PDAC_W0_19)	32	RW	0000_0000h
109Ch	Peripheral Domain Access Control (PDAC_W1_19)	32	RW	0000_0000h
10A0h	Peripheral Domain Access Control (PDAC_W0_20)	32	RW	0000_0000h
10A4h	Peripheral Domain Access Control (PDAC_W1_20)	32	RW	0000_0000h
10A8h	Peripheral Domain Access Control (PDAC_W0_21)	32	RW	0000_0000h
10ACh	Peripheral Domain Access Control (PDAC_W1_21)	32	RW	0000_0000h
10B0h	Peripheral Domain Access Control (PDAC_W0_22)	32	RW	0000_0000h
10B4h	Peripheral Domain Access Control (PDAC_W1_22)	32	RW	0000_0000h
10B8h	Peripheral Domain Access Control (PDAC_W0_23)	32	RW	0000_0000h
10BCh	Peripheral Domain Access Control (PDAC_W1_23)	32	RW	0000_0000h
10C0h	Peripheral Domain Access Control (PDAC_W0_24)	32	RW	0000_0000h
10C4h	Peripheral Domain Access Control (PDAC_W1_24)	32	RW	0000_0000h
10C8h	Peripheral Domain Access Control (PDAC_W0_25)	32	RW	0000_0000h
10CCh	Peripheral Domain Access Control (PDAC_W1_25)	32	RW	0000_0000h
10D0h	Peripheral Domain Access Control (PDAC_W0_26)	32	RW	0000_0000h
10D4h	Peripheral Domain Access Control (PDAC_W1_26)	32	RW	0000_0000h
10D8h	Peripheral Domain Access Control (PDAC_W0_27)	32	RW	0000_0000h
10DCh	Peripheral Domain Access Control (PDAC_W1_27)	32	RW	0000_0000h
10E0h	Peripheral Domain Access Control (PDAC_W0_28)	32	RW	0000_0000h
10E4h	Peripheral Domain Access Control (PDAC_W1_28)	32	RW	0000_0000h
10E8h	Peripheral Domain Access Control (PDAC_W0_29)	32	RW	0000_0000h
10ECh	Peripheral Domain Access Control (PDAC_W1_29)	32	RW	0000_0000h
10F0h	Peripheral Domain Access Control (PDAC_W0_30)	32	RW	0000_0000h
10F4h	Peripheral Domain Access Control (PDAC_W1_30)	32	RW	0000_0000h
10F8h	Peripheral Domain Access Control (PDAC_W0_31)	32	RW	0000_0000h
10FCh	Peripheral Domain Access Control (PDAC_W1_31)	32	RW	0000_0000h
1400h	Peripheral Domain Access Control (PDAC_W0_128)	32	RW	0000_0000h
1404h	Peripheral Domain Access Control (PDAC_W1_128)	32	RW	0000_0000h
1408h	Peripheral Domain Access Control (PDAC_W0_129)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
140Ch	Peripheral Domain Access Control (PDAC_W1_129)	32	RW	0000_0000h
1410h	Peripheral Domain Access Control (PDAC_W0_130)	32	RW	0000_0000h
1414h	Peripheral Domain Access Control (PDAC_W1_130)	32	RW	0000_0000h
1418h	Peripheral Domain Access Control (PDAC_W0_131)	32	RW	0000_0000h
141Ch	Peripheral Domain Access Control (PDAC_W1_131)	32	RW	0000_0000h
1420h	Peripheral Domain Access Control (PDAC_W0_132)	32	RW	0000_0000h
1424h	Peripheral Domain Access Control (PDAC_W1_132)	32	RW	0000_0000h
1428h	Peripheral Domain Access Control (PDAC_W0_133)	32	RW	0000_0000h
142Ch	Peripheral Domain Access Control (PDAC_W1_133)	32	RW	0000_0000h
1430h	Peripheral Domain Access Control (PDAC_W0_134)	32	RW	0000_0000h
1434h	Peripheral Domain Access Control (PDAC_W1_134)	32	RW	0000_0000h
1438h	Peripheral Domain Access Control (PDAC_W0_135)	32	RW	0000_0000h
143Ch	Peripheral Domain Access Control (PDAC_W1_135)	32	RW	0000_0000h
1440h	Peripheral Domain Access Control (PDAC_W0_136)	32	RW	0000_0000h
1444h	Peripheral Domain Access Control (PDAC_W1_136)	32	RW	0000_0000h
1448h	Peripheral Domain Access Control (PDAC_W0_137)	32	RW	0000_0000h
144Ch	Peripheral Domain Access Control (PDAC_W1_137)	32	RW	0000_0000h
1450h	Peripheral Domain Access Control (PDAC_W0_138)	32	RW	0000_0000h
1454h	Peripheral Domain Access Control (PDAC_W1_138)	32	RW	0000_0000h
1458h	Peripheral Domain Access Control (PDAC_W0_139)	32	RW	0000_0000h
145Ch	Peripheral Domain Access Control (PDAC_W1_139)	32	RW	0000_0000h
1460h	Peripheral Domain Access Control (PDAC_W0_140)	32	RW	0000_0000h
1464h	Peripheral Domain Access Control (PDAC_W1_140)	32	RW	0000_0000h
1468h	Peripheral Domain Access Control (PDAC_W0_141)	32	RW	0000_0000h
146Ch	Peripheral Domain Access Control (PDAC_W1_141)	32	RW	0000_0000h
1470h	Peripheral Domain Access Control (PDAC_W0_142)	32	RW	0000_0000h
1474h	Peripheral Domain Access Control (PDAC_W1_142)	32	RW	0000_0000h
1478h	Peripheral Domain Access Control (PDAC_W0_143)	32	RW	0000_0000h
147Ch	Peripheral Domain Access Control (PDAC_W1_143)	32	RW	0000_0000h
1480h	Peripheral Domain Access Control (PDAC_W0_144)	32	RW	0000_0000h
1484h	Peripheral Domain Access Control (PDAC_W1_144)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1488h	Peripheral Domain Access Control (PDAC_W0_145)	32	RW	0000_0000h
148Ch	Peripheral Domain Access Control (PDAC_W1_145)	32	RW	0000_0000h
1490h	Peripheral Domain Access Control (PDAC_W0_146)	32	RW	0000_0000h
1494h	Peripheral Domain Access Control (PDAC_W1_146)	32	RW	0000_0000h
1498h	Peripheral Domain Access Control (PDAC_W0_147)	32	RW	0000_0000h
149Ch	Peripheral Domain Access Control (PDAC_W1_147)	32	RW	0000_0000h
14A0h	Peripheral Domain Access Control (PDAC_W0_148)	32	RW	0000_0000h
14A4h	Peripheral Domain Access Control (PDAC_W1_148)	32	RW	0000_0000h
14A8h	Peripheral Domain Access Control (PDAC_W0_149)	32	RW	0000_0000h
14ACh	Peripheral Domain Access Control (PDAC_W1_149)	32	RW	0000_0000h
14B0h	Peripheral Domain Access Control (PDAC_W0_150)	32	RW	0000_0000h
14B4h	Peripheral Domain Access Control (PDAC_W1_150)	32	RW	0000_0000h
14B8h	Peripheral Domain Access Control (PDAC_W0_151)	32	RW	0000_0000h
14BCh	Peripheral Domain Access Control (PDAC_W1_151)	32	RW	0000_0000h
14C0h	Peripheral Domain Access Control (PDAC_W0_152)	32	RW	0000_0000h
14C4h	Peripheral Domain Access Control (PDAC_W1_152)	32	RW	0000_0000h
14C8h	Peripheral Domain Access Control (PDAC_W0_153)	32	RW	0000_0000h
14CCh	Peripheral Domain Access Control (PDAC_W1_153)	32	RW	0000_0000h
14D0h	Peripheral Domain Access Control (PDAC_W0_154)	32	RW	0000_0000h
14D4h	Peripheral Domain Access Control (PDAC_W1_154)	32	RW	0000_0000h
14D8h	Peripheral Domain Access Control (PDAC_W0_155)	32	RW	0000_0000h
14DCh	Peripheral Domain Access Control (PDAC_W1_155)	32	RW	0000_0000h
14E0h	Peripheral Domain Access Control (PDAC_W0_156)	32	RW	0000_0000h
14E4h	Peripheral Domain Access Control (PDAC_W1_156)	32	RW	0000_0000h
14E8h	Peripheral Domain Access Control (PDAC_W0_157)	32	RW	0000_0000h
14ECh	Peripheral Domain Access Control (PDAC_W1_157)	32	RW	0000_0000h
14F0h	Peripheral Domain Access Control (PDAC_W0_158)	32	RW	0000_0000h
14F4h	Peripheral Domain Access Control (PDAC_W1_158)	32	RW	0000_0000h
14F8h	Peripheral Domain Access Control (PDAC_W0_159)	32	RW	0000_0000h
14FCh	Peripheral Domain Access Control (PDAC_W1_159)	32	RW	0000_0000h
1800h	Peripheral Domain Access Control (PDAC_W0_256)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1804h	Peripheral Domain Access Control (PDAC_W1_256)	32	RW	0000_0000h
1808h	Peripheral Domain Access Control (PDAC_W0_257)	32	RW	0000_0000h
180Ch	Peripheral Domain Access Control (PDAC_W1_257)	32	RW	0000_0000h
1810h	Peripheral Domain Access Control (PDAC_W0_258)	32	RW	0000_0000h
1814h	Peripheral Domain Access Control (PDAC_W1_258)	32	RW	0000_0000h
1818h	Peripheral Domain Access Control (PDAC_W0_259)	32	RW	0000_0000h
181Ch	Peripheral Domain Access Control (PDAC_W1_259)	32	RW	0000_0000h
1820h	Peripheral Domain Access Control (PDAC_W0_260)	32	RW	0000_0000h
1824h	Peripheral Domain Access Control (PDAC_W1_260)	32	RW	0000_0000h
1828h	Peripheral Domain Access Control (PDAC_W0_261)	32	RW	0000_0000h
182Ch	Peripheral Domain Access Control (PDAC_W1_261)	32	RW	0000_0000h
1830h	Peripheral Domain Access Control (PDAC_W0_262)	32	RW	0000_0000h
1834h	Peripheral Domain Access Control (PDAC_W1_262)	32	RW	0000_0000h
1838h	Peripheral Domain Access Control (PDAC_W0_263)	32	RW	0000_0000h
183Ch	Peripheral Domain Access Control (PDAC_W1_263)	32	RW	0000_0000h
1840h	Peripheral Domain Access Control (PDAC_W0_264)	32	RW	0000_0000h
1844h	Peripheral Domain Access Control (PDAC_W1_264)	32	RW	0000_0000h
1848h	Peripheral Domain Access Control (PDAC_W0_265)	32	RW	0000_0000h
184Ch	Peripheral Domain Access Control (PDAC_W1_265)	32	RW	0000_0000h
1850h	Peripheral Domain Access Control (PDAC_W0_266)	32	RW	0000_0000h
1854h	Peripheral Domain Access Control (PDAC_W1_266)	32	RW	0000_0000h
1858h	Peripheral Domain Access Control (PDAC_W0_267)	32	RW	0000_0000h
185Ch	Peripheral Domain Access Control (PDAC_W1_267)	32	RW	0000_0000h
1860h	Peripheral Domain Access Control (PDAC_W0_268)	32	RW	0000_0000h
1864h	Peripheral Domain Access Control (PDAC_W1_268)	32	RW	0000_0000h
1868h	Peripheral Domain Access Control (PDAC_W0_269)	32	RW	0000_0000h
186Ch	Peripheral Domain Access Control (PDAC_W1_269)	32	RW	0000_0000h
1870h	Peripheral Domain Access Control (PDAC_W0_270)	32	RW	0000_0000h
1874h	Peripheral Domain Access Control (PDAC_W1_270)	32	RW	0000_0000h
1878h	Peripheral Domain Access Control (PDAC_W0_271)	32	RW	0000_0000h
187Ch	Peripheral Domain Access Control (PDAC_W1_271)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1880h	Peripheral Domain Access Control (PDAC_W0_272)	32	RW	0000_0000h
1884h	Peripheral Domain Access Control (PDAC_W1_272)	32	RW	0000_0000h
1888h	Peripheral Domain Access Control (PDAC_W0_273)	32	RW	0000_0000h
188Ch	Peripheral Domain Access Control (PDAC_W1_273)	32	RW	0000_0000h
1890h	Peripheral Domain Access Control (PDAC_W0_274)	32	RW	0000_0000h
1894h	Peripheral Domain Access Control (PDAC_W1_274)	32	RW	0000_0000h
1898h	Peripheral Domain Access Control (PDAC_W0_275)	32	RW	0000_0000h
189Ch	Peripheral Domain Access Control (PDAC_W1_275)	32	RW	0000_0000h
18A0h	Peripheral Domain Access Control (PDAC_W0_276)	32	RW	0000_0000h
18A4h	Peripheral Domain Access Control (PDAC_W1_276)	32	RW	0000_0000h
18A8h	Peripheral Domain Access Control (PDAC_W0_277)	32	RW	0000_0000h
18ACh	Peripheral Domain Access Control (PDAC_W1_277)	32	RW	0000_0000h
18B0h	Peripheral Domain Access Control (PDAC_W0_278)	32	RW	0000_0000h
18B4h	Peripheral Domain Access Control (PDAC_W1_278)	32	RW	0000_0000h
18B8h	Peripheral Domain Access Control (PDAC_W0_279)	32	RW	0000_0000h
18BCh	Peripheral Domain Access Control (PDAC_W1_279)	32	RW	0000_0000h
18C0h	Peripheral Domain Access Control (PDAC_W0_280)	32	RW	0000_0000h
18C4h	Peripheral Domain Access Control (PDAC_W1_280)	32	RW	0000_0000h
18C8h	Peripheral Domain Access Control (PDAC_W0_281)	32	RW	0000_0000h
18CCh	Peripheral Domain Access Control (PDAC_W1_281)	32	RW	0000_0000h
18D0h	Peripheral Domain Access Control (PDAC_W0_282)	32	RW	0000_0000h
18D4h	Peripheral Domain Access Control (PDAC_W1_282)	32	RW	0000_0000h
18D8h	Peripheral Domain Access Control (PDAC_W0_283)	32	RW	0000_0000h
18DCh	Peripheral Domain Access Control (PDAC_W1_283)	32	RW	0000_0000h
18E0h	Peripheral Domain Access Control (PDAC_W0_284)	32	RW	0000_0000h
18E4h	Peripheral Domain Access Control (PDAC_W1_284)	32	RW	0000_0000h
18E8h	Peripheral Domain Access Control (PDAC_W0_285)	32	RW	0000_0000h
18ECh	Peripheral Domain Access Control (PDAC_W1_285)	32	RW	0000_0000h
18F0h	Peripheral Domain Access Control (PDAC_W0_286)	32	RW	0000_0000h
18F4h	Peripheral Domain Access Control (PDAC_W1_286)	32	RW	0000_0000h
18F8h	Peripheral Domain Access Control (PDAC_W0_287)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
18FCh	Peripheral Domain Access Control (PDAC_W1_287)	32	RW	0000_0000h
1C00h	Peripheral Domain Access Control (PDAC_W0_384)	32	RW	0000_0000h
1C04h	Peripheral Domain Access Control (PDAC_W1_384)	32	RW	0000_0000h
1C08h	Peripheral Domain Access Control (PDAC_W0_385)	32	RW	0000_0000h
1C0Ch	Peripheral Domain Access Control (PDAC_W1_385)	32	RW	0000_0000h
1C10h	Peripheral Domain Access Control (PDAC_W0_386)	32	RW	0000_0000h
1C14h	Peripheral Domain Access Control (PDAC_W1_386)	32	RW	0000_0000h
1C18h	Peripheral Domain Access Control (PDAC_W0_387)	32	RW	0000_0000h
1C1Ch	Peripheral Domain Access Control (PDAC_W1_387)	32	RW	0000_0000h
1C20h	Peripheral Domain Access Control (PDAC_W0_388)	32	RW	0000_0000h
1C24h	Peripheral Domain Access Control (PDAC_W1_388)	32	RW	0000_0000h
1C28h	Peripheral Domain Access Control (PDAC_W0_389)	32	RW	0000_0000h
1C2Ch	Peripheral Domain Access Control (PDAC_W1_389)	32	RW	0000_0000h
1C30h	Peripheral Domain Access Control (PDAC_W0_390)	32	RW	0000_0000h
1C34h	Peripheral Domain Access Control (PDAC_W1_390)	32	RW	0000_0000h
1C38h	Peripheral Domain Access Control (PDAC_W0_391)	32	RW	0000_0000h
1C3Ch	Peripheral Domain Access Control (PDAC_W1_391)	32	RW	0000_0000h
1C40h	Peripheral Domain Access Control (PDAC_W0_392)	32	RW	0000_0000h
1C44h	Peripheral Domain Access Control (PDAC_W1_392)	32	RW	0000_0000h
1C48h	Peripheral Domain Access Control (PDAC_W0_393)	32	RW	0000_0000h
1C4Ch	Peripheral Domain Access Control (PDAC_W1_393)	32	RW	0000_0000h
1C50h	Peripheral Domain Access Control (PDAC_W0_394)	32	RW	0000_0000h
1C54h	Peripheral Domain Access Control (PDAC_W1_394)	32	RW	0000_0000h
1C58h	Peripheral Domain Access Control (PDAC_W0_395)	32	RW	0000_0000h
1C5Ch	Peripheral Domain Access Control (PDAC_W1_395)	32	RW	0000_0000h
1C60h	Peripheral Domain Access Control (PDAC_W0_396)	32	RW	0000_0000h
1C64h	Peripheral Domain Access Control (PDAC_W1_396)	32	RW	0000_0000h
1C68h	Peripheral Domain Access Control (PDAC_W0_397)	32	RW	0000_0000h
1C6Ch	Peripheral Domain Access Control (PDAC_W1_397)	32	RW	0000_0000h
1C70h	Peripheral Domain Access Control (PDAC_W0_398)	32	RW	0000_0000h
1C74h	Peripheral Domain Access Control (PDAC_W1_398)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1C80h	Peripheral Domain Access Control (PDAC_W0_400)	32	RW	0000_0000h
1C84h	Peripheral Domain Access Control (PDAC_W1_400)	32	RW	0000_0000h
1C88h	Peripheral Domain Access Control (PDAC_W0_401)	32	RW	0000_0000h
1C8Ch	Peripheral Domain Access Control (PDAC_W1_401)	32	RW	0000_0000h
1CB0h	Peripheral Domain Access Control (PDAC_W0_406)	32	RW	0000_0000h
1CB4h	Peripheral Domain Access Control (PDAC_W1_406)	32	RW	0000_0000h
1CB8h	Peripheral Domain Access Control (PDAC_W0_407)	32	RW	0000_0000h
1CBCh	Peripheral Domain Access Control (PDAC_W1_407)	32	RW	0000_0000h
1CC0h	Peripheral Domain Access Control (PDAC_W0_408)	32	RW	0000_0000h
1CC4h	Peripheral Domain Access Control (PDAC_W1_408)	32	RW	0000_0000h
2000h	Memory Region Descriptor (MRGD_W0_0)	32	RW	0000_0001h
2004h	Memory Region Descriptor (MRGD_W1_0)	32	RW	0000_0001h
2008h	Memory Region Descriptor (MRGD_W2_0)	32	RW	0000_0000h
200Ch	Memory Region Descriptor (MRGD_W3_0)	32	RW	0000_0000h
2020h	Memory Region Descriptor (MRGD_W0_1)	32	RW	0000_0001h
2024h	Memory Region Descriptor (MRGD_W1_1)	32	RW	0000_0001h
2028h	Memory Region Descriptor (MRGD_W2_1)	32	RW	0000_0000h
202Ch	Memory Region Descriptor (MRGD_W3_1)	32	RW	0000_0000h
2040h	Memory Region Descriptor (MRGD_W0_2)	32	RW	0000_0001h
2044h	Memory Region Descriptor (MRGD_W1_2)	32	RW	0000_0001h
2048h	Memory Region Descriptor (MRGD_W2_2)	32	RW	0000_0000h
204Ch	Memory Region Descriptor (MRGD_W3_2)	32	RW	0000_0000h
2060h	Memory Region Descriptor (MRGD_W0_3)	32	RW	0000_0001h
2064h	Memory Region Descriptor (MRGD_W1_3)	32	RW	0000_0001h
2068h	Memory Region Descriptor (MRGD_W2_3)	32	RW	0000_0000h
206Ch	Memory Region Descriptor (MRGD_W3_3)	32	RW	0000_0000h
2080h	Memory Region Descriptor (MRGD_W0_4)	32	RW	0000_0001h
2084h	Memory Region Descriptor (MRGD_W1_4)	32	RW	0000_0001h
2088h	Memory Region Descriptor (MRGD_W2_4)	32	RW	0000_0000h
208Ch	Memory Region Descriptor (MRGD_W3_4)	32	RW	0000_0000h
20A0h	Memory Region Descriptor (MRGD_W0_5)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
20A4h	Memory Region Descriptor (MRGD_W1_5)	32	RW	0000_0001h
20A8h	Memory Region Descriptor (MRGD_W2_5)	32	RW	0000_0000h
20ACh	Memory Region Descriptor (MRGD_W3_5)	32	RW	0000_0000h
20C0h	Memory Region Descriptor (MRGD_W0_6)	32	RW	0000_0001h
20C4h	Memory Region Descriptor (MRGD_W1_6)	32	RW	0000_0001h
20C8h	Memory Region Descriptor (MRGD_W2_6)	32	RW	0000_0000h
20CCh	Memory Region Descriptor (MRGD_W3_6)	32	RW	0000_0000h
20E0h	Memory Region Descriptor (MRGD_W0_7)	32	RW	0000_0001h
20E4h	Memory Region Descriptor (MRGD_W1_7)	32	RW	0000_0001h
20E8h	Memory Region Descriptor (MRGD_W2_7)	32	RW	0000_0000h
20ECh	Memory Region Descriptor (MRGD_W3_7)	32	RW	0000_0000h
2100h	Memory Region Descriptor (MRGD_W0_8)	32	RW	0000_0001h
2104h	Memory Region Descriptor (MRGD_W1_8)	32	RW	0000_0001h
2108h	Memory Region Descriptor (MRGD_W2_8)	32	RW	0000_0000h
210Ch	Memory Region Descriptor (MRGD_W3_8)	32	RW	0000_0000h
2120h	Memory Region Descriptor (MRGD_W0_9)	32	RW	0000_0001h
2124h	Memory Region Descriptor (MRGD_W1_9)	32	RW	0000_0001h
2128h	Memory Region Descriptor (MRGD_W2_9)	32	RW	0000_0000h
212Ch	Memory Region Descriptor (MRGD_W3_9)	32	RW	0000_0000h
2140h	Memory Region Descriptor (MRGD_W0_10)	32	RW	0000_0001h
2144h	Memory Region Descriptor (MRGD_W1_10)	32	RW	0000_0001h
2148h	Memory Region Descriptor (MRGD_W2_10)	32	RW	0000_0000h
214Ch	Memory Region Descriptor (MRGD_W3_10)	32	RW	0000_0000h
2160h	Memory Region Descriptor (MRGD_W0_11)	32	RW	0000_0001h
2164h	Memory Region Descriptor (MRGD_W1_11)	32	RW	0000_0001h
2168h	Memory Region Descriptor (MRGD_W2_11)	32	RW	0000_0000h
216Ch	Memory Region Descriptor (MRGD_W3_11)	32	RW	0000_0000h
2180h	Memory Region Descriptor (MRGD_W0_12)	32	RW	0000_0001h
2184h	Memory Region Descriptor (MRGD_W1_12)	32	RW	0000_0001h
2188h	Memory Region Descriptor (MRGD_W2_12)	32	RW	0000_0000h
218Ch	Memory Region Descriptor (MRGD_W3_12)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
21A0h	Memory Region Descriptor (MRGD_W0_13)	32	RW	0000_0001h
21A4h	Memory Region Descriptor (MRGD_W1_13)	32	RW	0000_0001h
21A8h	Memory Region Descriptor (MRGD_W2_13)	32	RW	0000_0000h
21ACh	Memory Region Descriptor (MRGD_W3_13)	32	RW	0000_0000h
21C0h	Memory Region Descriptor (MRGD_W0_14)	32	RW	0000_0001h
21C4h	Memory Region Descriptor (MRGD_W1_14)	32	RW	0000_0001h
21C8h	Memory Region Descriptor (MRGD_W2_14)	32	RW	0000_0000h
21CCh	Memory Region Descriptor (MRGD_W3_14)	32	RW	0000_0000h
21E0h	Memory Region Descriptor (MRGD_W0_15)	32	RW	0000_0001h
21E4h	Memory Region Descriptor (MRGD_W1_15)	32	RW	0000_0001h
21E8h	Memory Region Descriptor (MRGD_W2_15)	32	RW	0000_0000h
21ECh	Memory Region Descriptor (MRGD_W3_15)	32	RW	0000_0000h
2400h	Memory Region Descriptor (MRGD_W0_32)	32	RW	0000_0001h
2404h	Memory Region Descriptor (MRGD_W1_32)	32	RW	0000_0001h
2408h	Memory Region Descriptor (MRGD_W2_32)	32	RW	0000_0000h
240Ch	Memory Region Descriptor (MRGD_W3_32)	32	RW	0000_0000h
2420h	Memory Region Descriptor (MRGD_W0_33)	32	RW	0000_0001h
2424h	Memory Region Descriptor (MRGD_W1_33)	32	RW	0000_0001h
2428h	Memory Region Descriptor (MRGD_W2_33)	32	RW	0000_0000h
242Ch	Memory Region Descriptor (MRGD_W3_33)	32	RW	0000_0000h
2440h	Memory Region Descriptor (MRGD_W0_34)	32	RW	0000_0001h
2444h	Memory Region Descriptor (MRGD_W1_34)	32	RW	0000_0001h
2448h	Memory Region Descriptor (MRGD_W2_34)	32	RW	0000_0000h
244Ch	Memory Region Descriptor (MRGD_W3_34)	32	RW	0000_0000h
2460h	Memory Region Descriptor (MRGD_W0_35)	32	RW	0000_0001h
2464h	Memory Region Descriptor (MRGD_W1_35)	32	RW	0000_0001h
2468h	Memory Region Descriptor (MRGD_W2_35)	32	RW	0000_0000h
246Ch	Memory Region Descriptor (MRGD_W3_35)	32	RW	0000_0000h
2480h	Memory Region Descriptor (MRGD_W0_36)	32	RW	0000_0001h
2484h	Memory Region Descriptor (MRGD_W1_36)	32	RW	0000_0001h
2488h	Memory Region Descriptor (MRGD_W2_36)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
248Ch	Memory Region Descriptor (MRGD_W3_36)	32	RW	0000_0000h
24A0h	Memory Region Descriptor (MRGD_W0_37)	32	RW	0000_0001h
24A4h	Memory Region Descriptor (MRGD_W1_37)	32	RW	0000_0001h
24A8h	Memory Region Descriptor (MRGD_W2_37)	32	RW	0000_0000h
24ACh	Memory Region Descriptor (MRGD_W3_37)	32	RW	0000_0000h
24C0h	Memory Region Descriptor (MRGD_W0_38)	32	RW	0000_0001h
24C4h	Memory Region Descriptor (MRGD_W1_38)	32	RW	0000_0001h
24C8h	Memory Region Descriptor (MRGD_W2_38)	32	RW	0000_0000h
24CCh	Memory Region Descriptor (MRGD_W3_38)	32	RW	0000_0000h
24E0h	Memory Region Descriptor (MRGD_W0_39)	32	RW	0000_0001h
24E4h	Memory Region Descriptor (MRGD_W1_39)	32	RW	0000_0001h
24E8h	Memory Region Descriptor (MRGD_W2_39)	32	RW	0000_0000h
24ECh	Memory Region Descriptor (MRGD_W3_39)	32	RW	0000_0000h
2500h	Memory Region Descriptor (MRGD_W0_40)	32	RW	0000_0001h
2504h	Memory Region Descriptor (MRGD_W1_40)	32	RW	0000_0001h
2508h	Memory Region Descriptor (MRGD_W2_40)	32	RW	0000_0000h
250Ch	Memory Region Descriptor (MRGD_W3_40)	32	RW	0000_0000h
2520h	Memory Region Descriptor (MRGD_W0_41)	32	RW	0000_0001h
2524h	Memory Region Descriptor (MRGD_W1_41)	32	RW	0000_0001h
2528h	Memory Region Descriptor (MRGD_W2_41)	32	RW	0000_0000h
252Ch	Memory Region Descriptor (MRGD_W3_41)	32	RW	0000_0000h
2540h	Memory Region Descriptor (MRGD_W0_42)	32	RW	0000_0001h
2544h	Memory Region Descriptor (MRGD_W1_42)	32	RW	0000_0001h
2548h	Memory Region Descriptor (MRGD_W2_42)	32	RW	0000_0000h
254Ch	Memory Region Descriptor (MRGD_W3_42)	32	RW	0000_0000h
2560h	Memory Region Descriptor (MRGD_W0_43)	32	RW	0000_0001h
2564h	Memory Region Descriptor (MRGD_W1_43)	32	RW	0000_0001h
2568h	Memory Region Descriptor (MRGD_W2_43)	32	RW	0000_0000h
256Ch	Memory Region Descriptor (MRGD_W3_43)	32	RW	0000_0000h
2580h	Memory Region Descriptor (MRGD_W0_44)	32	RW	0000_0001h
2584h	Memory Region Descriptor (MRGD_W1_44)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2588h	Memory Region Descriptor (MRGD_W2_44)	32	RW	0000_0000h
258Ch	Memory Region Descriptor (MRGD_W3_44)	32	RW	0000_0000h
25A0h	Memory Region Descriptor (MRGD_W0_45)	32	RW	0000_0001h
25A4h	Memory Region Descriptor (MRGD_W1_45)	32	RW	0000_0001h
25A8h	Memory Region Descriptor (MRGD_W2_45)	32	RW	0000_0000h
25ACh	Memory Region Descriptor (MRGD_W3_45)	32	RW	0000_0000h
25C0h	Memory Region Descriptor (MRGD_W0_46)	32	RW	0000_0001h
25C4h	Memory Region Descriptor (MRGD_W1_46)	32	RW	0000_0001h
25C8h	Memory Region Descriptor (MRGD_W2_46)	32	RW	0000_0000h
25CCh	Memory Region Descriptor (MRGD_W3_46)	32	RW	0000_0000h
25E0h	Memory Region Descriptor (MRGD_W0_47)	32	RW	0000_0001h
25E4h	Memory Region Descriptor (MRGD_W1_47)	32	RW	0000_0001h
25E8h	Memory Region Descriptor (MRGD_W2_47)	32	RW	0000_0000h
25ECh	Memory Region Descriptor (MRGD_W3_47)	32	RW	0000_0000h
2600h	Memory Region Descriptor (MRGD_W0_48)	32	RW	0000_0001h
2604h	Memory Region Descriptor (MRGD_W1_48)	32	RW	0000_0001h
2608h	Memory Region Descriptor (MRGD_W2_48)	32	RW	0000_0000h
260Ch	Memory Region Descriptor (MRGD_W3_48)	32	RW	0000_0000h
2620h	Memory Region Descriptor (MRGD_W0_49)	32	RW	0000_0001h
2624h	Memory Region Descriptor (MRGD_W1_49)	32	RW	0000_0001h
2628h	Memory Region Descriptor (MRGD_W2_49)	32	RW	0000_0000h
262Ch	Memory Region Descriptor (MRGD_W3_49)	32	RW	0000_0000h
2640h	Memory Region Descriptor (MRGD_W0_50)	32	RW	0000_0001h
2644h	Memory Region Descriptor (MRGD_W1_50)	32	RW	0000_0001h
2648h	Memory Region Descriptor (MRGD_W2_50)	32	RW	0000_0000h
264Ch	Memory Region Descriptor (MRGD_W3_50)	32	RW	0000_0000h
2660h	Memory Region Descriptor (MRGD_W0_51)	32	RW	0000_0001h
2664h	Memory Region Descriptor (MRGD_W1_51)	32	RW	0000_0001h
2668h	Memory Region Descriptor (MRGD_W2_51)	32	RW	0000_0000h
266Ch	Memory Region Descriptor (MRGD_W3_51)	32	RW	0000_0000h
2680h	Memory Region Descriptor (MRGD_W0_52)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2684h	Memory Region Descriptor (MRGD_W1_52)	32	RW	0000_0001h
2688h	Memory Region Descriptor (MRGD_W2_52)	32	RW	0000_0000h
268Ch	Memory Region Descriptor (MRGD_W3_52)	32	RW	0000_0000h
26A0h	Memory Region Descriptor (MRGD_W0_53)	32	RW	0000_0001h
26A4h	Memory Region Descriptor (MRGD_W1_53)	32	RW	0000_0001h
26A8h	Memory Region Descriptor (MRGD_W2_53)	32	RW	0000_0000h
26ACh	Memory Region Descriptor (MRGD_W3_53)	32	RW	0000_0000h
26C0h	Memory Region Descriptor (MRGD_W0_54)	32	RW	0000_0001h
26C4h	Memory Region Descriptor (MRGD_W1_54)	32	RW	0000_0001h
26C8h	Memory Region Descriptor (MRGD_W2_54)	32	RW	0000_0000h
26CCh	Memory Region Descriptor (MRGD_W3_54)	32	RW	0000_0000h
26E0h	Memory Region Descriptor (MRGD_W0_55)	32	RW	0000_0001h
26E4h	Memory Region Descriptor (MRGD_W1_55)	32	RW	0000_0001h
26E8h	Memory Region Descriptor (MRGD_W2_55)	32	RW	0000_0000h
26ECh	Memory Region Descriptor (MRGD_W3_55)	32	RW	0000_0000h
2700h	Memory Region Descriptor (MRGD_W0_56)	32	RW	0000_0001h
2704h	Memory Region Descriptor (MRGD_W1_56)	32	RW	0000_0001h
2708h	Memory Region Descriptor (MRGD_W2_56)	32	RW	0000_0000h
270Ch	Memory Region Descriptor (MRGD_W3_56)	32	RW	0000_0000h
2720h	Memory Region Descriptor (MRGD_W0_57)	32	RW	0000_0001h
2724h	Memory Region Descriptor (MRGD_W1_57)	32	RW	0000_0001h
2728h	Memory Region Descriptor (MRGD_W2_57)	32	RW	0000_0000h
272Ch	Memory Region Descriptor (MRGD_W3_57)	32	RW	0000_0000h
2740h	Memory Region Descriptor (MRGD_W0_58)	32	RW	0000_0001h
2744h	Memory Region Descriptor (MRGD_W1_58)	32	RW	0000_0001h
2748h	Memory Region Descriptor (MRGD_W2_58)	32	RW	0000_0000h
274Ch	Memory Region Descriptor (MRGD_W3_58)	32	RW	0000_0000h
2760h	Memory Region Descriptor (MRGD_W0_59)	32	RW	0000_0001h
2764h	Memory Region Descriptor (MRGD_W1_59)	32	RW	0000_0001h
2768h	Memory Region Descriptor (MRGD_W2_59)	32	RW	0000_0000h
276Ch	Memory Region Descriptor (MRGD_W3_59)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2780h	Memory Region Descriptor (MRGD_W0_60)	32	RW	0000_0001h
2784h	Memory Region Descriptor (MRGD_W1_60)	32	RW	0000_0001h
2788h	Memory Region Descriptor (MRGD_W2_60)	32	RW	0000_0000h
278Ch	Memory Region Descriptor (MRGD_W3_60)	32	RW	0000_0000h
27A0h	Memory Region Descriptor (MRGD_W0_61)	32	RW	0000_0001h
27A4h	Memory Region Descriptor (MRGD_W1_61)	32	RW	0000_0001h
27A8h	Memory Region Descriptor (MRGD_W2_61)	32	RW	0000_0000h
27ACh	Memory Region Descriptor (MRGD_W3_61)	32	RW	0000_0000h
27C0h	Memory Region Descriptor (MRGD_W0_62)	32	RW	0000_0001h
27C4h	Memory Region Descriptor (MRGD_W1_62)	32	RW	0000_0001h
27C8h	Memory Region Descriptor (MRGD_W2_62)	32	RW	0000_0000h
27CCh	Memory Region Descriptor (MRGD_W3_62)	32	RW	0000_0000h
27E0h	Memory Region Descriptor (MRGD_W0_63)	32	RW	0000_0001h
27E4h	Memory Region Descriptor (MRGD_W1_63)	32	RW	0000_0001h
27E8h	Memory Region Descriptor (MRGD_W2_63)	32	RW	0000_0000h
27ECh	Memory Region Descriptor (MRGD_W3_63)	32	RW	0000_0000h
2800h	Memory Region Descriptor (MRGD_W0_64)	32	RW	0000_0001h
2804h	Memory Region Descriptor (MRGD_W1_64)	32	RW	0000_0001h
2808h	Memory Region Descriptor (MRGD_W2_64)	32	RW	0000_0000h
280Ch	Memory Region Descriptor (MRGD_W3_64)	32	RW	0000_0000h
2820h	Memory Region Descriptor (MRGD_W0_65)	32	RW	0000_0001h
2824h	Memory Region Descriptor (MRGD_W1_65)	32	RW	0000_0001h
2828h	Memory Region Descriptor (MRGD_W2_65)	32	RW	0000_0000h
282Ch	Memory Region Descriptor (MRGD_W3_65)	32	RW	0000_0000h
2840h	Memory Region Descriptor (MRGD_W0_66)	32	RW	0000_0001h
2844h	Memory Region Descriptor (MRGD_W1_66)	32	RW	0000_0001h
2848h	Memory Region Descriptor (MRGD_W2_66)	32	RW	0000_0000h
284Ch	Memory Region Descriptor (MRGD_W3_66)	32	RW	0000_0000h
2860h	Memory Region Descriptor (MRGD_W0_67)	32	RW	0000_0001h
2864h	Memory Region Descriptor (MRGD_W1_67)	32	RW	0000_0001h
2868h	Memory Region Descriptor (MRGD_W2_67)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
286Ch	Memory Region Descriptor (MRGD_W3_67)	32	RW	0000_0000h
2880h	Memory Region Descriptor (MRGD_W0_68)	32	RW	0000_0001h
2884h	Memory Region Descriptor (MRGD_W1_68)	32	RW	0000_0001h
2888h	Memory Region Descriptor (MRGD_W2_68)	32	RW	0000_0000h
288Ch	Memory Region Descriptor (MRGD_W3_68)	32	RW	0000_0000h
28A0h	Memory Region Descriptor (MRGD_W0_69)	32	RW	0000_0001h
28A4h	Memory Region Descriptor (MRGD_W1_69)	32	RW	0000_0001h
28A8h	Memory Region Descriptor (MRGD_W2_69)	32	RW	0000_0000h
28ACh	Memory Region Descriptor (MRGD_W3_69)	32	RW	0000_0000h
28C0h	Memory Region Descriptor (MRGD_W0_70)	32	RW	0000_0001h
28C4h	Memory Region Descriptor (MRGD_W1_70)	32	RW	0000_0001h
28C8h	Memory Region Descriptor (MRGD_W2_70)	32	RW	0000_0000h
28CCh	Memory Region Descriptor (MRGD_W3_70)	32	RW	0000_0000h
28E0h	Memory Region Descriptor (MRGD_W0_71)	32	RW	0000_0001h
28E4h	Memory Region Descriptor (MRGD_W1_71)	32	RW	0000_0001h
28E8h	Memory Region Descriptor (MRGD_W2_71)	32	RW	0000_0000h
28ECh	Memory Region Descriptor (MRGD_W3_71)	32	RW	0000_0000h
2900h	Memory Region Descriptor (MRGD_W0_72)	32	RW	0000_0001h
2904h	Memory Region Descriptor (MRGD_W1_72)	32	RW	0000_0001h
2908h	Memory Region Descriptor (MRGD_W2_72)	32	RW	0000_0000h
290Ch	Memory Region Descriptor (MRGD_W3_72)	32	RW	0000_0000h
2920h	Memory Region Descriptor (MRGD_W0_73)	32	RW	0000_0001h
2924h	Memory Region Descriptor (MRGD_W1_73)	32	RW	0000_0001h
2928h	Memory Region Descriptor (MRGD_W2_73)	32	RW	0000_0000h
292Ch	Memory Region Descriptor (MRGD_W3_73)	32	RW	0000_0000h
2940h	Memory Region Descriptor (MRGD_W0_74)	32	RW	0000_0001h
2944h	Memory Region Descriptor (MRGD_W1_74)	32	RW	0000_0001h
2948h	Memory Region Descriptor (MRGD_W2_74)	32	RW	0000_0000h
294Ch	Memory Region Descriptor (MRGD_W3_74)	32	RW	0000_0000h
2960h	Memory Region Descriptor (MRGD_W0_75)	32	RW	0000_0001h
2964h	Memory Region Descriptor (MRGD_W1_75)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2968h	Memory Region Descriptor (MRGD_W2_75)	32	RW	0000_0000h
296Ch	Memory Region Descriptor (MRGD_W3_75)	32	RW	0000_0000h
2980h	Memory Region Descriptor (MRGD_W0_76)	32	RW	0000_0001h
2984h	Memory Region Descriptor (MRGD_W1_76)	32	RW	0000_0001h
2988h	Memory Region Descriptor (MRGD_W2_76)	32	RW	0000_0000h
298Ch	Memory Region Descriptor (MRGD_W3_76)	32	RW	0000_0000h
29A0h	Memory Region Descriptor (MRGD_W0_77)	32	RW	0000_0001h
29A4h	Memory Region Descriptor (MRGD_W1_77)	32	RW	0000_0001h
29A8h	Memory Region Descriptor (MRGD_W2_77)	32	RW	0000_0000h
29ACh	Memory Region Descriptor (MRGD_W3_77)	32	RW	0000_0000h
29C0h	Memory Region Descriptor (MRGD_W0_78)	32	RW	0000_0001h
29C4h	Memory Region Descriptor (MRGD_W1_78)	32	RW	0000_0001h
29C8h	Memory Region Descriptor (MRGD_W2_78)	32	RW	0000_0000h
29CCh	Memory Region Descriptor (MRGD_W3_78)	32	RW	0000_0000h
29E0h	Memory Region Descriptor (MRGD_W0_79)	32	RW	0000_0001h
29E4h	Memory Region Descriptor (MRGD_W1_79)	32	RW	0000_0001h
29E8h	Memory Region Descriptor (MRGD_W2_79)	32	RW	0000_0000h
29ECh	Memory Region Descriptor (MRGD_W3_79)	32	RW	0000_0000h
2A00h	Memory Region Descriptor (MRGD_W0_80)	32	RW	0000_0001h
2A04h	Memory Region Descriptor (MRGD_W1_80)	32	RW	0000_0001h
2A08h	Memory Region Descriptor (MRGD_W2_80)	32	RW	0000_0000h
2A0Ch	Memory Region Descriptor (MRGD_W3_80)	32	RW	0000_0000h
2A20h	Memory Region Descriptor (MRGD_W0_81)	32	RW	0000_0001h
2A24h	Memory Region Descriptor (MRGD_W1_81)	32	RW	0000_0001h
2A28h	Memory Region Descriptor (MRGD_W2_81)	32	RW	0000_0000h
2A2Ch	Memory Region Descriptor (MRGD_W3_81)	32	RW	0000_0000h
2A40h	Memory Region Descriptor (MRGD_W0_82)	32	RW	0000_0001h
2A44h	Memory Region Descriptor (MRGD_W1_82)	32	RW	0000_0001h
2A48h	Memory Region Descriptor (MRGD_W2_82)	32	RW	0000_0000h
2A4Ch	Memory Region Descriptor (MRGD_W3_82)	32	RW	0000_0000h
2A60h	Memory Region Descriptor (MRGD_W0_83)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2A64h	Memory Region Descriptor (MRGD_W1_83)	32	RW	0000_0001h
2A68h	Memory Region Descriptor (MRGD_W2_83)	32	RW	0000_0000h
2A6Ch	Memory Region Descriptor (MRGD_W3_83)	32	RW	0000_0000h
2A80h	Memory Region Descriptor (MRGD_W0_84)	32	RW	0000_0001h
2A84h	Memory Region Descriptor (MRGD_W1_84)	32	RW	0000_0001h
2A88h	Memory Region Descriptor (MRGD_W2_84)	32	RW	0000_0000h
2A8Ch	Memory Region Descriptor (MRGD_W3_84)	32	RW	0000_0000h
2AA0h	Memory Region Descriptor (MRGD_W0_85)	32	RW	0000_0001h
2AA4h	Memory Region Descriptor (MRGD_W1_85)	32	RW	0000_0001h
2AA8h	Memory Region Descriptor (MRGD_W2_85)	32	RW	0000_0000h
2AACh	Memory Region Descriptor (MRGD_W3_85)	32	RW	0000_0000h
2AC0h	Memory Region Descriptor (MRGD_W0_86)	32	RW	0000_0001h
2AC4h	Memory Region Descriptor (MRGD_W1_86)	32	RW	0000_0001h
2AC8h	Memory Region Descriptor (MRGD_W2_86)	32	RW	0000_0000h
2ACCh	Memory Region Descriptor (MRGD_W3_86)	32	RW	0000_0000h
2AE0h	Memory Region Descriptor (MRGD_W0_87)	32	RW	0000_0001h
2AE4h	Memory Region Descriptor (MRGD_W1_87)	32	RW	0000_0001h
2AE8h	Memory Region Descriptor (MRGD_W2_87)	32	RW	0000_0000h
2AECh	Memory Region Descriptor (MRGD_W3_87)	32	RW	0000_0000h
2B00h	Memory Region Descriptor (MRGD_W0_88)	32	RW	0000_0001h
2B04h	Memory Region Descriptor (MRGD_W1_88)	32	RW	0000_0001h
2B08h	Memory Region Descriptor (MRGD_W2_88)	32	RW	0000_0000h
2B0Ch	Memory Region Descriptor (MRGD_W3_88)	32	RW	0000_0000h
2B20h	Memory Region Descriptor (MRGD_W0_89)	32	RW	0000_0001h
2B24h	Memory Region Descriptor (MRGD_W1_89)	32	RW	0000_0001h
2B28h	Memory Region Descriptor (MRGD_W2_89)	32	RW	0000_0000h
2B2Ch	Memory Region Descriptor (MRGD_W3_89)	32	RW	0000_0000h
2B40h	Memory Region Descriptor (MRGD_W0_90)	32	RW	0000_0001h
2B44h	Memory Region Descriptor (MRGD_W1_90)	32	RW	0000_0001h
2B48h	Memory Region Descriptor (MRGD_W2_90)	32	RW	0000_0000h
2B4Ch	Memory Region Descriptor (MRGD_W3_90)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2B60h	Memory Region Descriptor (MRGD_W0_91)	32	RW	0000_0001h
2B64h	Memory Region Descriptor (MRGD_W1_91)	32	RW	0000_0001h
2B68h	Memory Region Descriptor (MRGD_W2_91)	32	RW	0000_0000h
2B6Ch	Memory Region Descriptor (MRGD_W3_91)	32	RW	0000_0000h
2B80h	Memory Region Descriptor (MRGD_W0_92)	32	RW	0000_0001h
2B84h	Memory Region Descriptor (MRGD_W1_92)	32	RW	0000_0001h
2B88h	Memory Region Descriptor (MRGD_W2_92)	32	RW	0000_0000h
2B8Ch	Memory Region Descriptor (MRGD_W3_92)	32	RW	0000_0000h
2BA0h	Memory Region Descriptor (MRGD_W0_93)	32	RW	0000_0001h
2BA4h	Memory Region Descriptor (MRGD_W1_93)	32	RW	0000_0001h
2BA8h	Memory Region Descriptor (MRGD_W2_93)	32	RW	0000_0000h
2BACh	Memory Region Descriptor (MRGD_W3_93)	32	RW	0000_0000h
2BC0h	Memory Region Descriptor (MRGD_W0_94)	32	RW	0000_0001h
2BC4h	Memory Region Descriptor (MRGD_W1_94)	32	RW	0000_0001h
2BC8h	Memory Region Descriptor (MRGD_W2_94)	32	RW	0000_0000h
2BCCh	Memory Region Descriptor (MRGD_W3_94)	32	RW	0000_0000h
2BE0h	Memory Region Descriptor (MRGD_W0_95)	32	RW	0000_0001h
2BE4h	Memory Region Descriptor (MRGD_W1_95)	32	RW	0000_0001h
2BE8h	Memory Region Descriptor (MRGD_W2_95)	32	RW	0000_0000h
2BECh	Memory Region Descriptor (MRGD_W3_95)	32	RW	0000_0000h
2C00h	Memory Region Descriptor (MRGD_W0_96)	32	RW	0000_0001h
2C04h	Memory Region Descriptor (MRGD_W1_96)	32	RW	0000_0001h
2C08h	Memory Region Descriptor (MRGD_W2_96)	32	RW	0000_0000h
2C0Ch	Memory Region Descriptor (MRGD_W3_96)	32	RW	0000_0000h
2C20h	Memory Region Descriptor (MRGD_W0_97)	32	RW	0000_0001h
2C24h	Memory Region Descriptor (MRGD_W1_97)	32	RW	0000_0001h
2C28h	Memory Region Descriptor (MRGD_W2_97)	32	RW	0000_0000h
2C2Ch	Memory Region Descriptor (MRGD_W3_97)	32	RW	0000_0000h
2C40h	Memory Region Descriptor (MRGD_W0_98)	32	RW	0000_0001h
2C44h	Memory Region Descriptor (MRGD_W1_98)	32	RW	0000_0001h
2C48h	Memory Region Descriptor (MRGD_W2_98)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2C4Ch	Memory Region Descriptor (MRGD_W3_98)	32	RW	0000_0000h
2C60h	Memory Region Descriptor (MRGD_W0_99)	32	RW	0000_0001h
2C64h	Memory Region Descriptor (MRGD_W1_99)	32	RW	0000_0001h
2C68h	Memory Region Descriptor (MRGD_W2_99)	32	RW	0000_0000h
2C6Ch	Memory Region Descriptor (MRGD_W3_99)	32	RW	0000_0000h
2C80h	Memory Region Descriptor (MRGD_W0_100)	32	RW	0000_0001h
2C84h	Memory Region Descriptor (MRGD_W1_100)	32	RW	0000_0001h
2C88h	Memory Region Descriptor (MRGD_W2_100)	32	RW	0000_0000h
2C8Ch	Memory Region Descriptor (MRGD_W3_100)	32	RW	0000_0000h
2CA0h	Memory Region Descriptor (MRGD_W0_101)	32	RW	0000_0001h
2CA4h	Memory Region Descriptor (MRGD_W1_101)	32	RW	0000_0001h
2CA8h	Memory Region Descriptor (MRGD_W2_101)	32	RW	0000_0000h
2CACh	Memory Region Descriptor (MRGD_W3_101)	32	RW	0000_0000h
2CC0h	Memory Region Descriptor (MRGD_W0_102)	32	RW	0000_0001h
2CC4h	Memory Region Descriptor (MRGD_W1_102)	32	RW	0000_0001h
2CC8h	Memory Region Descriptor (MRGD_W2_102)	32	RW	0000_0000h
2CCCh	Memory Region Descriptor (MRGD_W3_102)	32	RW	0000_0000h
2CE0h	Memory Region Descriptor (MRGD_W0_103)	32	RW	0000_0001h
2CE4h	Memory Region Descriptor (MRGD_W1_103)	32	RW	0000_0001h
2CE8h	Memory Region Descriptor (MRGD_W2_103)	32	RW	0000_0000h
2CECh	Memory Region Descriptor (MRGD_W3_103)	32	RW	0000_0000h
2D00h	Memory Region Descriptor (MRGD_W0_104)	32	RW	0000_0001h
2D04h	Memory Region Descriptor (MRGD_W1_104)	32	RW	0000_0001h
2D08h	Memory Region Descriptor (MRGD_W2_104)	32	RW	0000_0000h
2D0Ch	Memory Region Descriptor (MRGD_W3_104)	32	RW	0000_0000h
2D20h	Memory Region Descriptor (MRGD_W0_105)	32	RW	0000_0001h
2D24h	Memory Region Descriptor (MRGD_W1_105)	32	RW	0000_0001h
2D28h	Memory Region Descriptor (MRGD_W2_105)	32	RW	0000_0000h
2D2Ch	Memory Region Descriptor (MRGD_W3_105)	32	RW	0000_0000h
2D40h	Memory Region Descriptor (MRGD_W0_106)	32	RW	0000_0001h
2D44h	Memory Region Descriptor (MRGD_W1_106)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2D48h	Memory Region Descriptor (MRGD_W2_106)	32	RW	0000_0000h
2D4Ch	Memory Region Descriptor (MRGD_W3_106)	32	RW	0000_0000h
2D60h	Memory Region Descriptor (MRGD_W0_107)	32	RW	0000_0001h
2D64h	Memory Region Descriptor (MRGD_W1_107)	32	RW	0000_0001h
2D68h	Memory Region Descriptor (MRGD_W2_107)	32	RW	0000_0000h
2D6Ch	Memory Region Descriptor (MRGD_W3_107)	32	RW	0000_0000h
2D80h	Memory Region Descriptor (MRGD_W0_108)	32	RW	0000_0001h
2D84h	Memory Region Descriptor (MRGD_W1_108)	32	RW	0000_0001h
2D88h	Memory Region Descriptor (MRGD_W2_108)	32	RW	0000_0000h
2D8Ch	Memory Region Descriptor (MRGD_W3_108)	32	RW	0000_0000h
2DA0h	Memory Region Descriptor (MRGD_W0_109)	32	RW	0000_0001h
2DA4h	Memory Region Descriptor (MRGD_W1_109)	32	RW	0000_0001h
2DA8h	Memory Region Descriptor (MRGD_W2_109)	32	RW	0000_0000h
2DACh	Memory Region Descriptor (MRGD_W3_109)	32	RW	0000_0000h
2DC0h	Memory Region Descriptor (MRGD_W0_110)	32	RW	0000_0001h
2DC4h	Memory Region Descriptor (MRGD_W1_110)	32	RW	0000_0001h
2DC8h	Memory Region Descriptor (MRGD_W2_110)	32	RW	0000_0000h
2DCCh	Memory Region Descriptor (MRGD_W3_110)	32	RW	0000_0000h
2DE0h	Memory Region Descriptor (MRGD_W0_111)	32	RW	0000_0001h
2DE4h	Memory Region Descriptor (MRGD_W1_111)	32	RW	0000_0001h
2DE8h	Memory Region Descriptor (MRGD_W2_111)	32	RW	0000_0000h
2DECh	Memory Region Descriptor (MRGD_W3_111)	32	RW	0000_0000h
2E00h	Memory Region Descriptor (MRGD_W0_112)	32	RW	0000_0001h
2E04h	Memory Region Descriptor (MRGD_W1_112)	32	RW	0000_0001h
2E08h	Memory Region Descriptor (MRGD_W2_112)	32	RW	0000_0000h
2E0Ch	Memory Region Descriptor (MRGD_W3_112)	32	RW	0000_0000h
2E20h	Memory Region Descriptor (MRGD_W0_113)	32	RW	0000_0001h
2E24h	Memory Region Descriptor (MRGD_W1_113)	32	RW	0000_0001h
2E28h	Memory Region Descriptor (MRGD_W2_113)	32	RW	0000_0000h
2E2Ch	Memory Region Descriptor (MRGD_W3_113)	32	RW	0000_0000h
2E40h	Memory Region Descriptor (MRGD_W0_114)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2E44h	Memory Region Descriptor (MRGD_W1_114)	32	RW	0000_0001h
2E48h	Memory Region Descriptor (MRGD_W2_114)	32	RW	0000_0000h
2E4Ch	Memory Region Descriptor (MRGD_W3_114)	32	RW	0000_0000h
2E60h	Memory Region Descriptor (MRGD_W0_115)	32	RW	0000_0001h
2E64h	Memory Region Descriptor (MRGD_W1_115)	32	RW	0000_0001h
2E68h	Memory Region Descriptor (MRGD_W2_115)	32	RW	0000_0000h
2E6Ch	Memory Region Descriptor (MRGD_W3_115)	32	RW	0000_0000h
2E80h	Memory Region Descriptor (MRGD_W0_116)	32	RW	0000_0001h
2E84h	Memory Region Descriptor (MRGD_W1_116)	32	RW	0000_0001h
2E88h	Memory Region Descriptor (MRGD_W2_116)	32	RW	0000_0000h
2E8Ch	Memory Region Descriptor (MRGD_W3_116)	32	RW	0000_0000h
2EA0h	Memory Region Descriptor (MRGD_W0_117)	32	RW	0000_0001h
2EA4h	Memory Region Descriptor (MRGD_W1_117)	32	RW	0000_0001h
2EA8h	Memory Region Descriptor (MRGD_W2_117)	32	RW	0000_0000h
2EACh	Memory Region Descriptor (MRGD_W3_117)	32	RW	0000_0000h
2EC0h	Memory Region Descriptor (MRGD_W0_118)	32	RW	0000_0001h
2EC4h	Memory Region Descriptor (MRGD_W1_118)	32	RW	0000_0001h
2EC8h	Memory Region Descriptor (MRGD_W2_118)	32	RW	0000_0000h
2ECCh	Memory Region Descriptor (MRGD_W3_118)	32	RW	0000_0000h
2EE0h	Memory Region Descriptor (MRGD_W0_119)	32	RW	0000_0001h
2EE4h	Memory Region Descriptor (MRGD_W1_119)	32	RW	0000_0001h
2EE8h	Memory Region Descriptor (MRGD_W2_119)	32	RW	0000_0000h
2EECh	Memory Region Descriptor (MRGD_W3_119)	32	RW	0000_0000h
2F00h	Memory Region Descriptor (MRGD_W0_120)	32	RW	0000_0001h
2F04h	Memory Region Descriptor (MRGD_W1_120)	32	RW	0000_0001h
2F08h	Memory Region Descriptor (MRGD_W2_120)	32	RW	0000_0000h
2F0Ch	Memory Region Descriptor (MRGD_W3_120)	32	RW	0000_0000h
2F20h	Memory Region Descriptor (MRGD_W0_121)	32	RW	0000_0001h
2F24h	Memory Region Descriptor (MRGD_W1_121)	32	RW	0000_0001h
2F28h	Memory Region Descriptor (MRGD_W2_121)	32	RW	0000_0000h
2F2Ch	Memory Region Descriptor (MRGD_W3_121)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2F40h	Memory Region Descriptor (MRGD_W0_122)	32	RW	0000_0001h
2F44h	Memory Region Descriptor (MRGD_W1_122)	32	RW	0000_0001h
2F48h	Memory Region Descriptor (MRGD_W2_122)	32	RW	0000_0000h
2F4Ch	Memory Region Descriptor (MRGD_W3_122)	32	RW	0000_0000h
2F60h	Memory Region Descriptor (MRGD_W0_123)	32	RW	0000_0001h
2F64h	Memory Region Descriptor (MRGD_W1_123)	32	RW	0000_0001h
2F68h	Memory Region Descriptor (MRGD_W2_123)	32	RW	0000_0000h
2F6Ch	Memory Region Descriptor (MRGD_W3_123)	32	RW	0000_0000h
2F80h	Memory Region Descriptor (MRGD_W0_124)	32	RW	0000_0001h
2F84h	Memory Region Descriptor (MRGD_W1_124)	32	RW	0000_0001h
2F88h	Memory Region Descriptor (MRGD_W2_124)	32	RW	0000_0000h
2F8Ch	Memory Region Descriptor (MRGD_W3_124)	32	RW	0000_0000h
2FA0h	Memory Region Descriptor (MRGD_W0_125)	32	RW	0000_0001h
2FA4h	Memory Region Descriptor (MRGD_W1_125)	32	RW	0000_0001h
2FA8h	Memory Region Descriptor (MRGD_W2_125)	32	RW	0000_0000h
2FACh	Memory Region Descriptor (MRGD_W3_125)	32	RW	0000_0000h
2FC0h	Memory Region Descriptor (MRGD_W0_126)	32	RW	0000_0001h
2FC4h	Memory Region Descriptor (MRGD_W1_126)	32	RW	0000_0001h
2FC8h	Memory Region Descriptor (MRGD_W2_126)	32	RW	0000_0000h
2FCCh	Memory Region Descriptor (MRGD_W3_126)	32	RW	0000_0000h
2FE0h	Memory Region Descriptor (MRGD_W0_127)	32	RW	0000_0001h
2FE4h	Memory Region Descriptor (MRGD_W1_127)	32	RW	0000_0001h
2FE8h	Memory Region Descriptor (MRGD_W2_127)	32	RW	0000_0000h
2FECh	Memory Region Descriptor (MRGD_W3_127)	32	RW	0000_0000h
3000h	Memory Region Descriptor (MRGD_W0_128)	32	RW	0000_0001h
3004h	Memory Region Descriptor (MRGD_W1_128)	32	RW	0000_0001h
3008h	Memory Region Descriptor (MRGD_W2_128)	32	RW	0000_0000h
300Ch	Memory Region Descriptor (MRGD_W3_128)	32	RW	0000_0000h
3020h	Memory Region Descriptor (MRGD_W0_129)	32	RW	0000_0001h
3024h	Memory Region Descriptor (MRGD_W1_129)	32	RW	0000_0001h
3028h	Memory Region Descriptor (MRGD_W2_129)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
302Ch	Memory Region Descriptor (MRGD_W3_129)	32	RW	0000_0000h
3040h	Memory Region Descriptor (MRGD_W0_130)	32	RW	0000_0001h
3044h	Memory Region Descriptor (MRGD_W1_130)	32	RW	0000_0001h
3048h	Memory Region Descriptor (MRGD_W2_130)	32	RW	0000_0000h
304Ch	Memory Region Descriptor (MRGD_W3_130)	32	RW	0000_0000h
3060h	Memory Region Descriptor (MRGD_W0_131)	32	RW	0000_0001h
3064h	Memory Region Descriptor (MRGD_W1_131)	32	RW	0000_0001h
3068h	Memory Region Descriptor (MRGD_W2_131)	32	RW	0000_0000h
306Ch	Memory Region Descriptor (MRGD_W3_131)	32	RW	0000_0000h
3080h	Memory Region Descriptor (MRGD_W0_132)	32	RW	0000_0001h
3084h	Memory Region Descriptor (MRGD_W1_132)	32	RW	0000_0001h
3088h	Memory Region Descriptor (MRGD_W2_132)	32	RW	0000_0000h
308Ch	Memory Region Descriptor (MRGD_W3_132)	32	RW	0000_0000h
30A0h	Memory Region Descriptor (MRGD_W0_133)	32	RW	0000_0001h
30A4h	Memory Region Descriptor (MRGD_W1_133)	32	RW	0000_0001h
30A8h	Memory Region Descriptor (MRGD_W2_133)	32	RW	0000_0000h
30ACh	Memory Region Descriptor (MRGD_W3_133)	32	RW	0000_0000h
30C0h	Memory Region Descriptor (MRGD_W0_134)	32	RW	0000_0001h
30C4h	Memory Region Descriptor (MRGD_W1_134)	32	RW	0000_0001h
30C8h	Memory Region Descriptor (MRGD_W2_134)	32	RW	0000_0000h
30CCh	Memory Region Descriptor (MRGD_W3_134)	32	RW	0000_0000h
30E0h	Memory Region Descriptor (MRGD_W0_135)	32	RW	0000_0001h
30E4h	Memory Region Descriptor (MRGD_W1_135)	32	RW	0000_0001h
30E8h	Memory Region Descriptor (MRGD_W2_135)	32	RW	0000_0000h
30ECh	Memory Region Descriptor (MRGD_W3_135)	32	RW	0000_0000h
3100h	Memory Region Descriptor (MRGD_W0_136)	32	RW	0000_0001h
3104h	Memory Region Descriptor (MRGD_W1_136)	32	RW	0000_0001h
3108h	Memory Region Descriptor (MRGD_W2_136)	32	RW	0000_0000h
310Ch	Memory Region Descriptor (MRGD_W3_136)	32	RW	0000_0000h
3120h	Memory Region Descriptor (MRGD_W0_137)	32	RW	0000_0001h
3124h	Memory Region Descriptor (MRGD_W1_137)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3128h	Memory Region Descriptor (MRGD_W2_137)	32	RW	0000_0000h
312Ch	Memory Region Descriptor (MRGD_W3_137)	32	RW	0000_0000h
3140h	Memory Region Descriptor (MRGD_W0_138)	32	RW	0000_0001h
3144h	Memory Region Descriptor (MRGD_W1_138)	32	RW	0000_0001h
3148h	Memory Region Descriptor (MRGD_W2_138)	32	RW	0000_0000h
314Ch	Memory Region Descriptor (MRGD_W3_138)	32	RW	0000_0000h
3160h	Memory Region Descriptor (MRGD_W0_139)	32	RW	0000_0001h
3164h	Memory Region Descriptor (MRGD_W1_139)	32	RW	0000_0001h
3168h	Memory Region Descriptor (MRGD_W2_139)	32	RW	0000_0000h
316Ch	Memory Region Descriptor (MRGD_W3_139)	32	RW	0000_0000h
3180h	Memory Region Descriptor (MRGD_W0_140)	32	RW	0000_0001h
3184h	Memory Region Descriptor (MRGD_W1_140)	32	RW	0000_0001h
3188h	Memory Region Descriptor (MRGD_W2_140)	32	RW	0000_0000h
318Ch	Memory Region Descriptor (MRGD_W3_140)	32	RW	0000_0000h
31A0h	Memory Region Descriptor (MRGD_W0_141)	32	RW	0000_0001h
31A4h	Memory Region Descriptor (MRGD_W1_141)	32	RW	0000_0001h
31A8h	Memory Region Descriptor (MRGD_W2_141)	32	RW	0000_0000h
31ACh	Memory Region Descriptor (MRGD_W3_141)	32	RW	0000_0000h
31C0h	Memory Region Descriptor (MRGD_W0_142)	32	RW	0000_0001h
31C4h	Memory Region Descriptor (MRGD_W1_142)	32	RW	0000_0001h
31C8h	Memory Region Descriptor (MRGD_W2_142)	32	RW	0000_0000h
31CCh	Memory Region Descriptor (MRGD_W3_142)	32	RW	0000_0000h
31E0h	Memory Region Descriptor (MRGD_W0_143)	32	RW	0000_0001h
31E4h	Memory Region Descriptor (MRGD_W1_143)	32	RW	0000_0001h
31E8h	Memory Region Descriptor (MRGD_W2_143)	32	RW	0000_0000h
31ECh	Memory Region Descriptor (MRGD_W3_143)	32	RW	0000_0000h
3200h	Memory Region Descriptor (MRGD_W0_144)	32	RW	0000_0001h
3204h	Memory Region Descriptor (MRGD_W1_144)	32	RW	0000_0001h
3208h	Memory Region Descriptor (MRGD_W2_144)	32	RW	0000_0000h
320Ch	Memory Region Descriptor (MRGD_W3_144)	32	RW	0000_0000h
3220h	Memory Region Descriptor (MRGD_W0_145)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3224h	Memory Region Descriptor (MRGD_W1_145)	32	RW	0000_0001h
3228h	Memory Region Descriptor (MRGD_W2_145)	32	RW	0000_0000h
322Ch	Memory Region Descriptor (MRGD_W3_145)	32	RW	0000_0000h
3240h	Memory Region Descriptor (MRGD_W0_146)	32	RW	0000_0001h
3244h	Memory Region Descriptor (MRGD_W1_146)	32	RW	0000_0001h
3248h	Memory Region Descriptor (MRGD_W2_146)	32	RW	0000_0000h
324Ch	Memory Region Descriptor (MRGD_W3_146)	32	RW	0000_0000h
3260h	Memory Region Descriptor (MRGD_W0_147)	32	RW	0000_0001h
3264h	Memory Region Descriptor (MRGD_W1_147)	32	RW	0000_0001h
3268h	Memory Region Descriptor (MRGD_W2_147)	32	RW	0000_0000h
326Ch	Memory Region Descriptor (MRGD_W3_147)	32	RW	0000_0000h
3280h	Memory Region Descriptor (MRGD_W0_148)	32	RW	0000_0001h
3284h	Memory Region Descriptor (MRGD_W1_148)	32	RW	0000_0001h
3288h	Memory Region Descriptor (MRGD_W2_148)	32	RW	0000_0000h
328Ch	Memory Region Descriptor (MRGD_W3_148)	32	RW	0000_0000h
32A0h	Memory Region Descriptor (MRGD_W0_149)	32	RW	0000_0001h
32A4h	Memory Region Descriptor (MRGD_W1_149)	32	RW	0000_0001h
32A8h	Memory Region Descriptor (MRGD_W2_149)	32	RW	0000_0000h
32ACh	Memory Region Descriptor (MRGD_W3_149)	32	RW	0000_0000h
32C0h	Memory Region Descriptor (MRGD_W0_150)	32	RW	0000_0001h
32C4h	Memory Region Descriptor (MRGD_W1_150)	32	RW	0000_0001h
32C8h	Memory Region Descriptor (MRGD_W2_150)	32	RW	0000_0000h
32CCh	Memory Region Descriptor (MRGD_W3_150)	32	RW	0000_0000h
32E0h	Memory Region Descriptor (MRGD_W0_151)	32	RW	0000_0001h
32E4h	Memory Region Descriptor (MRGD_W1_151)	32	RW	0000_0001h
32E8h	Memory Region Descriptor (MRGD_W2_151)	32	RW	0000_0000h
32ECh	Memory Region Descriptor (MRGD_W3_151)	32	RW	0000_0000h
3300h	Memory Region Descriptor (MRGD_W0_152)	32	RW	0000_0001h
3304h	Memory Region Descriptor (MRGD_W1_152)	32	RW	0000_0001h
3308h	Memory Region Descriptor (MRGD_W2_152)	32	RW	0000_0000h
330Ch	Memory Region Descriptor (MRGD_W3_152)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3320h	Memory Region Descriptor (MRGD_W0_153)	32	RW	0000_0001h
3324h	Memory Region Descriptor (MRGD_W1_153)	32	RW	0000_0001h
3328h	Memory Region Descriptor (MRGD_W2_153)	32	RW	0000_0000h
332Ch	Memory Region Descriptor (MRGD_W3_153)	32	RW	0000_0000h
3340h	Memory Region Descriptor (MRGD_W0_154)	32	RW	0000_0001h
3344h	Memory Region Descriptor (MRGD_W1_154)	32	RW	0000_0001h
3348h	Memory Region Descriptor (MRGD_W2_154)	32	RW	0000_0000h
334Ch	Memory Region Descriptor (MRGD_W3_154)	32	RW	0000_0000h
3360h	Memory Region Descriptor (MRGD_W0_155)	32	RW	0000_0001h
3364h	Memory Region Descriptor (MRGD_W1_155)	32	RW	0000_0001h
3368h	Memory Region Descriptor (MRGD_W2_155)	32	RW	0000_0000h
336Ch	Memory Region Descriptor (MRGD_W3_155)	32	RW	0000_0000h
3380h	Memory Region Descriptor (MRGD_W0_156)	32	RW	0000_0001h
3384h	Memory Region Descriptor (MRGD_W1_156)	32	RW	0000_0001h
3388h	Memory Region Descriptor (MRGD_W2_156)	32	RW	0000_0000h
338Ch	Memory Region Descriptor (MRGD_W3_156)	32	RW	0000_0000h
33A0h	Memory Region Descriptor (MRGD_W0_157)	32	RW	0000_0001h
33A4h	Memory Region Descriptor (MRGD_W1_157)	32	RW	0000_0001h
33A8h	Memory Region Descriptor (MRGD_W2_157)	32	RW	0000_0000h
33ACh	Memory Region Descriptor (MRGD_W3_157)	32	RW	0000_0000h
33C0h	Memory Region Descriptor (MRGD_W0_158)	32	RW	0000_0001h
33C4h	Memory Region Descriptor (MRGD_W1_158)	32	RW	0000_0001h
33C8h	Memory Region Descriptor (MRGD_W2_158)	32	RW	0000_0000h
33CCh	Memory Region Descriptor (MRGD_W3_158)	32	RW	0000_0000h
33E0h	Memory Region Descriptor (MRGD_W0_159)	32	RW	0000_0001h
33E4h	Memory Region Descriptor (MRGD_W1_159)	32	RW	0000_0001h
33E8h	Memory Region Descriptor (MRGD_W2_159)	32	RW	0000_0000h
33ECh	Memory Region Descriptor (MRGD_W3_159)	32	RW	0000_0000h
3400h	Memory Region Descriptor (MRGD_W0_160)	32	RW	0000_0001h
3404h	Memory Region Descriptor (MRGD_W1_160)	32	RW	0000_0001h
3408h	Memory Region Descriptor (MRGD_W2_160)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
340Ch	Memory Region Descriptor (MRGD_W3_160)	32	RW	0000_0000h
3420h	Memory Region Descriptor (MRGD_W0_161)	32	RW	0000_0001h
3424h	Memory Region Descriptor (MRGD_W1_161)	32	RW	0000_0001h
3428h	Memory Region Descriptor (MRGD_W2_161)	32	RW	0000_0000h
342Ch	Memory Region Descriptor (MRGD_W3_161)	32	RW	0000_0000h
3440h	Memory Region Descriptor (MRGD_W0_162)	32	RW	0000_0001h
3444h	Memory Region Descriptor (MRGD_W1_162)	32	RW	0000_0001h
3448h	Memory Region Descriptor (MRGD_W2_162)	32	RW	0000_0000h
344Ch	Memory Region Descriptor (MRGD_W3_162)	32	RW	0000_0000h
3460h	Memory Region Descriptor (MRGD_W0_163)	32	RW	0000_0001h
3464h	Memory Region Descriptor (MRGD_W1_163)	32	RW	0000_0001h
3468h	Memory Region Descriptor (MRGD_W2_163)	32	RW	0000_0000h
346Ch	Memory Region Descriptor (MRGD_W3_163)	32	RW	0000_0000h
3480h	Memory Region Descriptor (MRGD_W0_164)	32	RW	0000_0001h
3484h	Memory Region Descriptor (MRGD_W1_164)	32	RW	0000_0001h
3488h	Memory Region Descriptor (MRGD_W2_164)	32	RW	0000_0000h
348Ch	Memory Region Descriptor (MRGD_W3_164)	32	RW	0000_0000h
34A0h	Memory Region Descriptor (MRGD_W0_165)	32	RW	0000_0001h
34A4h	Memory Region Descriptor (MRGD_W1_165)	32	RW	0000_0001h
34A8h	Memory Region Descriptor (MRGD_W2_165)	32	RW	0000_0000h
34ACh	Memory Region Descriptor (MRGD_W3_165)	32	RW	0000_0000h
34C0h	Memory Region Descriptor (MRGD_W0_166)	32	RW	0000_0001h
34C4h	Memory Region Descriptor (MRGD_W1_166)	32	RW	0000_0001h
34C8h	Memory Region Descriptor (MRGD_W2_166)	32	RW	0000_0000h
34CCh	Memory Region Descriptor (MRGD_W3_166)	32	RW	0000_0000h
34E0h	Memory Region Descriptor (MRGD_W0_167)	32	RW	0000_0001h
34E4h	Memory Region Descriptor (MRGD_W1_167)	32	RW	0000_0001h
34E8h	Memory Region Descriptor (MRGD_W2_167)	32	RW	0000_0000h
34ECh	Memory Region Descriptor (MRGD_W3_167)	32	RW	0000_0000h
3500h	Memory Region Descriptor (MRGD_W0_168)	32	RW	0000_0001h
3504h	Memory Region Descriptor (MRGD_W1_168)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3508h	Memory Region Descriptor (MRGD_W2_168)	32	RW	0000_0000h
350Ch	Memory Region Descriptor (MRGD_W3_168)	32	RW	0000_0000h
3520h	Memory Region Descriptor (MRGD_W0_169)	32	RW	0000_0001h
3524h	Memory Region Descriptor (MRGD_W1_169)	32	RW	0000_0001h
3528h	Memory Region Descriptor (MRGD_W2_169)	32	RW	0000_0000h
352Ch	Memory Region Descriptor (MRGD_W3_169)	32	RW	0000_0000h
3540h	Memory Region Descriptor (MRGD_W0_170)	32	RW	0000_0001h
3544h	Memory Region Descriptor (MRGD_W1_170)	32	RW	0000_0001h
3548h	Memory Region Descriptor (MRGD_W2_170)	32	RW	0000_0000h
354Ch	Memory Region Descriptor (MRGD_W3_170)	32	RW	0000_0000h
3560h	Memory Region Descriptor (MRGD_W0_171)	32	RW	0000_0001h
3564h	Memory Region Descriptor (MRGD_W1_171)	32	RW	0000_0001h
3568h	Memory Region Descriptor (MRGD_W2_171)	32	RW	0000_0000h
356Ch	Memory Region Descriptor (MRGD_W3_171)	32	RW	0000_0000h
3580h	Memory Region Descriptor (MRGD_W0_172)	32	RW	0000_0001h
3584h	Memory Region Descriptor (MRGD_W1_172)	32	RW	0000_0001h
3588h	Memory Region Descriptor (MRGD_W2_172)	32	RW	0000_0000h
358Ch	Memory Region Descriptor (MRGD_W3_172)	32	RW	0000_0000h
35A0h	Memory Region Descriptor (MRGD_W0_173)	32	RW	0000_0001h
35A4h	Memory Region Descriptor (MRGD_W1_173)	32	RW	0000_0001h
35A8h	Memory Region Descriptor (MRGD_W2_173)	32	RW	0000_0000h
35ACh	Memory Region Descriptor (MRGD_W3_173)	32	RW	0000_0000h
35C0h	Memory Region Descriptor (MRGD_W0_174)	32	RW	0000_0001h
35C4h	Memory Region Descriptor (MRGD_W1_174)	32	RW	0000_0001h
35C8h	Memory Region Descriptor (MRGD_W2_174)	32	RW	0000_0000h
35CCh	Memory Region Descriptor (MRGD_W3_174)	32	RW	0000_0000h
35E0h	Memory Region Descriptor (MRGD_W0_175)	32	RW	0000_0001h
35E4h	Memory Region Descriptor (MRGD_W1_175)	32	RW	0000_0001h
35E8h	Memory Region Descriptor (MRGD_W2_175)	32	RW	0000_0000h
35ECh	Memory Region Descriptor (MRGD_W3_175)	32	RW	0000_0000h
3600h	Memory Region Descriptor (MRGD_W0_176)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3604h	Memory Region Descriptor (MRGD_W1_176)	32	RW	0000_0001h
3608h	Memory Region Descriptor (MRGD_W2_176)	32	RW	0000_0000h
360Ch	Memory Region Descriptor (MRGD_W3_176)	32	RW	0000_0000h
3620h	Memory Region Descriptor (MRGD_W0_177)	32	RW	0000_0001h
3624h	Memory Region Descriptor (MRGD_W1_177)	32	RW	0000_0001h
3628h	Memory Region Descriptor (MRGD_W2_177)	32	RW	0000_0000h
362Ch	Memory Region Descriptor (MRGD_W3_177)	32	RW	0000_0000h
3640h	Memory Region Descriptor (MRGD_W0_178)	32	RW	0000_0001h
3644h	Memory Region Descriptor (MRGD_W1_178)	32	RW	0000_0001h
3648h	Memory Region Descriptor (MRGD_W2_178)	32	RW	0000_0000h
364Ch	Memory Region Descriptor (MRGD_W3_178)	32	RW	0000_0000h
3660h	Memory Region Descriptor (MRGD_W0_179)	32	RW	0000_0001h
3664h	Memory Region Descriptor (MRGD_W1_179)	32	RW	0000_0001h
3668h	Memory Region Descriptor (MRGD_W2_179)	32	RW	0000_0000h
366Ch	Memory Region Descriptor (MRGD_W3_179)	32	RW	0000_0000h
3680h	Memory Region Descriptor (MRGD_W0_180)	32	RW	0000_0001h
3684h	Memory Region Descriptor (MRGD_W1_180)	32	RW	0000_0001h
3688h	Memory Region Descriptor (MRGD_W2_180)	32	RW	0000_0000h
368Ch	Memory Region Descriptor (MRGD_W3_180)	32	RW	0000_0000h
36A0h	Memory Region Descriptor (MRGD_W0_181)	32	RW	0000_0001h
36A4h	Memory Region Descriptor (MRGD_W1_181)	32	RW	0000_0001h
36A8h	Memory Region Descriptor (MRGD_W2_181)	32	RW	0000_0000h
36ACh	Memory Region Descriptor (MRGD_W3_181)	32	RW	0000_0000h
36C0h	Memory Region Descriptor (MRGD_W0_182)	32	RW	0000_0001h
36C4h	Memory Region Descriptor (MRGD_W1_182)	32	RW	0000_0001h
36C8h	Memory Region Descriptor (MRGD_W2_182)	32	RW	0000_0000h
36CCh	Memory Region Descriptor (MRGD_W3_182)	32	RW	0000_0000h
36E0h	Memory Region Descriptor (MRGD_W0_183)	32	RW	0000_0001h
36E4h	Memory Region Descriptor (MRGD_W1_183)	32	RW	0000_0001h
36E8h	Memory Region Descriptor (MRGD_W2_183)	32	RW	0000_0000h
36ECh	Memory Region Descriptor (MRGD_W3_183)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3700h	Memory Region Descriptor (MRGD_W0_184)	32	RW	0000_0001h
3704h	Memory Region Descriptor (MRGD_W1_184)	32	RW	0000_0001h
3708h	Memory Region Descriptor (MRGD_W2_184)	32	RW	0000_0000h
370Ch	Memory Region Descriptor (MRGD_W3_184)	32	RW	0000_0000h
3720h	Memory Region Descriptor (MRGD_W0_185)	32	RW	0000_0001h
3724h	Memory Region Descriptor (MRGD_W1_185)	32	RW	0000_0001h
3728h	Memory Region Descriptor (MRGD_W2_185)	32	RW	0000_0000h
372Ch	Memory Region Descriptor (MRGD_W3_185)	32	RW	0000_0000h
3740h	Memory Region Descriptor (MRGD_W0_186)	32	RW	0000_0001h
3744h	Memory Region Descriptor (MRGD_W1_186)	32	RW	0000_0001h
3748h	Memory Region Descriptor (MRGD_W2_186)	32	RW	0000_0000h
374Ch	Memory Region Descriptor (MRGD_W3_186)	32	RW	0000_0000h
3760h	Memory Region Descriptor (MRGD_W0_187)	32	RW	0000_0001h
3764h	Memory Region Descriptor (MRGD_W1_187)	32	RW	0000_0001h
3768h	Memory Region Descriptor (MRGD_W2_187)	32	RW	0000_0000h
376Ch	Memory Region Descriptor (MRGD_W3_187)	32	RW	0000_0000h
3800h	Memory Region Descriptor (MRGD_W0_192)	32	RW	0000_0001h
3804h	Memory Region Descriptor (MRGD_W1_192)	32	RW	0000_0001h
3808h	Memory Region Descriptor (MRGD_W2_192)	32	RW	0000_0000h
380Ch	Memory Region Descriptor (MRGD_W3_192)	32	RW	0000_0000h
3820h	Memory Region Descriptor (MRGD_W0_193)	32	RW	0000_0001h
3824h	Memory Region Descriptor (MRGD_W1_193)	32	RW	0000_0001h
3828h	Memory Region Descriptor (MRGD_W2_193)	32	RW	0000_0000h
382Ch	Memory Region Descriptor (MRGD_W3_193)	32	RW	0000_0000h
3840h	Memory Region Descriptor (MRGD_W0_194)	32	RW	0000_0001h
3844h	Memory Region Descriptor (MRGD_W1_194)	32	RW	0000_0001h
3848h	Memory Region Descriptor (MRGD_W2_194)	32	RW	0000_0000h
384Ch	Memory Region Descriptor (MRGD_W3_194)	32	RW	0000_0000h
3860h	Memory Region Descriptor (MRGD_W0_195)	32	RW	0000_0001h
3864h	Memory Region Descriptor (MRGD_W1_195)	32	RW	0000_0001h
3868h	Memory Region Descriptor (MRGD_W2_195)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
386Ch	Memory Region Descriptor (MRGD_W3_195)	32	RW	0000_0000h
3A00h	Memory Region Descriptor (MRGD_W0_208)	32	RW	0000_0001h
3A04h	Memory Region Descriptor (MRGD_W1_208)	32	RW	0000_0001h
3A08h	Memory Region Descriptor (MRGD_W2_208)	32	RW	0000_0000h
3A0Ch	Memory Region Descriptor (MRGD_W3_208)	32	RW	0000_0000h
3A20h	Memory Region Descriptor (MRGD_W0_209)	32	RW	0000_0001h
3A24h	Memory Region Descriptor (MRGD_W1_209)	32	RW	0000_0001h
3A28h	Memory Region Descriptor (MRGD_W2_209)	32	RW	0000_0000h
3A2Ch	Memory Region Descriptor (MRGD_W3_209)	32	RW	0000_0000h
3A40h	Memory Region Descriptor (MRGD_W0_210)	32	RW	0000_0001h
3A44h	Memory Region Descriptor (MRGD_W1_210)	32	RW	0000_0001h
3A48h	Memory Region Descriptor (MRGD_W2_210)	32	RW	0000_0000h
3A4Ch	Memory Region Descriptor (MRGD_W3_210)	32	RW	0000_0000h
3A60h	Memory Region Descriptor (MRGD_W0_211)	32	RW	0000_0001h
3A64h	Memory Region Descriptor (MRGD_W1_211)	32	RW	0000_0001h
3A68h	Memory Region Descriptor (MRGD_W2_211)	32	RW	0000_0000h
3A6Ch	Memory Region Descriptor (MRGD_W3_211)	32	RW	0000_0000h
3A80h	Memory Region Descriptor (MRGD_W0_212)	32	RW	0000_0001h
3A84h	Memory Region Descriptor (MRGD_W1_212)	32	RW	0000_0001h
3A88h	Memory Region Descriptor (MRGD_W2_212)	32	RW	0000_0000h
3A8Ch	Memory Region Descriptor (MRGD_W3_212)	32	RW	0000_0000h
3AA0h	Memory Region Descriptor (MRGD_W0_213)	32	RW	0000_0001h
3AA4h	Memory Region Descriptor (MRGD_W1_213)	32	RW	0000_0001h
3AA8h	Memory Region Descriptor (MRGD_W2_213)	32	RW	0000_0000h
3AACh	Memory Region Descriptor (MRGD_W3_213)	32	RW	0000_0000h
3AC0h	Memory Region Descriptor (MRGD_W0_214)	32	RW	0000_0001h
3AC4h	Memory Region Descriptor (MRGD_W1_214)	32	RW	0000_0001h
3AC8h	Memory Region Descriptor (MRGD_W2_214)	32	RW	0000_0000h
3ACCh	Memory Region Descriptor (MRGD_W3_214)	32	RW	0000_0000h
3AE0h	Memory Region Descriptor (MRGD_W0_215)	32	RW	0000_0001h
3AE4h	Memory Region Descriptor (MRGD_W1_215)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3AE8h	Memory Region Descriptor (MRGD_W2_215)	32	RW	0000_0000h
3AECh	Memory Region Descriptor (MRGD_W3_215)	32	RW	0000_0000h
3B00h	Memory Region Descriptor (MRGD_W0_216)	32	RW	0000_0001h
3B04h	Memory Region Descriptor (MRGD_W1_216)	32	RW	0000_0001h
3B08h	Memory Region Descriptor (MRGD_W2_216)	32	RW	0000_0000h
3B0Ch	Memory Region Descriptor (MRGD_W3_216)	32	RW	0000_0000h
3B20h	Memory Region Descriptor (MRGD_W0_217)	32	RW	0000_0001h
3B24h	Memory Region Descriptor (MRGD_W1_217)	32	RW	0000_0001h
3B28h	Memory Region Descriptor (MRGD_W2_217)	32	RW	0000_0000h
3B2Ch	Memory Region Descriptor (MRGD_W3_217)	32	RW	0000_0000h
3B40h	Memory Region Descriptor (MRGD_W0_218)	32	RW	0000_0001h
3B44h	Memory Region Descriptor (MRGD_W1_218)	32	RW	0000_0001h
3B48h	Memory Region Descriptor (MRGD_W2_218)	32	RW	0000_0000h
3B4Ch	Memory Region Descriptor (MRGD_W3_218)	32	RW	0000_0000h
3B60h	Memory Region Descriptor (MRGD_W0_219)	32	RW	0000_0001h
3B64h	Memory Region Descriptor (MRGD_W1_219)	32	RW	0000_0001h
3B68h	Memory Region Descriptor (MRGD_W2_219)	32	RW	0000_0000h
3B6Ch	Memory Region Descriptor (MRGD_W3_219)	32	RW	0000_0000h
3B80h	Memory Region Descriptor (MRGD_W0_220)	32	RW	0000_0001h
3B84h	Memory Region Descriptor (MRGD_W1_220)	32	RW	0000_0001h
3B88h	Memory Region Descriptor (MRGD_W2_220)	32	RW	0000_0000h
3B8Ch	Memory Region Descriptor (MRGD_W3_220)	32	RW	0000_0000h
3BA0h	Memory Region Descriptor (MRGD_W0_221)	32	RW	0000_0001h
3BA4h	Memory Region Descriptor (MRGD_W1_221)	32	RW	0000_0001h
3BA8h	Memory Region Descriptor (MRGD_W2_221)	32	RW	0000_0000h
3BACh	Memory Region Descriptor (MRGD_W3_221)	32	RW	0000_0000h
3BC0h	Memory Region Descriptor (MRGD_W0_222)	32	RW	0000_0001h
3BC4h	Memory Region Descriptor (MRGD_W1_222)	32	RW	0000_0001h
3BC8h	Memory Region Descriptor (MRGD_W2_222)	32	RW	0000_0000h
3BCCh	Memory Region Descriptor (MRGD_W3_222)	32	RW	0000_0000h
3BE0h	Memory Region Descriptor (MRGD_W0_223)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3BE4h	Memory Region Descriptor (MRGD_W1_223)	32	RW	0000_0001h
3BE8h	Memory Region Descriptor (MRGD_W2_223)	32	RW	0000_0000h
3BECh	Memory Region Descriptor (MRGD_W3_223)	32	RW	0000_0000h

15.4.1.2 Control (CR)

Offset

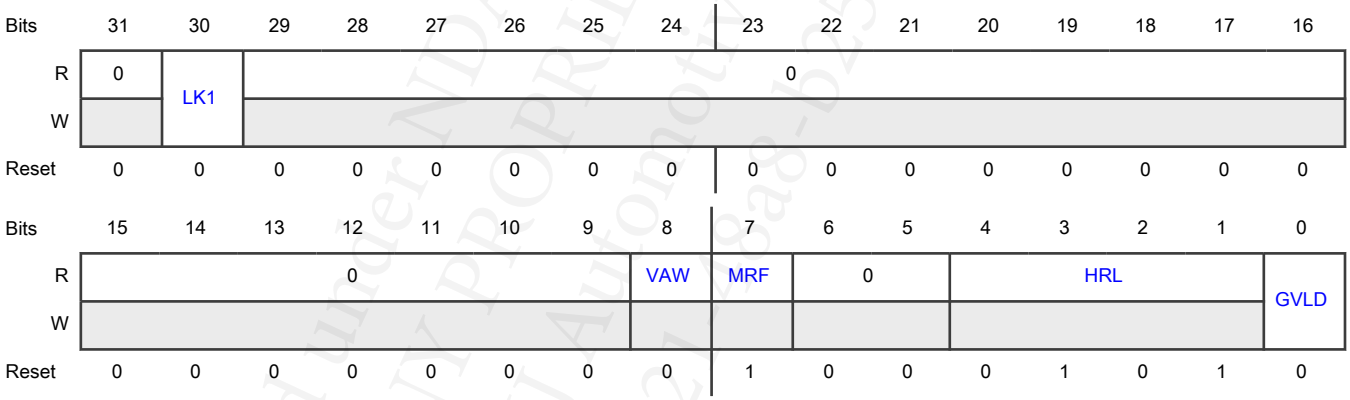
Register	Offset
CR	0h

Function

This register provides the read-only status of the XRDC and a global enable bit for the entire module's operation.

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31 —	Reserved
30 LK1	Lock This field provides a locking mechanism that can be used to prohibit the ability to write to this register. After you write 1 to this field, it remains asserted until the next reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Register can be written to by any secure privileged write 1b - Register is locked (read-only) until the next reset
29-9 —	Reserved
8 VAW	Virtualization Aware This field indicates that the domain assignments support the optional inclusion of a logical partition identifier (also known as an operating system number). 0b - Implementation is not virtualization-aware 1b - Implementation is virtualization-aware
7 MRF	Memory Region Format This field indicates that the format of the memory region descriptors and is defined by SoC requirements. 0b - Reserved 1b - SMPU family format
6-5 —	Reserved
4-1 HRL	Hardware Revision Level This field specifies XRDC's hardware and definition revision level. It can be read by software to determine the functional definition of the module. It is hardwired to the value 0101b as it represents an architectural descendent from previous {system} memory protection units ({S}MPU).
0 GVLD	Global Valid (XRDC Global Enable/Disable) 0b - XRDC disabled; all accesses from all bus masters to all slaves allowed 1b - XRDC enabled

15.4.1.3 Hardware Configuration 0 (HWCFG0)

Offset

Register	Offset
HWCFG0	F0h

Function

This read-only register contains information on XRDC's hardware configuration. Specifically, it defines the number of implemented domains and bus masters along with the number of instances of memory region controllers (MRCs) and peripheral access controllers (PACs). The register value at reset is chip-specific. Attempted writes are error-terminated.

Access: Secure privileged read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MID				NPAC				NMRC							
W																
Reset	0	0	0	1	0	0	1	1	0	0	0	0	1	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NMSTR								NDID							
W																
Reset	0	0	0	1	0	0	1	1	0	0	0	0	0	1	1	1

Fields

Field	Function
31-28 MID	Module ID
27-24 NPAC	Number Of PACs This field defines the number of PACs minus 1 in the chip. Therefore, you must add 1 to the field value to get the actual number of peripheral access controllers [1-4].
23-16 NMRC	Number Of MRCs This field defines the number of MRCs minus 1 in the chip. Therefore, you must add 1 to the field value to get the actual number of memory region controllers [1-16].
15-8 NMSTR	Number Of Bus Masters This field defines the number of bus masters minus 1 in the chip. Therefore, you must add 1 to the field value to get the actual number of bus masters [1-64].
7-0 NDID	Number Of Domains This field defines the number of domains minus 1 in the chip. Therefore, you must add 1 to the field value to get the actual number of domains [1-16].

15.4.1.4 Hardware Configuration 1 (HWCFG1)

Offset

Register	Offset
HWCFG1	F4h

Function

This register contains information on XRDC's hardware configuration. It provides a mechanism for software to determine its domain number by reading the register. See Domain error capture management for details on typical usage. Attempted writes are error-terminated.

Access: Secure Privileged Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												DID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	u[1]	u	u	u

1. Reset value is determined by current configuration of master accessing.

Fields

Field	Function
31-4 —	Reserved
3-0 DID	Domain Identifier Number This field provides the domain number [0-15] of the requesting bus master.

15.4.1.5 Hardware Configuration 2 (HWCFG2)

Offset

Register	Offset
HWCFG2	F8h

Function

This register contains information on XRDC's hardware configuration. It provides a bitmap that signals the presence of a process identifier (PID) register sourced from the given bus master. The HWCFG2 register is associated with bus masters [31-0]. Attempted writes are error-terminated.

Access: Secure privileged read

Protected under NDA only
COMPANY PROPRIETARY
Beijing C&J Automotive Technology Co., Ltd.
c8c462e5-821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PIDP31	PIDP30	PIDP29	PIDP28	PIDP27	PIDP26	PIDP25	PIDP24	PIDP23	PIDP22	PIDP21	PIDP20	PIDP19	PIDP18	PIDP17	PIDP16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PIDP15	PIDP14	PIDP13	PIDP12	PIDP11	PIDP10	PIDP9	PIDP8	PIDP7	PIDP6	PIDP5	PIDP4	PIDP3	PIDP2	PIDP1	PIDP0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PIDPn	Process Identifier Process identifier present from bus master *n*. This field provides a bitmap to signal that bus master *n* sources a process identifier register to the MDAC logic. 0b - Bus master *n* does not source a process identifier register; MDAC logic provides the needed PID for processor cores 1b - Bus master *i* sources a process identifier register to MDAC logic

15.4.1.6 Master Domain Assignment Configuration (MDACFG0 - MDACFG21)

Offset

Register	Offset
MDACFG0	100h
MDACFG1	101h
MDACFG2	102h
MDACFG3	103h
MDACFG4	104h
MDACFG5	105h
MDACFG6	106h
MDACFG7	107h
MDACFG8	108h
MDACFG9	109h
MDACFG10	10Ah

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MDACFG11	10Bh
MDACFG12	10Ch
MDACFG15	10Fh
MDACFG16	110h
MDACFG17	111h
MDACFG18	112h
MDACFG19	113h
MDACFG20	114h
MDACFG21	115h

Function

This register defines the number of implemented domain assignment registers for bus master *m*, where *m*+1 can specify bus masters from 1 to 64. These registers are organized as a byte-sized data array and can be read using 8-, 16- or 32-bit accesses. An all-zero value (NCM = 0, NMDAR = 0) indicates a non-existent bus master. Attempted writes are error-terminated.

Typically, processor bus masters include one or more domain assignment registers, while non-processor masters have a single domain assignment register.

Access: Secure privileged read

Diagram

Bits	7	6	5	4	3	2	1	0
R	NCM		0		NMDAR			
W								
Reset	See Register reset values.							

Register reset values

Register	Reset value
MDACFG0–MDACFG1	08h
MDACFG2–MDACFG7	81h
MDACFG8–MDACFG10	08h
MDACFG11–MDACFG15	81h
MDACFG13–MDACFG14	Register not supported
MDACFG16–MDACFG18	08h
MDACFG19–MDACFG21	81h

Licensed under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4921-4ea8-b251-3a6d3ab624c1

Fields

Field	Function
7 NCM	Non-CPU Master This field indicates that bus master *m* is a non-CPU master. It specifies that the format of the associated MDA_W*r*_*m* register defines a non-processor domain assignment. The value of the field is 0 for a non-existent bus master. 0b - Bus master is a processor 1b - Bus master is a non-processor
6-4 —	Reserved
3-0 NMDAR	Number Of Master Domain Assignment Registers For Bus Master *m* This field specifies the number of registers associated with the master domain assignment register for a given bus master. The value is limited to the range [0-8], where 0 indicates a non-existent bus master and non-zero values indicate the number of implemented registers associated with this MDA*m*.

15.4.1.7 Memory Region Configuration (MRCFG0 - MRCFG13)

Offset

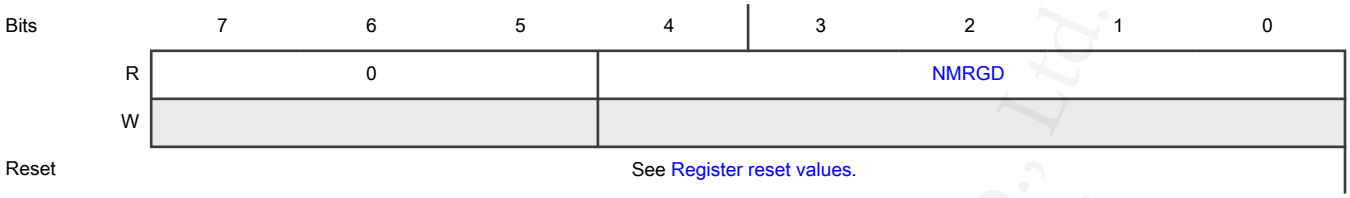
Register	Offset
MRCFG0	140h
MRCFG2	142h
MRCFG3	143h
MRCFG4	144h
MRCFG5	145h
MRCFG6	146h
MRCFG7	147h
MRCFG8	148h
MRCFG9	149h
MRCFG10	14Ah
MRCFG11	14Bh
MRCFG12	14Ch
MRCFG13	14Dh

Function

This read-only register defines the number of implemented memory region descriptors for each MRC*r*, where *r*+1 can specify up to 16 instances. These registers are organized as byte-sized data arrays and can be read using 8-, 16- or 32-bit accesses. A zero value indicates a non-existent memory region controller instance. Attempted writes are error-terminated.

Access: Secure Privileged Read

Diagram



Register reset values

Register	Reset value
MRCFG0–MRCFG10	10h
MRCFG1	Register not supported
MRCFG11	0Ch
MRCFG12	04h
MRCFG13	10h

Fields

Field	Function
7-5 —	Reserved
4-0 NMRGD	Number Of Memory Region Descriptors For Memory Region Controller *n* Number of memory region descriptors for MRC*r*. This field specifies the number of memory region descriptors associated with a given memory region controller instance. The value is limited to the range [0-16], where 0 indicates a non-existent MRC instance and non-zero values indicate the number of implemented memory region descriptors [4, 8,12,16] associated with the MRC*r* submodule.

15.4.1.8 Domain Error Location (DERRLOC0 - DERRLOC7)

Offset

For d = 0 to 7:

Register	Offset
DERRLOCd	200h + (d × 4h)

Function

This array of read-only registers provide the instance number of the submodule where an access violation occurred. These registers are organized as a word array, which is indexed by the faulting domain number, *d*. The two fields of this register provide a bitmap of instances associated with all submodules containing captured error information for that domain. These instance

numbers are then used as indices into the DERR_W0_*i*, DERR_W1_*i*, DERR_W2_*i*, and DERR_W3_*i* register arrays. See Domain error capture management for details.

When an access violation is detected by either a memory region controller (MRC) or a peripheral access controller (PAC), address and attribute information of the offending access is captured. Using the faulting DID number as the index, *d*, this array of read-only registers provides additional information signaling the instance number of the submodule where the access violation(s) occurred. Because the resulting exception handler needs the submodule instance to retrieve the captured address and attribute information from DERR_W0_*i* and DERR_W1_*i*, these registers provide the instance number details.

Attempted writes are error-terminated.

Access: Secure privileged read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												PACINST			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MRCINST															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19-16 PACINST	PAC Instance This field is a bitmap indicating the presence of a detected access violation for domain *d* in the implemented instances of the PAC. The least-significant bit of this field (bit 16) corresponds to PAC instance 0. The most-significant bit of this field (bit 19) corresponds to PAC instance 3, and so on. Multiple bits can be set at any time indicating access violations for this domain have been detected across multiple instances of the PACs. **NOTE** When a memory slot controller (MSC) is present, it appears here in the same way as a PAC instance. For each bit in this field: • 0 - The peripheral access controller has not detected an access violation or is not physically present. • 1 - The peripheral access controller has detected one or more access violations for this domain.
15-0 MRCINST	MRC Instance

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is a bitmap indicating the presence of a detected access violation for domain *d* in the implemented instances of the MRC. The least-significant bit of this field (bit 0) corresponds to MRC instance 0. The most-significant bit of this field (bit 3) corresponds to MRC instance 3, and so on. Multiple bits can be set at any time to indicate access violations for this domain have been detected across multiple instances of the MRCs. For each bit in this field: • 0 - The memory region controller has not detected an access violation or is not physically present. • 1 - The memory region controller has detected one or more access violations for this domain.

15.4.1.9 Domain Error Word0 (DERR_W0_0 - DERR_W0_19)

Offset

Register	Offset
DERR_W0_0	400h
DERR_W0_2	420h
DERR_W0_3	430h
DERR_W0_4	440h
DERR_W0_5	450h
DERR_W0_6	460h
DERR_W0_7	470h
DERR_W0_8	480h
DERR_W0_9	490h
DERR_W0_10	4A0h
DERR_W0_11	4B0h
DERR_W0_12	4C0h
DERR_W0_13	4D0h
DERR_W0_16	500h
DERR_W0_17	510h
DERR_W0_18	520h
DERR_W0_19	530h

Function

This read-only register array provides the address of an access violation detected by either an MRC or a PAC. These registers are organized as a word array, which is indexed by the violating submodule instance number. That is, the index, *i*, of this array is the instance number of the submodule that detected the access violation. The submodule instance numbers are provided by the DERRLOC registers. The memory-mapped error capture detail registers are organized as 20 sequential 16-byte entries. The

first 16 register sets are associated with the MRCs. The following four register sets are associated with the PACs. Each 16-byte structure contains DERR_W0_*i*, DERR_W1_*i*, DERR_W2_*i*, and DERR_W3_*i*.

The error capture registers in the MRC and PAC submodules contain physical registers for each domain, but are organized in the DERR_W0_*i* and DERR_W1_*i* registers to provide the information for the requesting domain only. Thus, the access violation exception handler for each domain only has visibility into the captured error information for its own domain. See Domain error capture management for details.

When an access violation is detected and the offending information captured, subsequent updates to this register are disabled until the required data pattern is written to the DERR_W3_*i* register. At that time, this register is cleared and re-enabled to capture the next access violation.

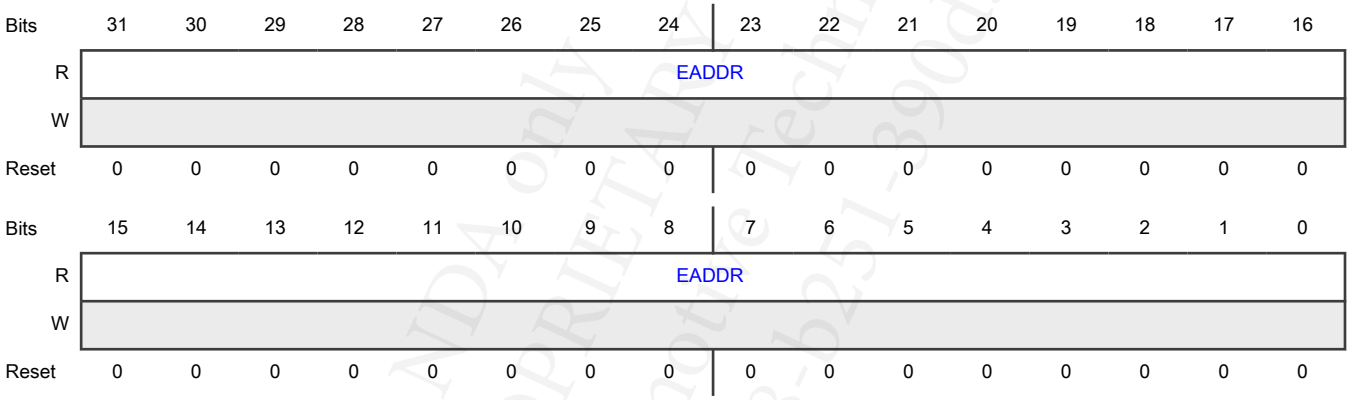
Attempted writes are error-terminated as are attempted reads of an MRC or PAC instance that is not physically present.

NOTE

If masters with the same DID cause simultaneous error accesses, DERR_W0_*i* records only the error of the lowest slave index.

Access: Secure privileged read

Diagram



Fields

Field	Function
31-0 EADDR	Error Address This is the access address that generated an access violation.

15.4.1.10 Domain Error Word1 (DERR_W1_0 - DERR_W1_19)

Offset

Register	Offset
DERR_W1_0	404h
DERR_W1_2	424h
DERR_W1_3	434h

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624d1

Table continued from the previous page...

Register	Offset
DERR_W1_4	444h
DERR_W1_5	454h
DERR_W1_6	464h
DERR_W1_7	474h
DERR_W1_8	484h
DERR_W1_9	494h
DERR_W1_10	4A4h
DERR_W1_11	4B4h
DERR_W1_12	4C4h
DERR_W1_13	4D4h
DERR_W1_16	504h
DERR_W1_17	514h
DERR_W1_18	524h
DERR_W1_19	534h

Function

This read-only register array provides the attributes of an access violation detected by either an MRC or a PAC. These registers are organized as a word array, which is indexed by the violating submodule instance number. See Domain Error Location (DERRLOC0 - DERRLOC7), Domain Error Word0 (DERR_W0_0 - DERR_W0_19), and Domain error capture management for more information.

NOTE

If masters with the same DID cause simultaneous error accesses, DERR_W1_*i*records only the error of the lowest slave index.

Access: Secure privileged read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EST		0			EPORT			0							EA40FMT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				ERW	EATR			0				EDID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 EST	Error State This field signals the state of access violations for this domain in this instance of the memory region controller or peripheral access controller. After an access violation has been detected and the faulting address and attribute information stored, subsequent errors are recorded as an overrun condition without any data captured. After retrieving the faulting address and attributes, the error capture mechanism must be rearmed by performing a write to DERR_W3_*i*. 00-01b - No access violation detected 10b - A single access violation was detected 11b - Multiple access violations for this domain are detected by this submodule instance. Only the address and attribute information for the first error are captured in DERR_W0_*i* and DERR_W1_*i*.
29-27 —	Reserved
26-24 EPORT	Error Port This field identifies the encoded port number of the MRC that detected the access violation. The MRC port number connection is chip-specific. See the chip configuration details for more information. For access violations detected by the PAC, this field is cleared.
23-17 —	Reserved
16 EA40FMT	Error Address 40 Format Domain Error Word2 (DERR_W2_0 - DERR_W2_19) contains EADDR[39:32] 0b - 32-bit format 1b - 40-bit format
15-12 —	Reserved
11 ERW	Error Read/Write This field signals whether the captured access violation occurred on a read or write reference. 0b - Read access 1b - Write access
10-8 EATR	Error Attributes This field captures certain attributes of the access violation 000b - Secure user mode, instruction fetch access 001b - Secure user mode, data access

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010b - Secure privileged mode, instruction fetch access 011b - Secure privileged mode, data access 100b - Nonsecure user mode, instruction fetch access 101b - Nonsecure user mode, data access 110b - Nonsecure privileged mode, instruction fetch access 111b - Nonsecure privileged mode, data access
7-4 —	Reserved
3-0 EDID	Error Domain Identifier This field captures the domain identifier of the access violation.

15.4.1.11 Domain Error Word2 (DERR_W2_0 - DERR_W2_19)

Offset

Register	Offset
DERR_W2_0	408h
DERR_W2_2	428h
DERR_W2_3	438h
DERR_W2_4	448h
DERR_W2_5	458h
DERR_W2_6	468h
DERR_W2_7	478h
DERR_W2_8	488h
DERR_W2_9	498h
DERR_W2_10	4A8h
DERR_W2_11	4B8h
DERR_W2_12	4C8h
DERR_W2_13	4D8h
DERR_W2_16	508h
DERR_W2_17	518h
DERR_W2_18	528h
DERR_W2_19	538h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Function

This is a word-sized location in the domain error details data structure. The registers are organized in a data structure based on instance number, that is, n = the instance number of the submodule which detected the access violation, and are read using information directly extracted from the DERRLOCn registers. The memory-mapped capture error detail registers are organized as 20 sequential 16-byte entries. The first 16 register sets are associated with the memory region controllers. The following four registers sets are associated with the peripheral access controllers. Each 16-byte record contains DERR_W0_*i*, DERR_W1_*i*, DERR_W2_*i* and DERR_W3_*i*. See Domain error capture management for more details. A read of this location returns zeroes. Attempted writes are error-terminated as are attempted reads of an MRC or PAC instance that is not physically present.

NOTE

If masters with the same DID cause simultaneous error accesses, DERR_W2_*i* will record only the error of the lowest slave index.

Access: Secure Privileged Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								EADDR39_32							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 EADDR39_32	EADDR[39:32]

15.4.1.12 Domain Error Word3 (DERR_W3_0 - DERR_W3_19)

Offset

Register	Offset
DERR_W3_0	40Ch
DERR_W3_2	42Ch
DERR_W3_3	43Ch

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,
c8c462e5-4821-48a8-b251-590a3b624c1

Table continued from the previous page...

Register	Offset
DERR_W3_4	44Ch
DERR_W3_5	45Ch
DERR_W3_6	46Ch
DERR_W3_7	47Ch
DERR_W3_8	48Ch
DERR_W3_9	49Ch
DERR_W3_10	4ACh
DERR_W3_11	4BCh
DERR_W3_12	4CCh
DERR_W3_13	4DCh
DERR_W3_16	50Ch
DERR_W3_17	51Ch
DERR_W3_18	52Ch
DERR_W3_19	53Ch

Function

This register rearms the error capture logic and clears the DERR_W0_*i* and DERR_W1_*i* registers. After the domain access violation error details are read, typically in an exception service routine, a 32-bit word write to this register is required to rearm the error capture logic.

It also clears the corresponding field in DERRLOC*n* register.

A read of this location returns a value of 0000h. Attempted reads of an MRC or PAC instance that is not physically present are error-terminated.

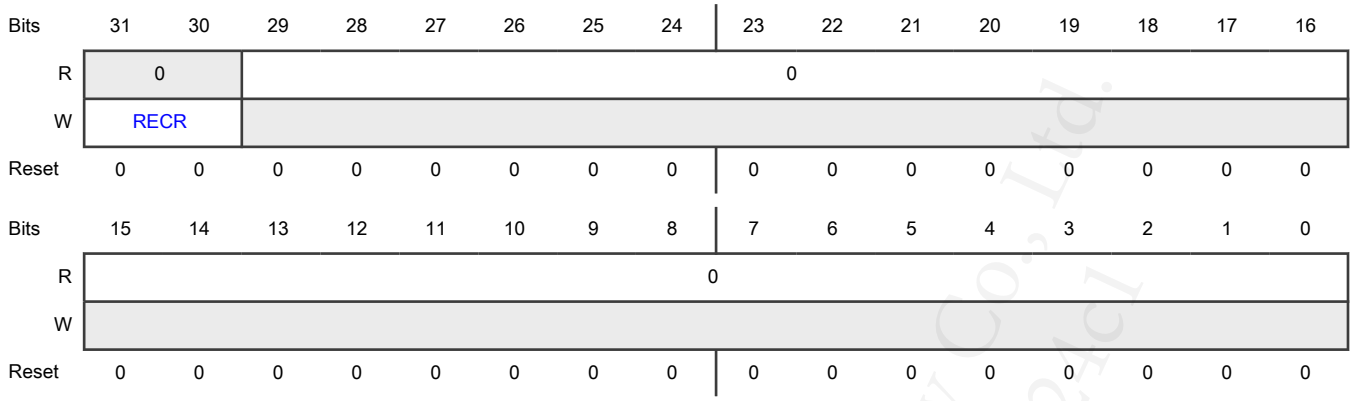
See Domain error capture management for details.

NOTE

If masters with the same DID cause simultaneous error accesses, DERR_W3_*i* records only the error of the lowest slave index.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4827-4888-b51f-390d3ab624c1

Diagram



Fields

Field	Function
31-30 RECR	Rearm Error Capture Registers This field controls the rearming of the domain error capture registers. After an access violation is detected with the faulting address and attribute information stored, subsequent errors are recorded as an overrun condition without any data captured. 00, 10, 11b - Writing this value has no effect 01b - Writing this value rearms the error capture mechanism and clears registers DERR_W0_*i* and DERR_W1_*i*
29-0 —	Reserved

15.4.1.13 Process Identifier (PID0 - PID18)

Offset

Register	Offset
PID0	700h
PID1	704h
PID8	720h
PID9	724h
PID10	728h
PID16	740h
PID17	744h
PID18	748h

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca462e5-4821-48a8-b25-39d8ab624c1

Function

In XRDC's access control definition, each processor has a corresponding process identifier (PID) that performs two important functions: the PID[5] bit defines the secure/nonsecure attribute associated with an executing task, and the entire value can be used to group tasks into different domains.

While certain processors include this register in their programming model definitions, others do not. This register is provided for those processors that do not include a PID register in their programming models. Secure privileged software saves and restores the PID as part of any context switch.

This data structure defines an array of 32-bit values, one per MDA module, that define the PID. Because this register resource is only applicable to processor cores, the data structure is typically sparsely populated. The HWCFG2 register provides a bitmap of the implemented PID*n* registers. This data structure is indexed using the corresponding MDA instance number.

For processors only supporting the 3-state access control model (SecurePriv, SecureUser, NonsecureUser), the nonsecure[n] output signal from the MDAC submodule is forced to 0 while being in privileged mode to enable precise state transitions between user and privileged modes. Specifically, the MDAC logic for bus master *n* forms the nonsecure attribute output signal as a function of two configuration bits: PID*m*[TSM] and HWCFG{2,3}[*n*]. See the following table that provides details on generation of the nonsecure_out[*n*] attribute.

Depending on the operating clock domain of each MDAC instance, there may be optional information stored in the corresponding PID*m* register to properly implement the LK2 = 2 functionality.

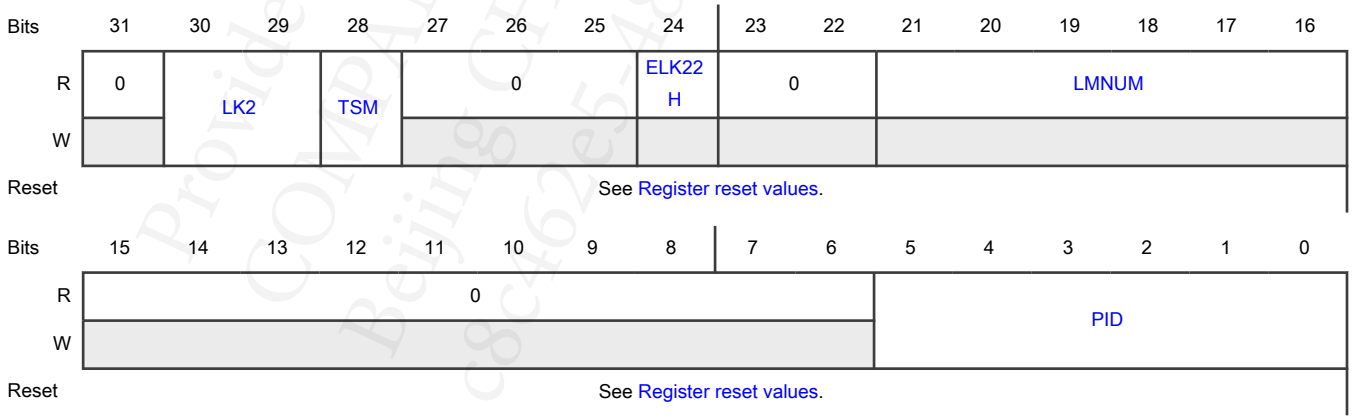
Specifically, if PIDm[ELK22H] = 1, then the MDAC hardware loads PIDm[LMNUM] with the bus master number of the processor that set PIDm[LK2] = 2. The LMNUM field is subsequently used to qualify attempted writes to the PIDm register. The LMNUM field is cleared when LK2 = [0,1,3].

Reads of the PID*n* register either return the contents of this register, or the PID*n* value directly sourced from a processor. For non-processor bus masters, this register does not exist and any attempted read is error-terminated.

Table 54. Generation of the nonsecure_out[*n*] attribute

Configuration {PID*n*[TSM], HWCFG{2,3}[*n*]}	local_nonsecure[*n*]	nonsecure_out[*n*]
00b	nonsecure_in[*n*]	local_nonsecure[*n*]
01b	nonsecure_in[*n*]	local_nonsecure[*n*] & ~priv_in[*n*]
10b	PID*n*[5]	local_nonsecure[*n*] & ~priv_in[*n*]
11b	pid_in[*n*][5]	local_nonsecure[*n*] & ~priv_in[*n*]

Diagram



Register reset values

Register	Reset value
PID0–PID1	0100_0000h
PID2–PID7	Register not supported
PID8–PID18	0000_0000h
PID11–PID15	Register not supported

Fields

Field	Function
31 — 	Reserved
30-29 LK2 	Lock This field provides a locking mechanism that can be used to limit the ability to write to the register. After you write 1 to the bits of this field, they individually remain asserted until the next reset. If the processor sources its PID*m* directly, that is, HWCFG{2-3}[PIDP*m* = 1], then secure privileged reads of this memory location return 0 for this field. 00, 01b - Register can be written to by any secure privileged write 10b - Register can only be written by a secure privileged write from bus master m 11b - Register locked (read-only) until the next reset
28 TSM 	Three-State Model If asserted, this field indicates that the associated processor core only supports the three-state access control model. This indicator is sticky; that is, after it is set to 1, it remains at that value until the next reset. For processors only supporting the three-state access control model, this field must be set before loading any nonsecure value into the PID. See table: Generation of the nonsecure_out[*n*] attribute for the specifications of the privileged and nonsecure attributes generated by MDAC.
27-25 — 	Reserved
24 ELK22H 	Enable (LK2 = 2) Special Handling 0b - LK2 is in its standard form. The LMNUM field of this register is reserved and always read as 0. 1b - This read-only field signals an MDAC instance clocking configuration where the special handling of the LK2 = 2 state is required and LMNUM will capture the current lock owner.
23-22 — 	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21-16 LMNUM	Locked Master Number This field is only enabled when ELK22K = 1. It is written by the MDAC instance with the bus master number of the locking processor that set LK2 = 2 and is used to qualify subsequent attempted writes to the register. The field is cleared to 0 when LK2 = [0,1,3] or ELK22H = 0.
15-6 —	Reserved
5-0 PID	Process Identifier This field determines the secure (0) or nonsecure (1) attribute for transactions associated with the corresponding processor. If the processor sources its PID*n* directly, then secure privileged reads of this memory location return the processor register for this field. See Cache coherency interconnect (CCI) for more information.

15.4.1.14 Master Domain Assignment (MDA_W0_0_DFMT0 - MDA_W7_18_DFMT0)

Offset

Register	Offset
MDA_W0_0_DFMT0	800h
MDA_W1_0_DFMT0	804h
MDA_W2_0_DFMT0	808h
MDA_W3_0_DFMT0	80Ch
MDA_W4_0_DFMT0	810h
MDA_W5_0_DFMT0	814h
MDA_W6_0_DFMT0	818h
MDA_W7_0_DFMT0	81Ch
MDA_W0_1_DFMT0	820h
MDA_W1_1_DFMT0	824h
MDA_W2_1_DFMT0	828h
MDA_W3_1_DFMT0	82Ch
MDA_W4_1_DFMT0	830h
MDA_W5_1_DFMT0	834h
MDA_W6_1_DFMT0	838h
MDA_W7_1_DFMT0	83Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MDA_W0_8_DFMT0	900h
MDA_W1_8_DFMT0	904h
MDA_W2_8_DFMT0	908h
MDA_W3_8_DFMT0	90Ch
MDA_W4_8_DFMT0	910h
MDA_W5_8_DFMT0	914h
MDA_W6_8_DFMT0	918h
MDA_W7_8_DFMT0	91Ch
MDA_W0_9_DFMT0	920h
MDA_W1_9_DFMT0	924h
MDA_W2_9_DFMT0	928h
MDA_W3_9_DFMT0	92Ch
MDA_W4_9_DFMT0	930h
MDA_W5_9_DFMT0	934h
MDA_W6_9_DFMT0	938h
MDA_W7_9_DFMT0	93Ch
MDA_W0_10_DFMT0	940h
MDA_W1_10_DFMT0	944h
MDA_W2_10_DFMT0	948h
MDA_W3_10_DFMT0	94Ch
MDA_W4_10_DFMT0	950h
MDA_W5_10_DFMT0	954h
MDA_W6_10_DFMT0	958h
MDA_W7_10_DFMT0	95Ch
MDA_W0_16_DFMT0	A00h
MDA_W1_16_DFMT0	A04h
MDA_W2_16_DFMT0	A08h
MDA_W3_16_DFMT0	A0Ch
MDA_W4_16_DFMT0	A10h
MDA_W5_16_DFMT0	A14h
MDA_W6_16_DFMT0	A18h
MDA_W7_16_DFMT0	A1Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MDA_W0_17_DFMT0	A20h
MDA_W1_17_DFMT0	A24h
MDA_W2_17_DFMT0	A28h
MDA_W3_17_DFMT0	A2Ch
MDA_W4_17_DFMT0	A30h
MDA_W5_17_DFMT0	A34h
MDA_W6_17_DFMT0	A38h
MDA_W7_17_DFMT0	A3Ch
MDA_W0_18_DFMT0	A40h
MDA_W1_18_DFMT0	A44h
MDA_W2_18_DFMT0	A48h
MDA_W3_18_DFMT0	A4Ch
MDA_W4_18_DFMT0	A50h
MDA_W5_18_DFMT0	A54h
MDA_W6_18_DFMT0	A58h
MDA_W7_18_DFMT0	A5Ch

Function

This set of registers provide a two-dimensional data structure for assigning bus masters to domains. The number of implemented registers is defined by MDACFG*m*[NMDAR]. This per-master domain assignment is then repeated for each bus master (MDA*m*). Thus, *m* specifies the master number and *r* refers to the specific MDA register for a given bus master.

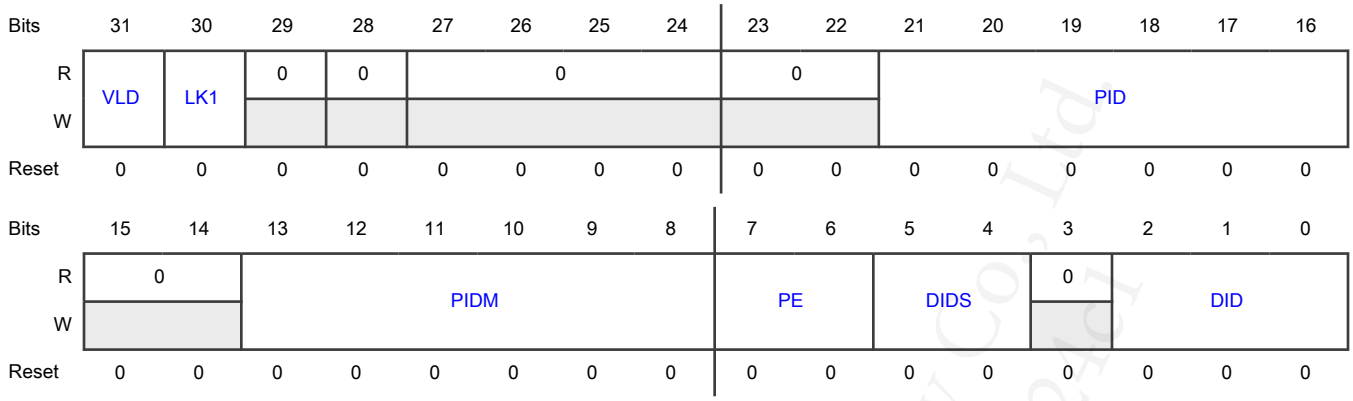
Each W*r* within the MDA*m* structure is a word-sized definition; there are two formats supported: one for processor cores and the other for non-processors. Processor masters typically support one or more W*r* domain definitions, while non-processor masters support a single W*r*.

The MDAC submodule is responsible for the generation of domain identifiers for every transaction from every bus master. If there is a single W*r* for a given master, then the specified domain identifier is used directly. If there are multiple W*r* values for a given master, then the MDAC evaluates the conditional terms to determine a “hit.” For all W*r* hits, their corresponding domain identifiers are logically summed together (using boolean OR). Use cases are typically expected to hit a single W*r* for a processor master. Special care is needed if none of the conditional terms hit any W*r* evaluation; for this case, the generated DID = 0 and software needs to be aware of any potential access rights granted for this DID.

Each MDA_W*r*_*m* register has one of two programming models depending on the state of the domain format field, DFMT, in this register. The model described in this section is applicable if DFMT = 0. This definition allows three different specifications of the DID for processors. The DFMT = 1 model is described in Master Domain Assignment (MDA_W*r*_*m*_DFMT1).

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31 VLD	Valid This field indicates that the domain assignment is valid. It is further qualified if the value of CR[GVLD] is 1. If the value of CR[GVLD] is 0, the DID output is defined by the chip-specific default DID value. See the XRDC chip configuration section for details. If both MDA_W*r_m*[VLD] and CR[GVLD] are asserted, the DID output is defined by the remaining contents of this register. 0b - W*r* domain assignment invalid 1b - W*r* domain assignment valid
30 LK1	Lock This field provides a locking mechanism that can be used to limit the ability to write to the register. After this field is set to 1, it remains at that value until the next reset. 0b - Register can be written to by any secure privileged write 1b - Register locked (read-only) until the next reset
29 DFMT	Domain Format Identifies this register's domain assignment. 0b - Processor-core domain assignment
28 —	Reserved
27-24 —	Reserved
23-22 —	Reserved
21-16	Process Identifier

Table continues on the next page...

Table continued from the previous page...

Field	Function
PID	This field specifies that the process identifier is to be combined with the PIDM field and included in the domain hit determination. The optional inclusion of the PID and PIDM is controlled by the MDA_W*r_m*[PE] field.
15-14 —	Reserved Always write 00b to this field.
13-8 PIDM	Process Identifier Mask This field provides a masking capability so that multiple process identifiers can be included as part of the domain hit determination. If a bit in the PIDM is set, then the corresponding bit of the PID is ignored in the comparison. The optional inclusion of the PID and PIDM is controlled by the MDA_W*r_m*[PE] field.
7-6 PE	Process Identifier Enable This field controls the optional inclusion of the PID, qualified by PIDM, into the domain hit evaluation. It provides the ability to include inclusive or exclusive sets of masked PID values. 00-01b - No process identifier is included in the domain hit evaluation 10b - The process identifier is included in the domain hit evaluation as defined by the following expression: `partial_domain_hit = (PE == 10b) && ((PID & ~PIDM) == (PIDn[PID] & ~PIDM))` 11b - The process identifier is included in the domain hit evaluation as defined by the following expression: `partial_domain_hit = (PE == 11b) && ~((PID & ~PIDM) == (PIDn[PID] & ~PIDM))`
5-4 DIDS	DID Select This field selects the source of the domain identifier. 00b - Use the DID field of this register as the domain identifier 01b - Use the input DID as the domain identifier 10b - Use bits [3:2] of this register concatenated with the low-order 2 bits of the input DID (DID_in[1:0]) as the domain identifier 11b - Reserved
3 —	Reserved
2-0 DID	Domain Identifier Contains the value of DID to be generated

Provided under NDA to COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c46215-4821-4c8-b23-3a6ab4c4c1

15.4.1.15 Master Domain Assignment (MDA_W0_2_DFMT1 - MDA_W0_21_DFMT1)

Offset

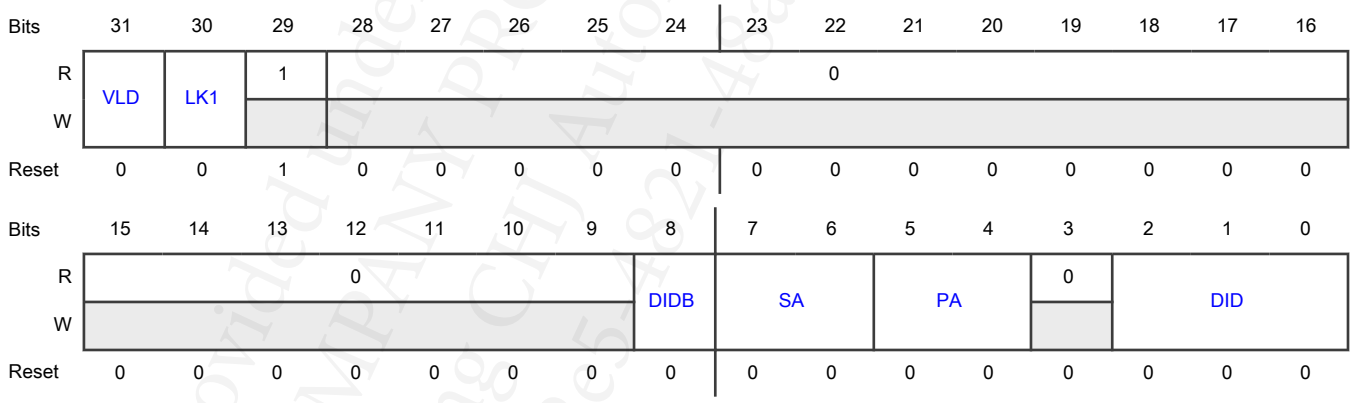
Register	Offset
MDA_W0_2_DFMT1	840h
MDA_W0_3_DFMT1	860h
MDA_W0_4_DFMT1	880h
MDA_W0_5_DFMT1	8A0h
MDA_W0_6_DFMT1	8C0h
MDA_W0_7_DFMT1	8E0h
MDA_W0_11_DFMT1	960h
MDA_W0_12_DFMT1	980h
MDA_W0_15_DFMT1	9E0h
MDA_W0_19_DFMT1	A60h
MDA_W0_20_DFMT1	A80h
MDA_W0_21_DFMT1	AA0h

Function

This register is identical to Master Domain Assignment (MDA_W*r_m*_DFMT0) except that the value of the domain format field, DFMT, is 1. This format supports two different specifications of DID for non-core bus masters.

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31	Valid

Table continues on the next page...

Table continued from the previous page...

Field	Function
VLD	This field indicates that the domain assignment is valid. It is further qualified if the value of CR[GVLD] is 1. If the value of CR[GVLD] is 0, the DID output is defined by the chip-specific default DID value. See the XRDC chip configuration section for details. If both MDA_W*r*_*m*[VLD] and CR[GVLD] are asserted, the DID output is defined by the remaining contents of this register. 0b - The W*r* domain assignment is invalid 1b - The W*r* domain assignment is valid
30 LK1	Lock This field provides a locking mechanism that can be used to limit the ability to write to the register. After this bit is set, it remains asserted until the next reset. 0b - Register can be written to by any secure privileged write 1b - Register is locked (read-only) until the next reset
29 DFMT	Domain Format Identifies this register's domain assignment. 1b - Non-processor domain assignment
28-9 —	Reserved
8 DIDB	DID Bypass If asserted, this field enables the bypassing of an input DID value as the domain identifier for this non-processor bus master. This capability allows non-processor bus masters (for example, a DMA) to masquerade as a processor. After this field is set to 1, it remains at that value until the next reset. 0b - Use MDA*n*[3:0] as the domain identifier 1b - Use the DID input as the domain identifier
7-6 SA	Secure Attribute This field defines the secure/nonsecure attribute for non-processor cores. **NOTE** The bus master's input secure/nonsecure attribute is used if SA = 1X, or this VLD = 0. 00b - Force the bus attribute for this master to secure. 01b - Force the bus attribute for this master to nonsecure. 1xb - Use the bus master's secure/nonsecure attribute directly.
5-4 PA	Privileged Attribute This field defines the privileged/user attribute for non-processor cores.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The bus master's input privileged or user attribute is used if PA = 1X, or this VLD = 0. 00b - Force the bus attribute for this master to user. 01b - Force the bus attribute for this master to privileged. 1xb - Use the bus master's privileged/user attribute directly.
3 —	Reserved
2-0 DID	Domain Identifier Contains the value of DID to be generated

15.4.1.16 Peripheral Domain Access Control (PDAC_W0_0 - PDAC_W0_408)

Offset

Register	Offset
PDAC_W0_(c * 128) + s	1000h + (c × 400h) + (s × 8h)

Function

This register array, PDAC_W*r*_*s*, provides a two-dimensional data structure for defining access control policies per domain for each implemented slave peripheral. In the offset equation above, *c* is the instance number and *s* is the slot number. See Table 55 for access control policies. There are two word-sized registers assigned for each peripheral address slot (W*r*).

Each W*r* within the PDAC*s* structure is a word-sized definition; a total of two words (64 bits) is needed to define the access control policies for all the domains.

These registers also allow a hardware semaphore to be optionally included in the access control evaluation for peripherals shared by multiple domains. If enabled, the state of the semaphore dynamically modifies the access control policies so that only the domain owning it has write permission into the peripheral. The write permissions for all other domains are revoked based on the semaphore state. If the semaphore is not owned by any domain, the D*x*ACP fields are evaluated normally.

The PAC submodule uses the peripheral access address to select the appropriate PDAC*s* register, and then uses the domain identifier associated with the access to select the appropriate D*x*ACP field. The selected D*x*ACP field is then evaluated against the attributes associated with the memory transaction to determine the validity of the access. See DxACP evaluation for details. If allowed, the access proceeds, else an access violation is signaled, the transfer error-terminated, and the appropriate address and attribute information captured in the corresponding error detail registers.

Access: Secure privileged read/write

Table 55. Access control policy

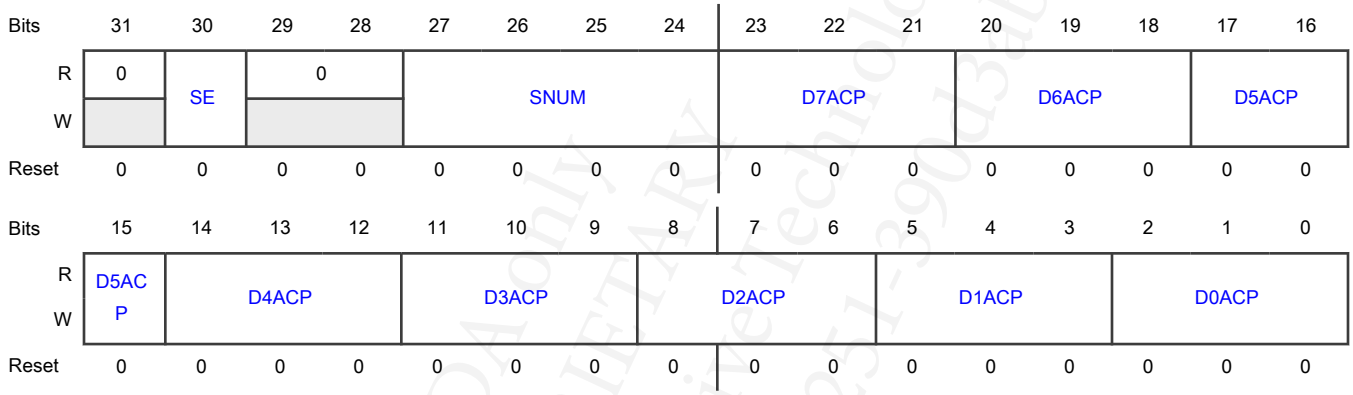
Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
111	r, w	r, w	r, w	r, w

Table continues on the next page...

Table 55. Access control policy (continued)

Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
110	r, w	r, w	r, w	none
101	r, w	r, w	r	r
100	r, w	r, w	r	none
011	r, w	r, w	none	none
010	r, w	none	none	none
001	r	r	none	none
000	none	none	none	none

Diagram



Fields

Field	Function
31 —	Reserved
30 SE	Semaphore Enable This field indicates if a semaphore, SNUM, is to be included in the D*x*ACP access evaluation. 0b - Do not include a semaphore in the D*x*ACP evaluation 1b - Include the semaphore defined by SNUM in the D*x*ACP evaluation
29-28 —	Reserved
27-24 SNUM	Semaphore Number Include this hardware semaphore in the D*x*ACP access evaluation.
23-21: D7ACP 20-18: D6ACP	Domain *n* Access Control Policy

Table continues on the next page...

Table continued from the previous page...

Field	Function
17-15: D5ACP 14-12: D4ACP 11-9: D3ACP 8-6: D2ACP 5-3: D1ACP 2-0: D0ACP	This field defines access control rights for the selected domain. The encodings specify read and write access capabilities based on the four operating states. These fields are only implemented for supported DIDs; unsupported DIDs are not implemented and default to a zero value, providing no access rights. See Table 55 for access control policies.

15.4.1.17 Peripheral Domain Access Control (PDAC_W1_0 - PDAC_W1_408)

Offset

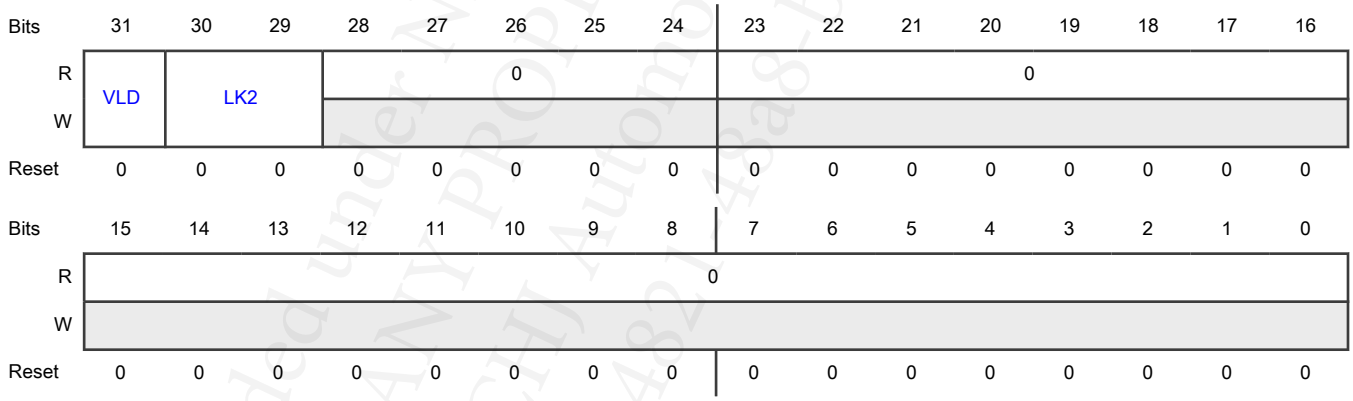
Register	Offset
PDAC_W1_(c * 128) + s	1004h + (c × 400h) + (s × 8h)

Function

See Peripheral Domain Access Control (PDAC_W0_0 - PDAC_W0_408) for description.

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31 VLD	Valid This field indicates that the peripheral domain access control definition is valid. It is further qualified if the value of CR[GVLD] is 1. If value of either CR[GVLD] or this field is 0, all accesses to the peripheral are

Table continues on the next page...

Table continued from the previous page...

Field	Function
	allowed. Any write to PDAC_W0_*s* clears the PDAC_W1_*s*[VLD] indicator, so a coherent register state can be supported. 0b - The PDAC*s* assignment is invalid. 1b - The PDAC*s* assignment is valid.
30-29 LK2	Lock This field provides a mechanism to limit writes to the PDAC registers to protect their contents. After you write 1 to the bits of this field, they individually remain asserted until the next reset. 00-01b - All PDAC*s* can be written to 10b - Domain *x* can only update the D*x*ACP field; no other PDAC fields can be written to 11b - PDACs are locked (read-only) until the next reset
28-24 —	Reserved
23-0 —	Reserved

15.4.1.18 Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_223)

Offset

Register	Offset
MRGD_W0_n * 16 + m	2000h + (n × 200h) + (m × 20h)

Function

This set of registers provide a two-dimensional data structure for defining access control policies (see Table 56) per domain for each supported memory region. There are four word-size registers (W*o*) defined in each memory region. There are up to 16 memory region controllers (MRCs), *n*, each supporting up to 16 memory region descriptors (MRGD), *m*. The number of MRCs is defined by HWCFG0[NMRC].

Each instance of an MRC submodule can monitor up to eight system buses concurrently.

NOTE

An AXI port counts as two system buses because the read address and write address channels operate independently.

Each W*o* within the MRGD*m* structure is a word-sized definition; a total of four words (128 bits, W0-W3) are needed to completely define the memory region descriptor.

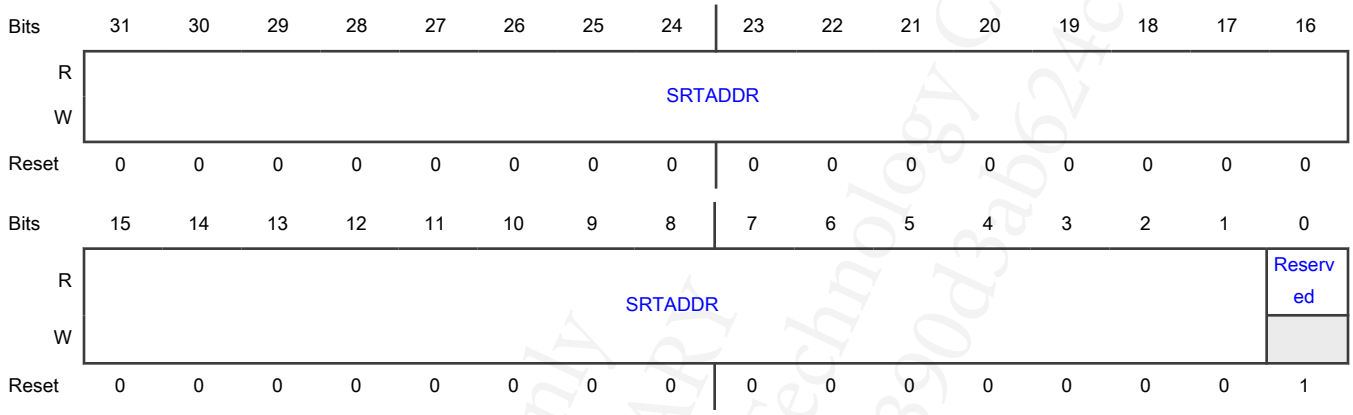
XRDC uses separate start (W0) and end addresses (W1) to define memory regions and leverages capabilities formerly provided in SMPUs.

The region descriptor also allows a hardware semaphore to be optionally included in the access control evaluation for memory regions shared by multiple domains. If enabled, the state of the semaphore dynamically modifies the access control policies so

that only the domain owning it has write permission into the region. The write permissions for all other domains are revoked based on the semaphore state. If the semaphore is not owned by any domain, the D*x*ACP fields are evaluated normally.

The MRC submodule evaluates multiple system bus access addresses versus the programmed address ranges to determine a region hit. After a region hit is determined, the MRC uses the domain identifier associated with the access to select the appropriate D*x*ACP field from the corresponding MRGD*n*. The selected D*x*ACP field is then evaluated against the attributes associated with the memory transaction to determine the validity of the access. See Memory region descriptor hit determination and Memory region DxACP evaluation for details. If allowed, the access proceeds, else an access violation is signaled, the transfer error terminated, and the appropriate address and attribute information captured in the corresponding error detail registers.

Diagram



Fields

Field	Function
31-1 SRTADDR	Start Address This field defines bits 35:5 of the 0-modulo-32-byte start address of the memory region. The minimum region size is 32 bytes. This field expects the address to be entered 4 bits shifted to the right.
0 —	Reserved

15.4.1.19 Memory Region Descriptor (MRGD_W1_0 - MRGD_W1_223)

Offset

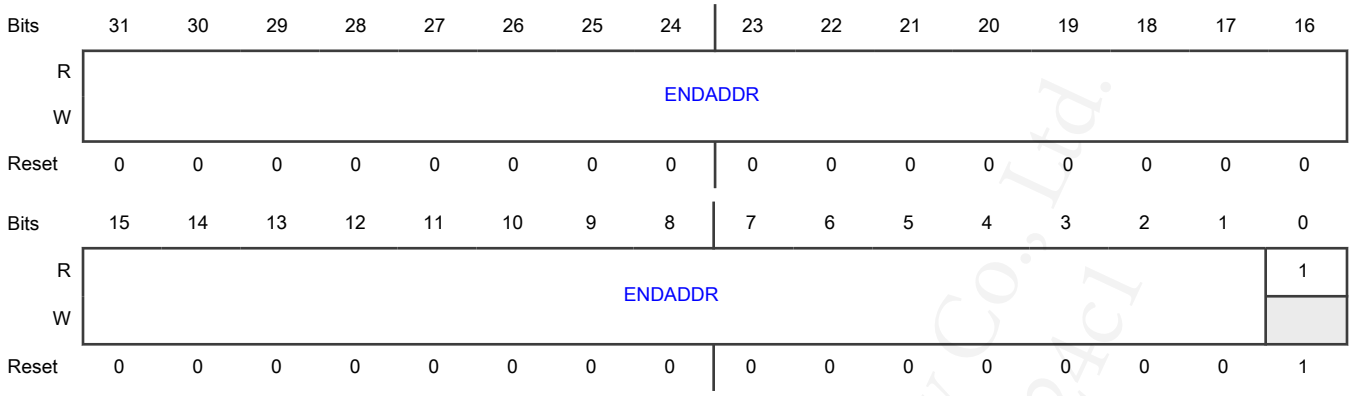
Register	Offset
MRGD_W1_n * 16 + m	2004h + (n × 200h) + (m × 20h)

Function

See Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_223) for description.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,
c84615e5-4891-4818-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-1 ENDADDR	End Address This field defines bits 35:5 of the 31-modulo-32-byte end address of the memory region. This field expects the address to be entered 4 bits shifted to the right.
0 —	Reserved

15.4.1.20 Memory Region Descriptor (MRGD_W2_0 - MRGD_W2_223)

Offset

Register	Offset
MRGD_W2_n * 16 + m	2008h + (n × 200h) + (m × 20h)

Function

See Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_223) for description.

Table 56. Access control policy

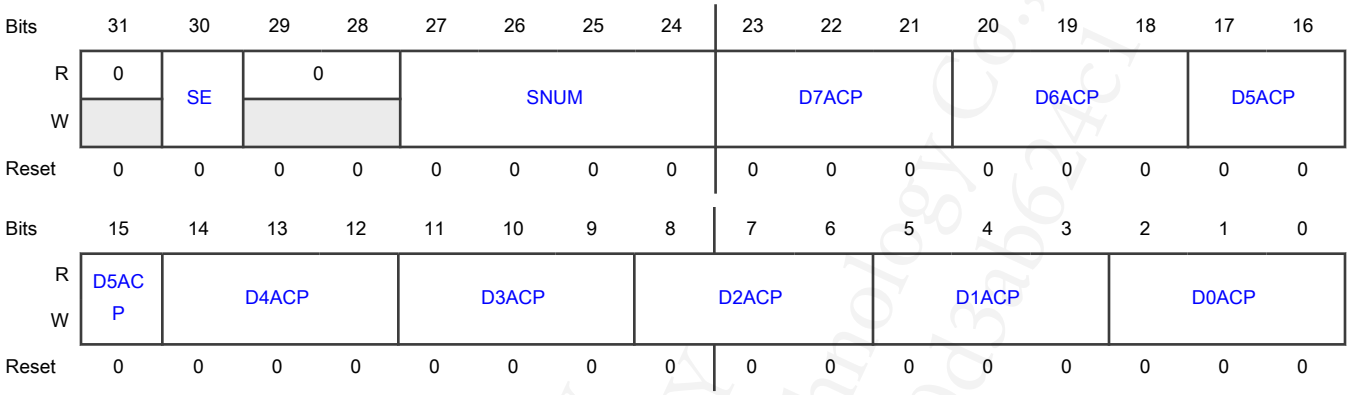
Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
111	r, w	r, w	r, w	r, w
110	r, w	r, w	r, w	none
101	r, w	r, w	r	r
100	r, w	r, w	r	none
011	r, w	r, w	none	none
010	r, w	none	none	none

Table continues on the next page...

Table 56. Access control policy (continued)

Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
001	r	r	none	none
000	none	none	none	none

Diagram



Fields

Field	Function
31 —	Reserved
30 SE	Semaphore Enable This field indicates if a semaphore, SNUM, is to be included in the D*x*ACP access evaluation. 0b - Do not include a semaphore in the D*x*ACP evaluation. 1b - Include the semaphore defined by SNUM in the D*x*ACP evaluation.
29-28 —	Reserved
27-24 SNUM	Semaphore Number Include this hardware semaphore in the D*x*ACP access evaluation.
23-21: D7ACP 20-18: D6ACP 17-15: D5ACP 14-12: D4ACP 11-9: D3ACP 8-6: D2ACP 5-3: D1ACP 2-0: D0ACP	Domain *n* Access Control Policy This field defines the access control rights for the selected domain. The encodings specify read and write access capabilities based on the four operating states. These fields are only implemented for supported DIDs; unsupported DIDs are not implemented and default to a zero value, providing no access rights. See the Access control policy table above.

15.4.1.21 Memory Region Descriptor (MRGD_W3_0 - MRGD_W3_223)

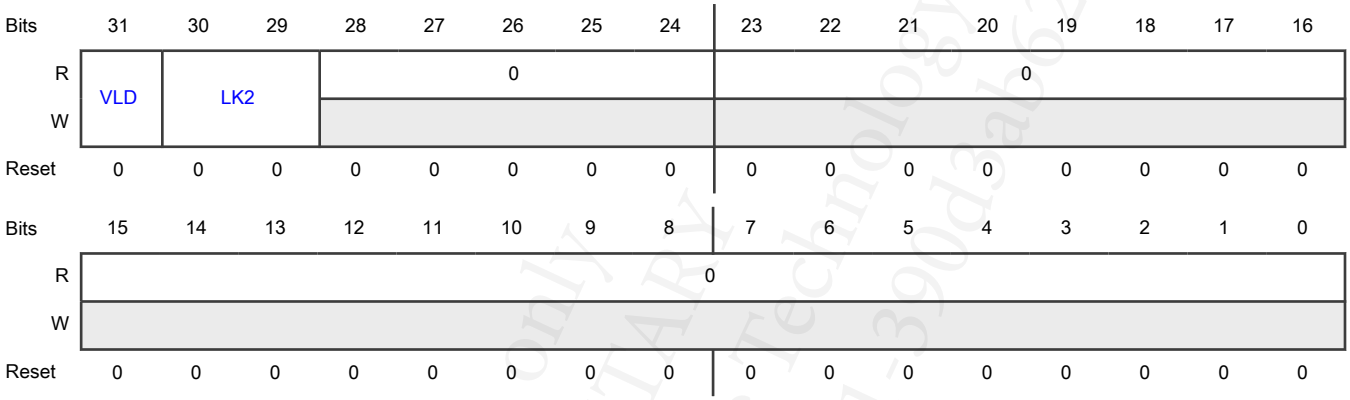
Offset

Register	Offset
MRGD_W3_n * 16 + m	200Ch + (n × 200h) + (m × 20h)

Function

See Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_223) for description.

Diagram



Fields

Field	Function
31 VLD	Valid This field indicates that the memory region descriptor definition is valid. It is further qualified if the value of CR[GVLD] is 1. If the value of CR[GVLD] is 0, then all accesses to the memory region are allowed. Any write to MRGD_W[0-2]_*n* clears the MRGD_W3_*n*[VLD] indicator, so a coherent register state can be supported. 0b - The MRGD*n* assignment is invalid. 1b - The MRGD*n* assignment is valid.
30-29 LK2	Lock This field provides a mechanism to limit writes to the MRGD*n* register to protect its contents. After you write 1 to the bits of this field, they individually remain asserted until the next reset. 00b - All MRGD*n* registers can be written to 01b - Reserved 10b - Domain *x* can only update the D*x*ACP field for domain *x*; no other ACP fields can be written to 11b - MRGD*n* registers are locked (read-only) until the next reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-24 —	Reserved
23-0 —	Reserved

15.4.2 XRDC_1 register descriptions

15.4.2.1 XRDC_1 memory map

XRDC_1 base address: 4400_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	0000_008Ah
F0h	Hardware Configuration 0 (HWCFG0)	32	RO	1005_0707h
F4h	Hardware Configuration 1 (HWCFG1)	32	RO	See description
F8h	Hardware Configuration 2 (HWCFG2)	32	RO	0000_0000h
100h	Master Domain Assignment Configuration (MDACFG0)	8	RO	81h
101h	Master Domain Assignment Configuration (MDACFG1)	8	RO	81h
102h	Master Domain Assignment Configuration (MDACFG2)	8	RO	81h
103h	Master Domain Assignment Configuration (MDACFG3)	8	RO	81h
104h	Master Domain Assignment Configuration (MDACFG4)	8	RO	81h
105h	Master Domain Assignment Configuration (MDACFG5)	8	RO	81h
106h	Master Domain Assignment Configuration (MDACFG6)	8	RO	81h
107h	Master Domain Assignment Configuration (MDACFG7)	8	RO	81h
140h	Memory Region Configuration (MRCFG0)	8	RO	04h
141h	Memory Region Configuration (MRCFG1)	8	RO	04h
142h	Memory Region Configuration (MRCFG2)	8	RO	10h
143h	Memory Region Configuration (MRCFG3)	8	RO	04h
144h	Memory Region Configuration (MRCFG4)	8	RO	04h
145h	Memory Region Configuration (MRCFG5)	8	RO	04h
200h - 21Ch	Domain Error Location (DERRLOC0 - DERRLOC7)	32	RO	0000_0000h
400h	Domain Error Word0 (DERR_W0_0)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
404h	Domain Error Word1 (DERR_W1_0)	32	RO	0000_0000h
408h	Domain Error Word2 (DERR_W2_0)	32	RO	0000_0000h
40Ch	Domain Error Word3 (DERR_W3_0)	32	WORZ	0000_0000h
410h	Domain Error Word0 (DERR_W0_1)	32	RO	0000_0000h
414h	Domain Error Word1 (DERR_W1_1)	32	RO	0000_0000h
418h	Domain Error Word2 (DERR_W2_1)	32	RO	0000_0000h
41Ch	Domain Error Word3 (DERR_W3_1)	32	WORZ	0000_0000h
420h	Domain Error Word0 (DERR_W0_2)	32	RO	0000_0000h
424h	Domain Error Word1 (DERR_W1_2)	32	RO	0000_0000h
428h	Domain Error Word2 (DERR_W2_2)	32	RO	0000_0000h
42Ch	Domain Error Word3 (DERR_W3_2)	32	WORZ	0000_0000h
430h	Domain Error Word0 (DERR_W0_3)	32	RO	0000_0000h
434h	Domain Error Word1 (DERR_W1_3)	32	RO	0000_0000h
438h	Domain Error Word2 (DERR_W2_3)	32	RO	0000_0000h
43Ch	Domain Error Word3 (DERR_W3_3)	32	WORZ	0000_0000h
440h	Domain Error Word0 (DERR_W0_4)	32	RO	0000_0000h
444h	Domain Error Word1 (DERR_W1_4)	32	RO	0000_0000h
448h	Domain Error Word2 (DERR_W2_4)	32	RO	0000_0000h
44Ch	Domain Error Word3 (DERR_W3_4)	32	WORZ	0000_0000h
450h	Domain Error Word0 (DERR_W0_5)	32	RO	0000_0000h
454h	Domain Error Word1 (DERR_W1_5)	32	RO	0000_0000h
458h	Domain Error Word2 (DERR_W2_5)	32	RO	0000_0000h
45Ch	Domain Error Word3 (DERR_W3_5)	32	WORZ	0000_0000h
500h	Domain Error Word0 (DERR_W0_16)	32	RO	0000_0000h
504h	Domain Error Word1 (DERR_W1_16)	32	RO	0000_0000h
508h	Domain Error Word2 (DERR_W2_16)	32	RO	0000_0000h
50Ch	Domain Error Word3 (DERR_W3_16)	32	WORZ	0000_0000h
800h	Master Domain Assignment (MDA_W0_0_DFMT1)	32	RW	2000_0000h
820h	Master Domain Assignment (MDA_W0_1_DFMT1)	32	RW	2000_0000h
840h	Master Domain Assignment (MDA_W0_2_DFMT1)	32	RW	2000_0000h
860h	Master Domain Assignment (MDA_W0_3_DFMT1)	32	RW	2000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
880h	Master Domain Assignment (MDA_W0_4_DFMT1)	32	RW	2000_0000h
8A0h	Master Domain Assignment (MDA_W0_5_DFMT1)	32	RW	2000_0000h
8C0h	Master Domain Assignment (MDA_W0_6_DFMT1)	32	RW	2000_0000h
8E0h	Master Domain Assignment (MDA_W0_7_DFMT1)	32	RW	2000_0000h
1000h	Peripheral Domain Access Control (PDAC_W0_0)	32	RW	0000_0000h
1004h	Peripheral Domain Access Control (PDAC_W1_0)	32	RW	0000_0000h
1008h	Peripheral Domain Access Control (PDAC_W0_1)	32	RW	0000_0000h
100Ch	Peripheral Domain Access Control (PDAC_W1_1)	32	RW	0000_0000h
1010h	Peripheral Domain Access Control (PDAC_W0_2)	32	RW	0000_0000h
1014h	Peripheral Domain Access Control (PDAC_W1_2)	32	RW	0000_0000h
1018h	Peripheral Domain Access Control (PDAC_W0_3)	32	RW	0000_0000h
101Ch	Peripheral Domain Access Control (PDAC_W1_3)	32	RW	0000_0000h
1020h	Peripheral Domain Access Control (PDAC_W0_4)	32	RW	0000_0000h
1024h	Peripheral Domain Access Control (PDAC_W1_4)	32	RW	0000_0000h
1028h	Peripheral Domain Access Control (PDAC_W0_5)	32	RW	0000_0000h
102Ch	Peripheral Domain Access Control (PDAC_W1_5)	32	RW	0000_0000h
1030h	Peripheral Domain Access Control (PDAC_W0_6)	32	RW	0000_0000h
1034h	Peripheral Domain Access Control (PDAC_W1_6)	32	RW	0000_0000h
1038h	Peripheral Domain Access Control (PDAC_W0_7)	32	RW	0000_0000h
103Ch	Peripheral Domain Access Control (PDAC_W1_7)	32	RW	0000_0000h
1040h	Peripheral Domain Access Control (PDAC_W0_8)	32	RW	0000_0000h
1044h	Peripheral Domain Access Control (PDAC_W1_8)	32	RW	0000_0000h
1048h	Peripheral Domain Access Control (PDAC_W0_9)	32	RW	0000_0000h
104Ch	Peripheral Domain Access Control (PDAC_W1_9)	32	RW	0000_0000h
1050h	Peripheral Domain Access Control (PDAC_W0_10)	32	RW	0000_0000h
1054h	Peripheral Domain Access Control (PDAC_W1_10)	32	RW	0000_0000h
1058h	Peripheral Domain Access Control (PDAC_W0_11)	32	RW	0000_0000h
105Ch	Peripheral Domain Access Control (PDAC_W1_11)	32	RW	0000_0000h
1060h	Peripheral Domain Access Control (PDAC_W0_12)	32	RW	0000_0000h
1064h	Peripheral Domain Access Control (PDAC_W1_12)	32	RW	0000_0000h
1068h	Peripheral Domain Access Control (PDAC_W0_13)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
106Ch	Peripheral Domain Access Control (PDAC_W1_13)	32	RW	0000_0000h
1070h	Peripheral Domain Access Control (PDAC_W0_14)	32	RW	0000_0000h
1074h	Peripheral Domain Access Control (PDAC_W1_14)	32	RW	0000_0000h
1078h	Peripheral Domain Access Control (PDAC_W0_15)	32	RW	0000_0000h
107Ch	Peripheral Domain Access Control (PDAC_W1_15)	32	RW	0000_0000h
2000h	Memory Region Descriptor (MRGD_W0_0)	32	RW	0000_0001h
2004h	Memory Region Descriptor (MRGD_W1_0)	32	RW	0000_0001h
2008h	Memory Region Descriptor (MRGD_W2_0)	32	RW	0000_0000h
200Ch	Memory Region Descriptor (MRGD_W3_0)	32	RW	0000_0000h
2020h	Memory Region Descriptor (MRGD_W0_1)	32	RW	0000_0001h
2024h	Memory Region Descriptor (MRGD_W1_1)	32	RW	0000_0001h
2028h	Memory Region Descriptor (MRGD_W2_1)	32	RW	0000_0000h
202Ch	Memory Region Descriptor (MRGD_W3_1)	32	RW	0000_0000h
2040h	Memory Region Descriptor (MRGD_W0_2)	32	RW	0000_0001h
2044h	Memory Region Descriptor (MRGD_W1_2)	32	RW	0000_0001h
2048h	Memory Region Descriptor (MRGD_W2_2)	32	RW	0000_0000h
204Ch	Memory Region Descriptor (MRGD_W3_2)	32	RW	0000_0000h
2060h	Memory Region Descriptor (MRGD_W0_3)	32	RW	0000_0001h
2064h	Memory Region Descriptor (MRGD_W1_3)	32	RW	0000_0001h
2068h	Memory Region Descriptor (MRGD_W2_3)	32	RW	0000_0000h
206Ch	Memory Region Descriptor (MRGD_W3_3)	32	RW	0000_0000h
2200h	Memory Region Descriptor (MRGD_W0_16)	32	RW	0000_0001h
2204h	Memory Region Descriptor (MRGD_W1_16)	32	RW	0000_0001h
2208h	Memory Region Descriptor (MRGD_W2_16)	32	RW	0000_0000h
220Ch	Memory Region Descriptor (MRGD_W3_16)	32	RW	0000_0000h
2220h	Memory Region Descriptor (MRGD_W0_17)	32	RW	0000_0001h
2224h	Memory Region Descriptor (MRGD_W1_17)	32	RW	0000_0001h
2228h	Memory Region Descriptor (MRGD_W2_17)	32	RW	0000_0000h
222Ch	Memory Region Descriptor (MRGD_W3_17)	32	RW	0000_0000h
2240h	Memory Region Descriptor (MRGD_W0_18)	32	RW	0000_0001h
2244h	Memory Region Descriptor (MRGD_W1_18)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2248h	Memory Region Descriptor (MRGD_W2_18)	32	RW	0000_0000h
224Ch	Memory Region Descriptor (MRGD_W3_18)	32	RW	0000_0000h
2260h	Memory Region Descriptor (MRGD_W0_19)	32	RW	0000_0001h
2264h	Memory Region Descriptor (MRGD_W1_19)	32	RW	0000_0001h
2268h	Memory Region Descriptor (MRGD_W2_19)	32	RW	0000_0000h
226Ch	Memory Region Descriptor (MRGD_W3_19)	32	RW	0000_0000h
2400h	Memory Region Descriptor (MRGD_W0_32)	32	RW	0000_0001h
2404h	Memory Region Descriptor (MRGD_W1_32)	32	RW	0000_0001h
2408h	Memory Region Descriptor (MRGD_W2_32)	32	RW	0000_0000h
240Ch	Memory Region Descriptor (MRGD_W3_32)	32	RW	0000_0000h
2420h	Memory Region Descriptor (MRGD_W0_33)	32	RW	0000_0001h
2424h	Memory Region Descriptor (MRGD_W1_33)	32	RW	0000_0001h
2428h	Memory Region Descriptor (MRGD_W2_33)	32	RW	0000_0000h
242Ch	Memory Region Descriptor (MRGD_W3_33)	32	RW	0000_0000h
2440h	Memory Region Descriptor (MRGD_W0_34)	32	RW	0000_0001h
2444h	Memory Region Descriptor (MRGD_W1_34)	32	RW	0000_0001h
2448h	Memory Region Descriptor (MRGD_W2_34)	32	RW	0000_0000h
244Ch	Memory Region Descriptor (MRGD_W3_34)	32	RW	0000_0000h
2460h	Memory Region Descriptor (MRGD_W0_35)	32	RW	0000_0001h
2464h	Memory Region Descriptor (MRGD_W1_35)	32	RW	0000_0001h
2468h	Memory Region Descriptor (MRGD_W2_35)	32	RW	0000_0000h
246Ch	Memory Region Descriptor (MRGD_W3_35)	32	RW	0000_0000h
2480h	Memory Region Descriptor (MRGD_W0_36)	32	RW	0000_0001h
2484h	Memory Region Descriptor (MRGD_W1_36)	32	RW	0000_0001h
2488h	Memory Region Descriptor (MRGD_W2_36)	32	RW	0000_0000h
248Ch	Memory Region Descriptor (MRGD_W3_36)	32	RW	0000_0000h
24A0h	Memory Region Descriptor (MRGD_W0_37)	32	RW	0000_0001h
24A4h	Memory Region Descriptor (MRGD_W1_37)	32	RW	0000_0001h
24A8h	Memory Region Descriptor (MRGD_W2_37)	32	RW	0000_0000h
24ACh	Memory Region Descriptor (MRGD_W3_37)	32	RW	0000_0000h
24C0h	Memory Region Descriptor (MRGD_W0_38)	32	RW	0000_0001h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
24C4h	Memory Region Descriptor (MRGD_W1_38)	32	RW	0000_0001h
24C8h	Memory Region Descriptor (MRGD_W2_38)	32	RW	0000_0000h
24CCh	Memory Region Descriptor (MRGD_W3_38)	32	RW	0000_0000h
24E0h	Memory Region Descriptor (MRGD_W0_39)	32	RW	0000_0001h
24E4h	Memory Region Descriptor (MRGD_W1_39)	32	RW	0000_0001h
24E8h	Memory Region Descriptor (MRGD_W2_39)	32	RW	0000_0000h
24ECh	Memory Region Descriptor (MRGD_W3_39)	32	RW	0000_0000h
2500h	Memory Region Descriptor (MRGD_W0_40)	32	RW	0000_0001h
2504h	Memory Region Descriptor (MRGD_W1_40)	32	RW	0000_0001h
2508h	Memory Region Descriptor (MRGD_W2_40)	32	RW	0000_0000h
250Ch	Memory Region Descriptor (MRGD_W3_40)	32	RW	0000_0000h
2520h	Memory Region Descriptor (MRGD_W0_41)	32	RW	0000_0001h
2524h	Memory Region Descriptor (MRGD_W1_41)	32	RW	0000_0001h
2528h	Memory Region Descriptor (MRGD_W2_41)	32	RW	0000_0000h
252Ch	Memory Region Descriptor (MRGD_W3_41)	32	RW	0000_0000h
2540h	Memory Region Descriptor (MRGD_W0_42)	32	RW	0000_0001h
2544h	Memory Region Descriptor (MRGD_W1_42)	32	RW	0000_0001h
2548h	Memory Region Descriptor (MRGD_W2_42)	32	RW	0000_0000h
254Ch	Memory Region Descriptor (MRGD_W3_42)	32	RW	0000_0000h
2560h	Memory Region Descriptor (MRGD_W0_43)	32	RW	0000_0001h
2564h	Memory Region Descriptor (MRGD_W1_43)	32	RW	0000_0001h
2568h	Memory Region Descriptor (MRGD_W2_43)	32	RW	0000_0000h
256Ch	Memory Region Descriptor (MRGD_W3_43)	32	RW	0000_0000h
2580h	Memory Region Descriptor (MRGD_W0_44)	32	RW	0000_0001h
2584h	Memory Region Descriptor (MRGD_W1_44)	32	RW	0000_0001h
2588h	Memory Region Descriptor (MRGD_W2_44)	32	RW	0000_0000h
258Ch	Memory Region Descriptor (MRGD_W3_44)	32	RW	0000_0000h
25A0h	Memory Region Descriptor (MRGD_W0_45)	32	RW	0000_0001h
25A4h	Memory Region Descriptor (MRGD_W1_45)	32	RW	0000_0001h
25A8h	Memory Region Descriptor (MRGD_W2_45)	32	RW	0000_0000h
25ACh	Memory Region Descriptor (MRGD_W3_45)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
25C0h	Memory Region Descriptor (MRGD_W0_46)	32	RW	0000_0001h
25C4h	Memory Region Descriptor (MRGD_W1_46)	32	RW	0000_0001h
25C8h	Memory Region Descriptor (MRGD_W2_46)	32	RW	0000_0000h
25CCh	Memory Region Descriptor (MRGD_W3_46)	32	RW	0000_0000h
25E0h	Memory Region Descriptor (MRGD_W0_47)	32	RW	0000_0001h
25E4h	Memory Region Descriptor (MRGD_W1_47)	32	RW	0000_0001h
25E8h	Memory Region Descriptor (MRGD_W2_47)	32	RW	0000_0000h
25ECh	Memory Region Descriptor (MRGD_W3_47)	32	RW	0000_0000h
2600h	Memory Region Descriptor (MRGD_W0_48)	32	RW	0000_0001h
2604h	Memory Region Descriptor (MRGD_W1_48)	32	RW	0000_0001h
2608h	Memory Region Descriptor (MRGD_W2_48)	32	RW	0000_0000h
260Ch	Memory Region Descriptor (MRGD_W3_48)	32	RW	0000_0000h
2620h	Memory Region Descriptor (MRGD_W0_49)	32	RW	0000_0001h
2624h	Memory Region Descriptor (MRGD_W1_49)	32	RW	0000_0001h
2628h	Memory Region Descriptor (MRGD_W2_49)	32	RW	0000_0000h
262Ch	Memory Region Descriptor (MRGD_W3_49)	32	RW	0000_0000h
2640h	Memory Region Descriptor (MRGD_W0_50)	32	RW	0000_0001h
2644h	Memory Region Descriptor (MRGD_W1_50)	32	RW	0000_0001h
2648h	Memory Region Descriptor (MRGD_W2_50)	32	RW	0000_0000h
264Ch	Memory Region Descriptor (MRGD_W3_50)	32	RW	0000_0000h
2660h	Memory Region Descriptor (MRGD_W0_51)	32	RW	0000_0001h
2664h	Memory Region Descriptor (MRGD_W1_51)	32	RW	0000_0001h
2668h	Memory Region Descriptor (MRGD_W2_51)	32	RW	0000_0000h
266Ch	Memory Region Descriptor (MRGD_W3_51)	32	RW	0000_0000h
2800h	Memory Region Descriptor (MRGD_W0_64)	32	RW	0000_0001h
2804h	Memory Region Descriptor (MRGD_W1_64)	32	RW	0000_0001h
2808h	Memory Region Descriptor (MRGD_W2_64)	32	RW	0000_0000h
280Ch	Memory Region Descriptor (MRGD_W3_64)	32	RW	0000_0000h
2820h	Memory Region Descriptor (MRGD_W0_65)	32	RW	0000_0001h
2824h	Memory Region Descriptor (MRGD_W1_65)	32	RW	0000_0001h
2828h	Memory Region Descriptor (MRGD_W2_65)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
282Ch	Memory Region Descriptor (MRGD_W3_65)	32	RW	0000_0000h
2840h	Memory Region Descriptor (MRGD_W0_66)	32	RW	0000_0001h
2844h	Memory Region Descriptor (MRGD_W1_66)	32	RW	0000_0001h
2848h	Memory Region Descriptor (MRGD_W2_66)	32	RW	0000_0000h
284Ch	Memory Region Descriptor (MRGD_W3_66)	32	RW	0000_0000h
2860h	Memory Region Descriptor (MRGD_W0_67)	32	RW	0000_0001h
2864h	Memory Region Descriptor (MRGD_W1_67)	32	RW	0000_0001h
2868h	Memory Region Descriptor (MRGD_W2_67)	32	RW	0000_0000h
286Ch	Memory Region Descriptor (MRGD_W3_67)	32	RW	0000_0000h
2A00h	Memory Region Descriptor (MRGD_W0_80)	32	RW	0000_0001h
2A04h	Memory Region Descriptor (MRGD_W1_80)	32	RW	0000_0001h
2A08h	Memory Region Descriptor (MRGD_W2_80)	32	RW	0000_0000h
2A0Ch	Memory Region Descriptor (MRGD_W3_80)	32	RW	0000_0000h
2A20h	Memory Region Descriptor (MRGD_W0_81)	32	RW	0000_0001h
2A24h	Memory Region Descriptor (MRGD_W1_81)	32	RW	0000_0001h
2A28h	Memory Region Descriptor (MRGD_W2_81)	32	RW	0000_0000h
2A2Ch	Memory Region Descriptor (MRGD_W3_81)	32	RW	0000_0000h
2A40h	Memory Region Descriptor (MRGD_W0_82)	32	RW	0000_0001h
2A44h	Memory Region Descriptor (MRGD_W1_82)	32	RW	0000_0001h
2A48h	Memory Region Descriptor (MRGD_W2_82)	32	RW	0000_0000h
2A4Ch	Memory Region Descriptor (MRGD_W3_82)	32	RW	0000_0000h
2A60h	Memory Region Descriptor (MRGD_W0_83)	32	RW	0000_0001h
2A64h	Memory Region Descriptor (MRGD_W1_83)	32	RW	0000_0001h
2A68h	Memory Region Descriptor (MRGD_W2_83)	32	RW	0000_0000h
2A6Ch	Memory Region Descriptor (MRGD_W3_83)	32	RW	0000_0000h

15.4.2.2 Control (CR)

Offset

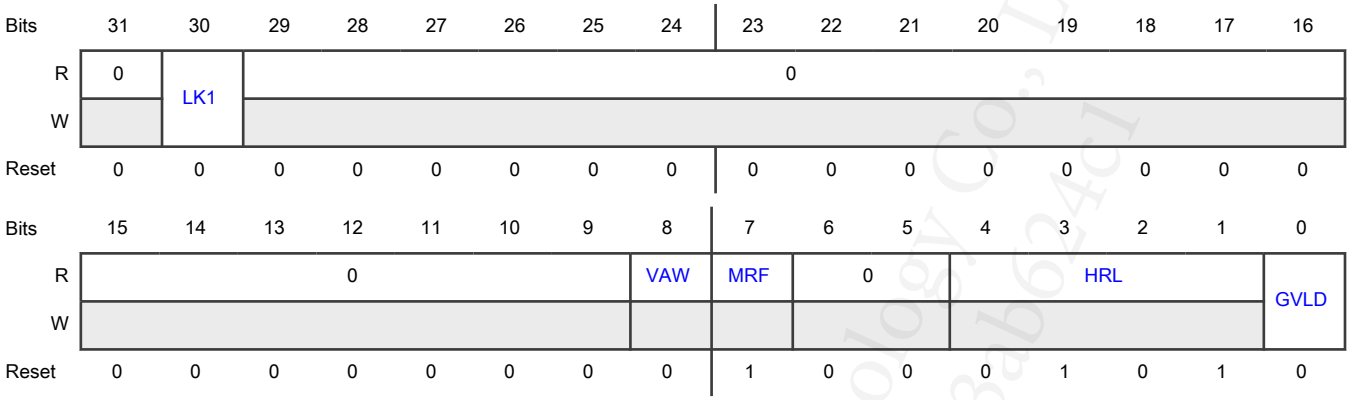
Register	Offset
CR	0h

Function

This register provides the read-only status of the XRDC and a global enable bit for the entire module's operation.

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31 —	Reserved
30 LK1	Lock This field provides a locking mechanism that can be used to prohibit the ability to write to this register. After you write 1 to this field, it remains asserted until the next reset. 0b - Register can be written to by any secure privileged write 1b - Register is locked (read-only) until the next reset
29-9 —	Reserved
8 VAW	Virtualization Aware This field indicates that the domain assignments support the optional inclusion of a logical partition identifier (also known as an operating system number). 0b - Implementation is not virtualization-aware 1b - Implementation is virtualization-aware
7 MRF	Memory Region Format This field indicates that the format of the memory region descriptors and is defined by SoC requirements. 0b - Reserved 1b - SMPU family format
6-5	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4-1 HRL	Hardware Revision Level This field specifies XRDC's hardware and definition revision level. It can be read by software to determine the functional definition of the module. It is hardwired to the value 0101b as it represents an architectural descendent from previous {system} memory protection units ({S}MPU).
0 GVLD	Global Valid (XRDC Global Enable/Disable) 0b - XRDC disabled; all accesses from all bus masters to all slaves allowed 1b - XRDC enabled

15.4.2.3 Hardware Configuration 0 (HWCFG0)

Offset

Register	Offset
HWCFG0	F0h

Function

This read-only register contains information on XRDC's hardware configuration. Specifically, it defines the number of implemented domains and bus masters along with the number of instances of memory region controllers (MRCs) and peripheral access controllers (PACs). The register value at reset is chip-specific. Attempted writes are error-terminated.

Access: Secure privileged read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MID				NPAC				NMRC							
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NMSTR								NDID							
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

Provided under NDA only
COMPANY PROPRIETARY
Beijing CFIT Automotive Technology Co., Ltd.
c8c46e54821a8a8e251390d3ab624c1

Fields

Field	Function
31-28 MID	Module ID
27-24 NPAC	Number Of PACs This field defines the number of PACs minus 1 in the chip. Therefore, you must add 1 to the field value to get the actual number of peripheral access controllers [1-4].
23-16 NMRC	Number Of MRCs This field defines the number of MRCs minus 1 in the chip. Therefore, you must add 1 to the field value to get the actual number of memory region controllers [1-16].
15-8 NMSTR	Number Of Bus Masters This field defines the number of bus masters minus 1 in the chip. Therefore, you must add 1 to the field value to get the actual number of bus masters [1-64].
7-0 NDID	Number Of Domains This field defines the number of domains minus 1 in the chip. Therefore, you must add 1 to the field value to get the actual number of domains [1-16].

15.4.2.4 Hardware Configuration 1 (HWCFG1)

Offset

Register	Offset
HWCFG1	F4h

Function

This register contains information on XRDC's hardware configuration. It provides a mechanism for software to determine its domain number by reading the register. See Domain error capture management for details on typical usage. Attempted writes are error-terminated.

Access: Secure Privileged Read

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												DID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	u[1]	u	u	u

1. Reset value is determined by current configuration of master accessing.

Fields

Field	Function
31-4 —	Reserved
3-0 DID	Domain Identifier Number This field provides the domain number [0-15] of the requesting bus master.

15.4.2.5 Hardware Configuration 2 (HWCFG2)

Offset

Register	Offset
HWCFG2	F8h

Function

This register contains information on XRDC's hardware configuration. It provides a bitmap that signals the presence of a process identifier (PID) register sourced from the given bus master. The HWCFG2 register is associated with bus masters [31-0]. Attempted writes are error-terminated.

Access: Secure privileged read

Provided under NDA only
COMPANY PROPRIETARY
Beijing CFA Automotive Technology Co., Ltd.
c8c462e5-4861-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PIDP31	PIDP30	PIDP29	PIDP28	PIDP27	PIDP26	PIDP25	PIDP24	PIDP23	PIDP22	PIDP21	PIDP20	PIDP19	PIDP18	PIDP17	PIDP16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PIDP15	PIDP14	PIDP13	PIDP12	PIDP11	PIDP10	PIDP9	PIDP8	PIDP7	PIDP6	PIDP5	PIDP4	PIDP3	PIDP2	PIDP1	PIDP0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 PIDPn	Process Identifier Process identifier present from bus master *n*. This field provides a bitmap to signal that bus master *n* sources a process identifier register to the MDAC logic. 0b - Bus master *n* does not source a process identifier register; MDAC logic provides the needed PID for processor cores 1b - Bus master *i* sources a process identifier register to MDAC logic

15.4.2.6 Master Domain Assignment Configuration (MDACFG0 - MDACFG7)

Offset

For m = 0 to 7:

Register	Offset
MDACFGm	100h + (m × 1h)

Function

This register defines the number of implemented domain assignment registers for bus master *m*, where *m*+1 can specify bus masters from 1 to 64. These registers are organized as a byte-sized data array and can be read using 8-, 16- or 32-bit accesses. An all-zero value (NCM = 0, NMDAR = 0) indicates a non-existent bus master. Attempted writes are error-terminated.

Typically, processor bus masters include one or more domain assignment registers, while non-processor masters have a single domain assignment register.

Access: Secure privileged read

Provided under NDA only
COMMERCIAL IN CONFIDENCE
Beijing HJ Automotive Technology Co., Ltd.
c8c462c3-821-48a8-b213-3e0d3ab6240

Diagram

Bits	7	6	5	4	3	2	1	0
R	NCM	0			NMDAR			
W								
Reset	1	0	0	0	0	0	0	1

Fields

Field	Function
7 NCM	Non-CPU Master This field indicates that bus master *m* is a non-CPU master. It specifies that the format of the associated MDA_W*r_m* register defines a non-processor domain assignment. The value of the field is 0 for a non-existent bus master. 0b - Bus master is a processor 1b - Bus master is a non-processor
6-4 —	Reserved
3-0 NMDAR	Number Of Master Domain Assignment Registers For Bus Master *m* This field specifies the number of registers associated with the master domain assignment register for a given bus master. The value is limited to the range [0-8], where 0 indicates a non-existent bus master and non-zero values indicate the number of implemented registers associated with this MDA*m*.

15.4.2.7 Memory Region Configuration (MRCFG0 - MRCFG5)

Offset

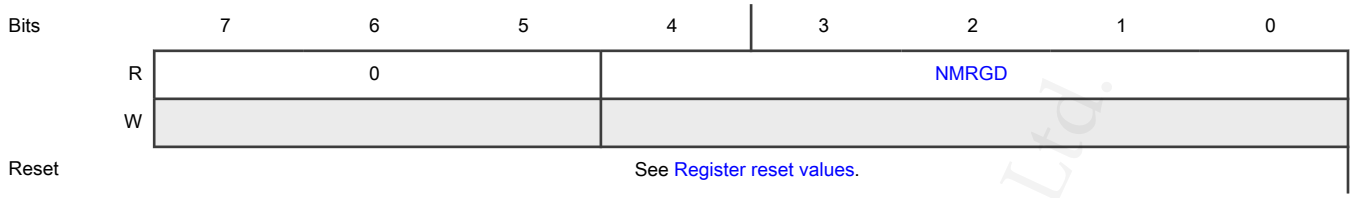
Register	Offset
MRCFG0	140h
MRCFG1	141h
MRCFG2	142h
MRCFG3	143h
MRCFG4	144h
MRCFG5	145h

Function

This read-only register defines the number of implemented memory region descriptors for each MRC*r*, where *r*+1 can specify up to 16 instances. These registers are organized as byte-sized data arrays and can be read using 8-, 16- or 32-bit accesses. A zero value indicates a non-existent memory region controller instance. Attempted writes are error-terminated.

Access: Secure Privileged Read

Diagram



Register reset values

Register	Reset value
MRCFG0–MRCFG1	04h
MRCFG2	10h
MRCFG3–MRCFG5	04h

Fields

Field	Function
7-5 —	Reserved
4-0 NMRGD	Number Of Memory Region Descriptors For Memory Region Controller *n* Number of memory region descriptors for MRC*r*. This field specifies the number of memory region descriptors associated with a given memory region controller instance. The value is limited to the range [0-16], where 0 indicates a non-existent MRC instance and non-zero values indicate the number of implemented memory region descriptors [4, 8,12,16] associated with the MRC*r* submodule.

15.4.2.8 Domain Error Location (DERRLOC0 - DERRLOC7)

Offset

For d = 0 to 7:

Register	Offset
DERRLOCd	200h + (d × 4h)

Function

This array of read-only registers provide the instance number of the submodule where an access violation occurred. These registers are organized as a word array, which is indexed by the faulting domain number, *d*. The two fields of this register provide a bitmap of instances associated with all submodules containing captured error information for that domain. These instance numbers are then used as indices into the DERR_W0_*i*, DERR_W1_*i*, DERR_W2_*i*, and DERR_W3_*i* register arrays. See Domain error capture management for details.

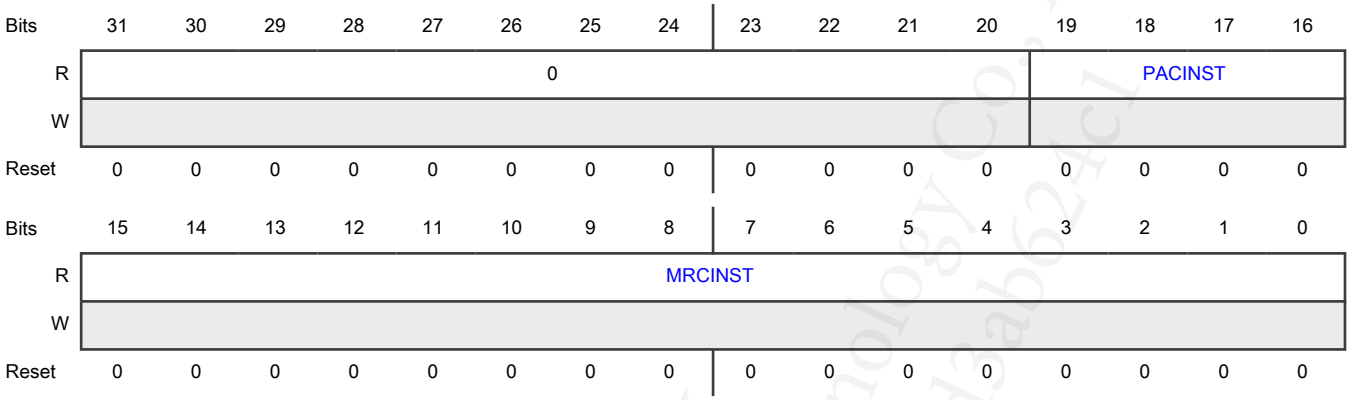
When an access violation is detected by either a memory region controller (MRC) or a peripheral access controller (PAC), address and attribute information of the offending access is captured. Using the faulting DID number as the index, *d*, this array of read-only registers provides additional information signaling the instance number of the submodule where the access violation(s) occurred.

Because the resulting exception handler needs the submodule instance to retrieve the captured address and attribute information from DERR_W0_*i* and DERR_W1_*i*, these registers provide the instance number details.

Attempted writes are error-terminated.

Access: Secure privileged read

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 PACINST	PAC Instance This field is a bitmap indicating the presence of a detected access violation for domain *d* in the implemented instances of the PAC. The least-significant bit of this field (bit 16) corresponds to PAC instance 0. The most-significant bit of this field (bit 19) corresponds to PAC instance 3, and so on. Multiple bits can be set at any time indicating access violations for this domain have been detected across multiple instances of the PACs. **NOTE** When a memory slot controller (MSC) is present, it appears here in the same way as a PAC instance. For each bit in this field: • 0 - The peripheral access controller has not detected an access violation or is not physically present. • 1 - The peripheral access controller has detected one or more access violations for this domain.
15-0 MRCINST	MRC Instance This field is a bitmap indicating the presence of a detected access violation for domain *d* in the implemented instances of the MRC. The least-significant bit of this field (bit 0) corresponds to MRC instance 0. The most-significant bit of this field (bit 3) corresponds to MRC instance 3, and so on. Multiple bits can be set at any time to indicate access violations for this domain have been detected across multiple instances of the MRCs. For each bit in this field:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 0 - The memory region controller has not detected an access violation or is not physically present. • 1 - The memory region controller has detected one or more access violations for this domain.

15.4.2.9 Domain Error Word0 (DERR_W0_0 - DERR_W0_16)

Offset

Register	Offset
DERR_W0_0	400h
DERR_W0_1	410h
DERR_W0_2	420h
DERR_W0_3	430h
DERR_W0_4	440h
DERR_W0_5	450h
DERR_W0_16	500h

Function

This read-only register array provides the address of an access violation detected by either an MRC or a PAC. These registers are organized as a word array, which is indexed by the violating submodule instance number. That is, the index, *i*, of this array is the instance number of the submodule that detected the access violation. The submodule instance numbers are provided by the DERRLOC registers. The memory-mapped error capture detail registers are organized as 20 sequential 16-byte entries. The first 16 register sets are associated with the MRCs. The following four register sets are associated with the PACs. Each 16-byte structure contains DERR_W0_*i*, DERR_W1_*i*, DERR_W2_*i*, and DERR_W3_*i*.

The error capture registers in the MRC and PAC submodules contain physical registers for each domain, but are organized in the DERR_W0_*i* and DERR_W1_*i* registers to provide the information for the requesting domain only. Thus, the access violation exception handler for each domain only has visibility into the captured error information for its own domain. See Domain error capture management for details.

When an access violation is detected and the offending information captured, subsequent updates to this register are disabled until the required data pattern is written to the DERR_W3_*i* register. At that time, this register is cleared and re-enabled to capture the next access violation.

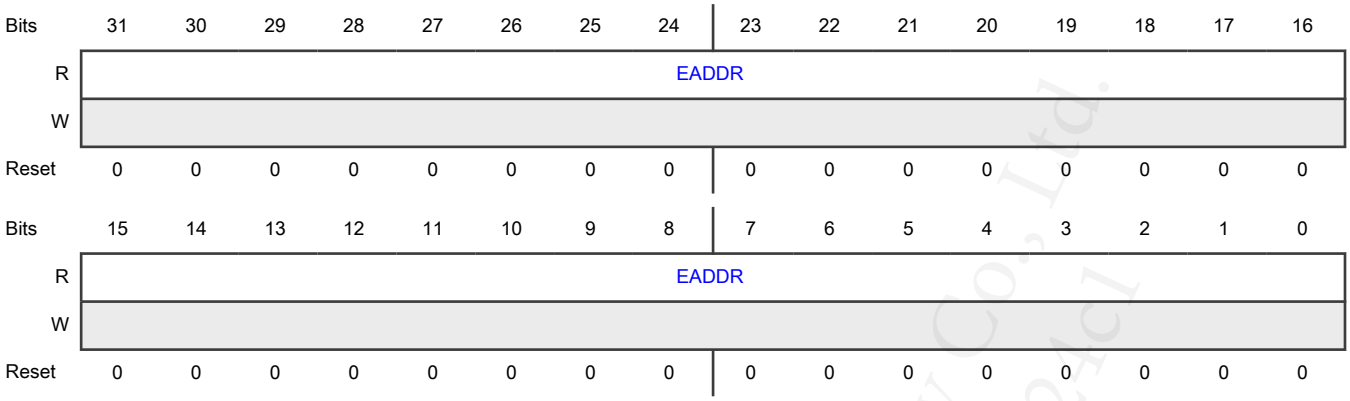
Attempted writes are error-terminated as are attempted reads of an MRC or PAC instance that is not physically present.

NOTE

If masters with the same DID cause simultaneous error accesses, DERR_W0_*i* records only the error of the lowest slave index.

Access: Secure privileged read

Diagram



Fields

Field	Function
31-0 EADDR	Error Address This is the access address that generated an access violation.

15.4.2.10 Domain Error Word1 (DERR_W1_0 - DERR_W1_16)

Offset

Register	Offset
DERR_W1_0	404h
DERR_W1_1	414h
DERR_W1_2	424h
DERR_W1_3	434h
DERR_W1_4	444h
DERR_W1_5	454h
DERR_W1_16	504h

Function

This read-only register array provides the attributes of an access violation detected by either an MRC or a PAC. These registers are organized as a word array, which is indexed by the violating submodule instance number. See Domain Error Location (DERRLOC0 - DERRLOC7), Domain Error Word0 (DERR_W0_0 - DERR_W0_16), and Domain error capture management for more information.

NOTE

If masters with the same DID cause simultaneous error accesses, DERR_W1_*i*records only the error of the lowest slave index.

Access: Secure privileged read

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462f-4821-48a8-b251-390d8ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EST		0			EPORT			0							EA40FMT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				ERW	EATR			0				EDID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 EST	Error State This field signals the state of access violations for this domain in this instance of the memory region controller or peripheral access controller. After an access violation has been detected and the faulting address and attribute information stored, subsequent errors are recorded as an overrun condition without any data captured. After retrieving the faulting address and attributes, the error capture mechanism must be rearmed by performing a write to DERR_W3_*i*. 00-01b - No access violation detected 10b - A single access violation was detected 11b - Multiple access violations for this domain are detected by this submodule instance. Only the address and attribute information for the first error are captured in DERR_W0_*i* and DERR_W1_*i*.
29-27 —	Reserved
26-24 EPORT	Error Port This field identifies the encoded port number of the MRC that detected the access violation. The MRC port number connection is chip-specific. See the chip configuration details for more information. For access violations detected by the PAC, this field is cleared.
23-17 —	Reserved
16 EA40FMT	Error Address 40 Format Domain Error Word2 (DERR_W2_0 - DERR_W2_16) contains EADDR[39:32] 0b - 32-bit format 1b - 40-bit format

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 —	Reserved
11 ERW	Error Read/Write This field signals whether the captured access violation occurred on a read or write reference. 0b - Read access 1b - Write access
10-8 EATR	Error Attributes This field captures certain attributes of the access violation 000b - Secure user mode, instruction fetch access 001b - Secure user mode, data access 010b - Secure privileged mode, instruction fetch access 011b - Secure privileged mode, data access 100b - Nonsecure user mode, instruction fetch access 101b - Nonsecure user mode, data access 110b - Nonsecure privileged mode, instruction fetch access 111b - Nonsecure privileged mode, data access
7-4 —	Reserved
3-0 EDID	Error Domain Identifier This field captures the domain identifier of the access violation.

15.4.2.11 Domain Error Word2 (DERR_W2_0 - DERR_W2_16)

Offset

Register	Offset
DERR_W2_0	408h
DERR_W2_1	418h
DERR_W2_2	428h
DERR_W2_3	438h
DERR_W2_4	448h
DERR_W2_5	458h
DERR_W2_16	508h

Provided under NDA to Beijing CHJ Automotive Technology Co., Ltd. COMPANY PROPRIETARY c8c462e5-4821-4aa8-b251-390d3ab624c1

Function

This is a word-sized location in the domain error details data structure. The registers are organized in a data structure based on instance number, that is, n = the instance number of the submodule which detected the access violation, and are read using information directly extracted from the DERRLOCn registers. The memory-mapped capture error detail registers are organized as 20 sequential 16-byte entries. The first 16 register sets are associated with the memory region controllers. The following four registers sets are associated with the peripheral access controllers. Each 16-byte record contains DERR_W0_*i*, DERR_W1_*i*, DERR_W2_*i* and DERR_W3_*i*. See Domain error capture management for more details. A read of this location returns zeroes. Attempted writes are error-terminated as are attempted reads of an MRC or PAC instance that is not physically present.

NOTE

If masters with the same DID cause simultaneous error accesses, DERR_W2_*i* will record only the error of the lowest slave index.

Access: Secure Privileged Read

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								EADDR39_32							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 EADDR39_32	EADDR[39:32]

15.4.2.12 Domain Error Word3 (DERR_W3_0 - DERR_W3_16)

Offset

Register	Offset
DERR_W3_0	40Ch
DERR_W3_1	41Ch
DERR_W3_2	42Ch

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,
c8c462e5-4821-48a8-b251-590d3b624c1

Table continued from the previous page...

Register	Offset
DERR_W3_3	43Ch
DERR_W3_4	44Ch
DERR_W3_5	45Ch
DERR_W3_16	50Ch

Function

This register rearms the error capture logic and clears the DERR_W0_*i* and DERR_W1_*i* registers. After the domain access violation error details are read, typically in an exception service routine, a 32-bit word write to this register is required to rearm the error capture logic.

It also clears the corresponding field in DERRLOC*n* register.

A read of this location returns a value of 0000h. Attempted reads of an MRC or PAC instance that is not physically present are error-terminated.

See Domain error capture management for details.

NOTE

If masters with the same DID cause simultaneous error accesses, DERR_W3_*i* records only the error of the lowest slave index.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0													
W	RECR															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30	Rearm Error Capture Registers
RECR	This field controls the rearming of the domain error capture registers. After an access violation is detected with the faulting address and attribute information stored, subsequent errors are recorded as an overrun condition without any data captured. 00, 10, 11b - Writing this value has no effect

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Writing this value rearms the error capture mechanism and clears registers DERR_W0_*i* and DERR_W1_*i*
29-0 —	Reserved

15.4.2.13 Master Domain Assignment (MDA_W0_0_DFMT1 - MDA_W0_7_DFMT1)

Offset

For m = 0 to 7:

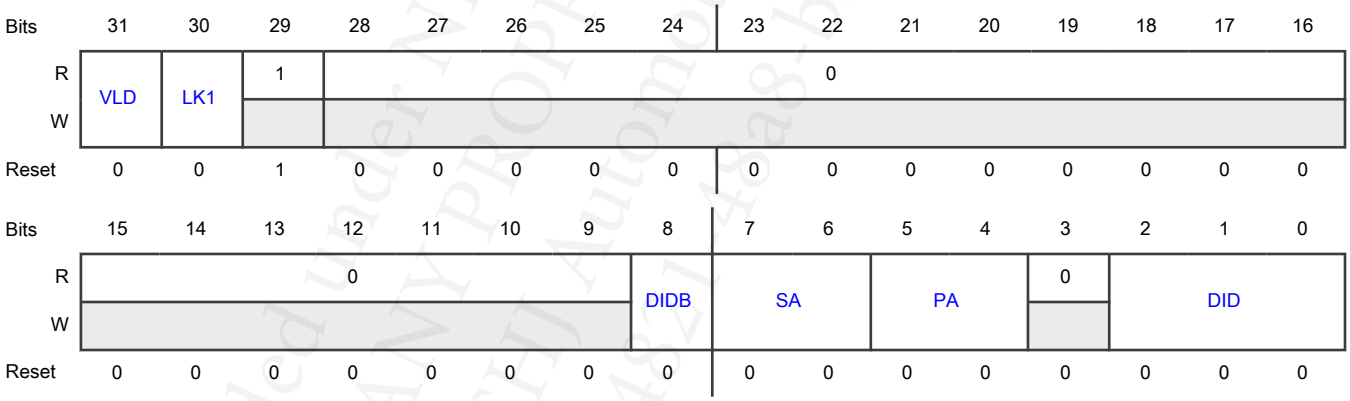
Register	Offset
MDA_W0_m_DFMT1	800h + (m × 20h)

Function

This register is identical to Master Domain Assignment (MDA_W*r_m*_DFMT0) except that the value of the domain format field, DFMT, is 1. This format supports two different specifications of DID for non-core bus masters.

Access: Secure privileged read/write

Diagram



Fields

Field	Function
31 VLD	Valid This field indicates that the domain assignment is valid. It is further qualified if the value of CR[GVLD] is 1. If the value of CR[GVLD] is 0, the DID output is defined by the chip-specific default DID value. See the XRDC chip configuration section for details. If both MDA_W*r_m*[VLD] and CR[GVLD] are asserted, the DID output is defined by the remaining contents of this register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - The W*r* domain assignment is invalid 1b - The W*r* domain assignment is valid
30 LK1	Lock This field provides a locking mechanism that can be used to limit the ability to write to the register. After this bit is set, it remains asserted until the next reset. 0b - Register can be written to by any secure privileged write 1b - Register is locked (read-only) until the next reset
29 DFMT	Domain Format Identifies this register's domain assignment. 1b - Non-processor domain assignment
28-9 —	Reserved
8 DIDB	DID Bypass If asserted, this field enables the bypassing of an input DID value as the domain identifier for this non-processor bus master. This capability allows non-processor bus masters (for example, a DMA) to masquerade as a processor. After this field is set to 1, it remains at that value until the next reset. 0b - Use MDA*n*[3:0] as the domain identifier 1b - Use the DID input as the domain identifier
7-6 SA	Secure Attribute This field defines the secure/nonsecure attribute for non-processor cores. **NOTE** The bus master's input secure/nonsecure attribute is used if SA = 1X, or this VLD = 0. 00b - Force the bus attribute for this master to secure. 01b - Force the bus attribute for this master to nonsecure. 1xb - Use the bus master's secure/nonsecure attribute directly.
5-4 PA	Privileged Attribute This field defines the privileged/user attribute for non-processor cores. **NOTE** The bus master's input privileged or user attribute is used if PA = 1X, or this VLD = 0. 00b - Force the bus attribute for this master to user. 01b - Force the bus attribute for this master to privileged.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1xb - Use the bus master's privileged/user attribute directly.
3 —	Reserved
2-0 DID	Domain Identifier Contains the value of DID to be generated

15.4.2.14 Peripheral Domain Access Control (PDAC_W0_0 - PDAC_W0_15)

Offset

Register	Offset
PDAC_W0_(0 * 128) + s	1000h + (s × 8h)

Function

This register array, PDAC_W*r*_*s*, provides a two-dimensional data structure for defining access control policies per domain for each implemented slave peripheral. In the offset equation above, *c* is the instance number and *s* is the slot number. See Table 57 for access control policies. There are two word-sized registers assigned for each peripheral address slot (W*r*).

Each W*r* within the PDAC*s* structure is a word-sized definition; a total of two words (64 bits) is needed to define the access control policies for all the domains.

These registers also allow a hardware semaphore to be optionally included in the access control evaluation for peripherals shared by multiple domains. If enabled, the state of the semaphore dynamically modifies the access control policies so that only the domain owning it has write permission into the peripheral. The write permissions for all other domains are revoked based on the semaphore state. If the semaphore is not owned by any domain, the D*x*ACP fields are evaluated normally.

The PAC submodule uses the peripheral access address to select the appropriate PDAC*s* register, and then uses the domain identifier associated with the access to select the appropriate D*x*ACP field. The selected D*x*ACP field is then evaluated against the attributes associated with the memory transaction to determine the validity of the access. See DxACP evaluation for details. If allowed, the access proceeds, else an access violation is signaled, the transfer error-terminated, and the appropriate address and attribute information captured in the corresponding error detail registers.

Access: Secure privileged read/write

Table 57. Access control policy

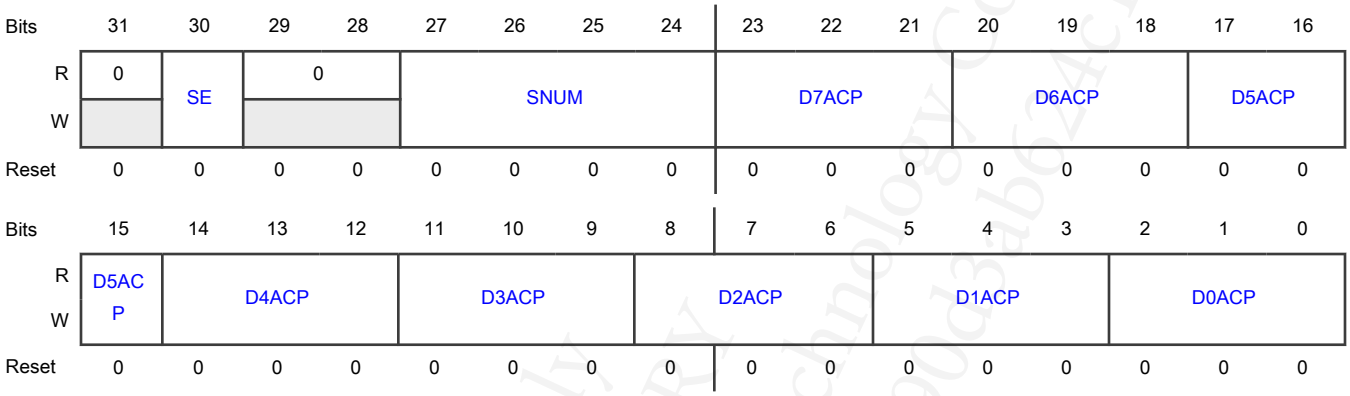
Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
111	r, w	r, w	r, w	r, w
110	r, w	r, w	r, w	none
101	r, w	r, w	r	r
100	r, w	r, w	r	none
011	r, w	r, w	none	none

Table continues on the next page...

Table 57. Access control policy (continued)

Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
010	r, w	none	none	none
001	r	r	none	none
000	none	none	none	none

Diagram



Fields

Field	Function
31 —	Reserved
30 SE	Semaphore Enable This field indicates if a semaphore, SNUM, is to be included in the D*x*ACP access evaluation. 0b - Do not include a semaphore in the D*x*ACP evaluation 1b - Include the semaphore defined by SNUM in the D*x*ACP evaluation
29-28 —	Reserved
27-24 SNUM	Semaphore Number Include this hardware semaphore in the D*x*ACP access evaluation.
23-21: D7ACP 20-18: D6ACP 17-15: D5ACP 14-12: D4ACP 11-9: D3ACP 8-6: D2ACP	Domain *n* Access Control Policy This field defines access control rights for the selected domain. The encodings specify read and write access capabilities based on the four operating states. These fields are only implemented for supported DIDs; unsupported DIDs are not implemented and default to a zero value, providing no access rights. See Table 57 for access control policies.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-3: D1ACP 2-0: D0ACP	

15.4.2.15 Peripheral Domain Access Control (PDAC_W1_0 - PDAC_W1_15)

Offset

Register	Offset
PDAC_W1_(0 * 128) + s	1004h + (s × 8h)

Function

See Peripheral Domain Access Control (PDAC_W0_0 - PDAC_W0_15) for description.

Access: Secure privileged read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	VLD	LK2		0					0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 VLD	Valid This field indicates that the peripheral domain access control definition is valid. It is further qualified if the value of CR[GVLD] is 1. If value of either CR[GVLD] or this field is 0, all accesses to the peripheral are allowed. Any write to PDAC_W0_*s* clears the PDAC_W1_*s*[VLD] indicator, so a coherent register state can be supported. 0b - The PDAC*s* assignment is invalid. 1b - The PDAC*s* assignment is valid.
30-29	Lock

Table continues on the next page...

Table continued from the previous page...

Field	Function
LK2	This field provides a mechanism to limit writes to the PDAC registers to protect their contents. After you write 1 to the bits of this field, they individually remain asserted until the next reset. 00-01b - All PDAC*s* can be written to 10b - Domain *x* can only update the D*x*ACP field; no other PDAC fields can be written to 11b - PDACs are locked (read-only) until the next reset
28-24 —	Reserved
23-0 —	Reserved

15.4.2.16 Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_83)

Offset

Register	Offset
MRGD_W0_n * 16 + m	2000h + (n × 200h) + (m × 20h)

Function

This set of registers provide a two-dimensional data structure for defining access control policies (see Table 58) per domain for each supported memory region. There are four word-size registers (W*o*) defined in each memory region. There are up to 16 memory region controllers (MRCs), *n*, each supporting up to 16 memory region descriptors (MRGD), *m*. The number of MRCs is defined by HWCFG0[NMRC].

Each instance of an MRC submodule can monitor up to eight system buses concurrently.

NOTE

An AXI port counts as two system buses because the read address and write address channels operate independently.

Each W*o* within the MRGD*m* structure is a word-sized definition; a total of four words (128 bits, W0-W3) are needed to completely define the memory region descriptor.

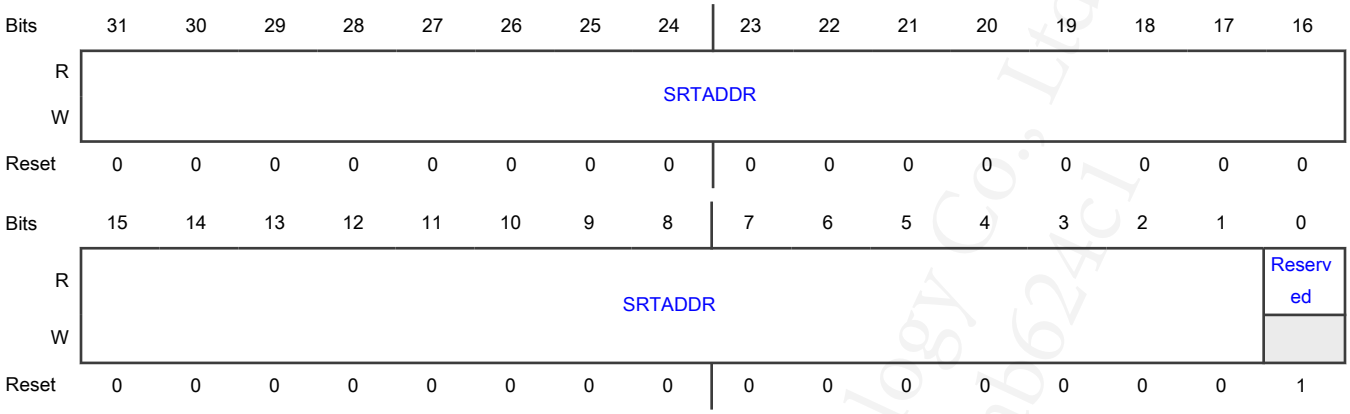
XRDC uses separate start (W0) and end addresses (W1) to define memory regions and leverages capabilities formerly provided in SMPUs.

The region descriptor also allows a hardware semaphore to be optionally included in the access control evaluation for memory regions shared by multiple domains. If enabled, the state of the semaphore dynamically modifies the access control policies so that only the domain owning it has write permission into the region. The write permissions for all other domains are revoked based on the semaphore state. If the semaphore is not owned by any domain, the D*x*ACP fields are evaluated normally.

The MRC submodule evaluates multiple system bus access addresses versus the programmed address ranges to determine a region hit. After a region hit is determined, the MRC uses the domain identifier associated with the access to select the appropriate D*x*ACP field from the corresponding MRGD*n*. The selected D*x*ACP field is then evaluated against the attributes associated with the memory transaction to determine the validity of the access. See Memory region descriptor hit determination and Memory

region DxACP evaluation for details. If allowed, the access proceeds, else an access violation is signaled, the transfer error terminated, and the appropriate address and attribute information captured in the corresponding error detail registers.

Diagram



Fields

Field	Function
31-1 SRTADDR	Start Address This field defines bits 35:5 of the 0-modulo-32-byte start address of the memory region. The minimum region size is 32 bytes. This field expects the address to be entered 4 bits shifted to the right.
0 —	Reserved

15.4.2.17 Memory Region Descriptor (MRGD_W1_0 - MRGD_W1_83)

Offset

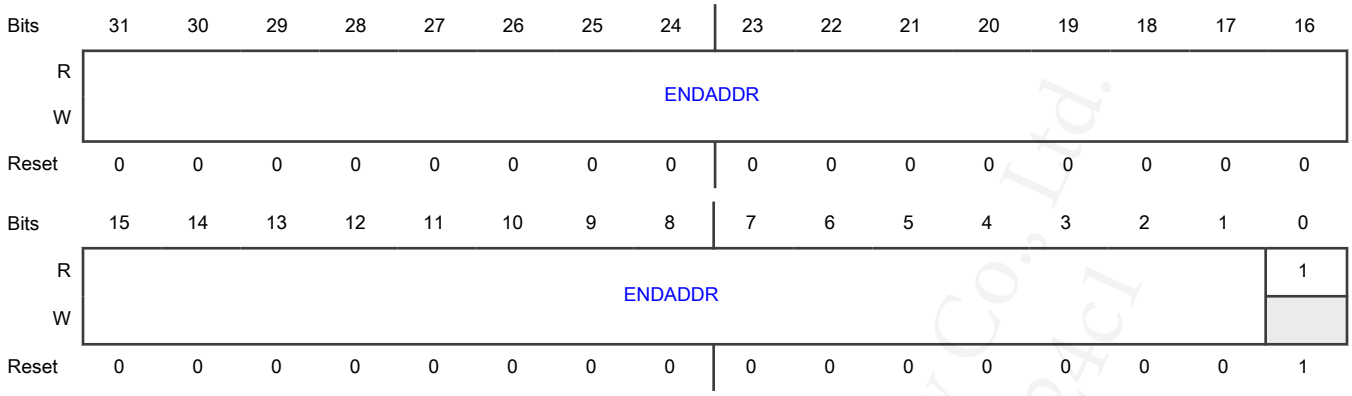
Register	Offset
MRGD_W1_n * 16 + m	2004h + (n × 200h) + (m × 20h)

Function

See Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_83) for description.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-4888-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-1 ENDADDR	End Address This field defines bits 35:5 of the 31-modulo-32-byte end address of the memory region. This field expects the address to be entered 4 bits shifted to the right.
0 —	Reserved

15.4.2.18 Memory Region Descriptor (MRGD_W2_0 - MRGD_W2_83)

Offset

Register	Offset
MRGD_W2_n * 16 + m	2008h + (n × 200h) + (m × 20h)

Function

See Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_83) for description.

Table 58. Access control policy

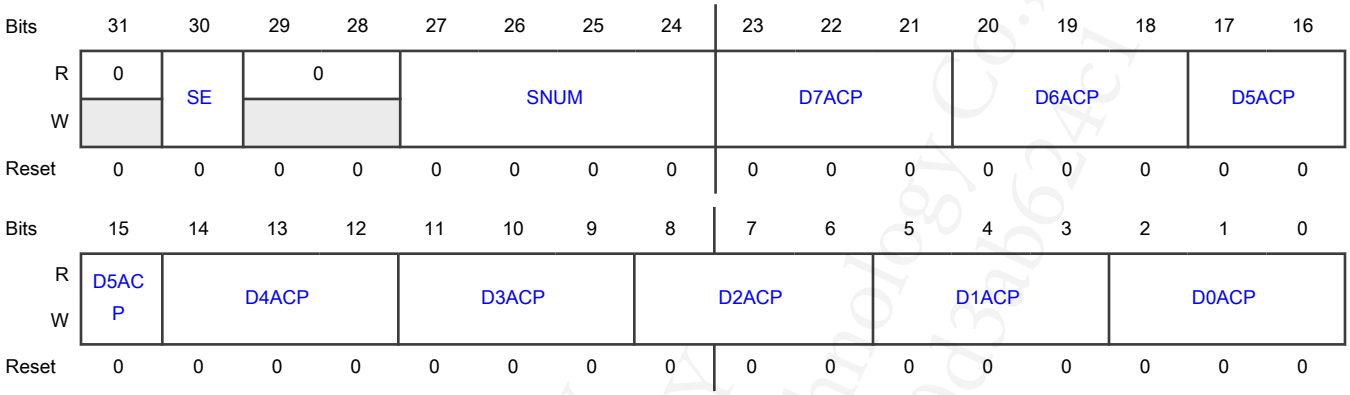
Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
111	r, w	r, w	r, w	r, w
110	r, w	r, w	r, w	none
101	r, w	r, w	r	r
100	r, w	r, w	r	none
011	r, w	r, w	none	none
010	r, w	none	none	none

Table continues on the next page...

Table 58. Access control policy (continued)

Policy	SecurePriv	SecureUser	NonsecurePriv	NonsecureUser
001	r	r	none	none
000	none	none	none	none

Diagram



Fields

Field	Function
31 —	Reserved
30 SE	Semaphore Enable This field indicates if a semaphore, SNUM, is to be included in the D*x*ACP access evaluation. 0b - Do not include a semaphore in the D*x*ACP evaluation. 1b - Include the semaphore defined by SNUM in the D*x*ACP evaluation.
29-28 —	Reserved
27-24 SNUM	Semaphore Number Include this hardware semaphore in the D*x*ACP access evaluation.
23-21: D7ACP 20-18: D6ACP 17-15: D5ACP 14-12: D4ACP 11-9: D3ACP 8-6: D2ACP 5-3: D1ACP 2-0: D0ACP	Domain *n* Access Control Policy This field defines the access control rights for the selected domain. The encodings specify read and write access capabilities based on the four operating states. These fields are only implemented for supported DIDs; unsupported DIDs are not implemented and default to a zero value, providing no access rights. See the Access control policy table above.

Preliminary Information
COMPANY CONFIDENTIAL

15.4.2.19 Memory Region Descriptor (MRGD_W3_0 - MRGD_W3_83)

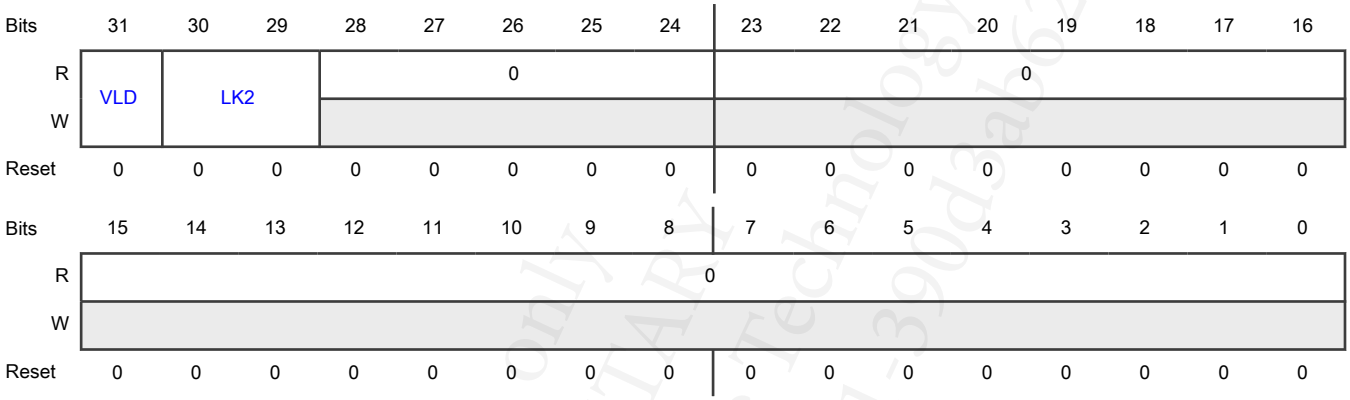
Offset

Register	Offset
MRGD_W3_n * 16 + m	200Ch + (n × 200h) + (m × 20h)

Function

See Memory Region Descriptor (MRGD_W0_0 - MRGD_W0_83) for description.

Diagram



Fields

Field	Function
31 VLD	Valid This field indicates that the memory region descriptor definition is valid. It is further qualified if the value of CR[GVLD] is 1. If the value of CR[GVLD] is 0, then all accesses to the memory region are allowed. Any write to MRGD_W[0-2]_*n* clears the MRGD_W3_*n*[VLD] indicator, so a coherent register state can be supported. 0b - The MRGD*n* assignment is invalid. 1b - The MRGD*n* assignment is valid.
30-29 LK2	Lock This field provides a mechanism to limit writes to the MRGD*n* register to protect its contents. After you write 1 to the bits of this field, they individually remain asserted until the next reset. 00b - All MRGD*n* registers can be written to 01b - Reserved 10b - Domain *x* can only update the D*x*ACP field for domain *x*; no other ACP fields can be written to 11b - MRGD*n* registers are locked (read-only) until the next reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
28-24 —	Reserved
23-0 —	Reserved

15.5 Functional description

This section provides details on the operation and implementation of the XRDC submodules.

15.5.1 Manager (MGR)

The MGR submodule is responsible for coordinating XRDC's programming model reads and writes.

The complete programming model is large, requiring four 4 KB slave peripheral address space slots, although most chips define a sparsely populated subset. Additionally, the programming model is distributed across the various submodule instances (MDAC, MRC, and PAC). The MGR submodule provides the required hardware management for all programming model accesses, generating and distributing the appropriate address decodes and write data to the MDAC, MRC, and PAC submodule instances, and collecting and combining all the register read data buses and responses.

MGR implements the Control Register, all the read-only hardware configuration registers (HWCFG[0-3], MDACFG*n*, and MRCFG), and the program-visible versions of the domain error capture registers. To support the domain error reporting functionality, the MGR submodule collects the individually captured error indicator output signals from all the submodule instances and remaps them so they are organized into an instance bitmap by domain, as specified by the DERRLOC*n* register array.

15.5.2 Master domain assignment controller (MDAC)

The MDAC submodule is responsible for the generation of the domainID on every memory transaction for every bus master in the chip. The resulting domainID and {nonsecure, privileged} signals are generated. They are then treated as address attributes and associated with each transaction as a given transaction moves through the system.

MDAC modules that drive the DIDs for processor cores include an option of processor-identifier (PID) matching to select between DIDs. This is accomplished by first enabling PID checking by setting MDA_W*m_n*[PE] = [2,3]. With PID matching enabled, a domain hit is determined by matching MDA_W*m_n*[PID] with the PID input or PID in the PID register. The PIDM field can be used to match specific groups of PIDs.

As a simple example, assume a system with two DIDs.

- DID 1 is for critical tasks (criticality is system-specific and may refer to timing, safety, and so on).
- DID 2 is for non-critical tasks.
- Two DIDs require two MDA_W*m_n* registers, MDA_W0 and MDA_W1.

Further, assume that the processor's task identifier defines whether a task is critical or non-critical, as defined by system software.

- PIDs [0-15] are assigned to critical tasks.
- PIDs != [0-15] are assigned to noncritical tasks.

During startup, software initializes the MDA_W[0-1] registers. MDA_W0 defines DID = 1 for PIDs [0-15] and MDA_W1 defines DID = 2 for PIDs != [0-15].

- MDA_W0 value is 8000_0F81h, which corresponds to these individual field values:
  - VLD = 1

  - PID = 0h
  - PIDM = Fh
  - PE = 2
  - DIDS = 0
  - DID = 1
- MDA_W1 value is 8000_0FC2h, which corresponds to these individual field values:
  - VLD = 1
  - PID = 0h
  - PIDM = Fh
  - PE = 3
  - DIDS = 0
  - DID = 2

When the processor executes, it loads the appropriate task ID into its PID register. The processor's PID register value is input for the corresponding MDAC module and is used by the MDA_W*m* comparison logic. The system then dynamically generates two DIDs, and the downstream XRDC access check logic can distinguish and enforce different access control rights based on the different DIDs.

A special mode is provided for driving the DID of non-processor masters. This is enabled with the DID bypass field, MDA_W0_*n*_DFMT1[DIDB]. The MDAC has a DID input that can be driven by a master and used directly as the DID output. This allows variability in the DID, which is dictated by the master and is intended to be used by DMA masters that masquerade as (that is, drive the DID of) the master that programmed it.

15.5.3 D*x*ACP evaluation

Fundamental to XRDC's operation is the actual access violation check performed by the MRC and PAC controllers.

Previous descriptions have detailed the domain assignment mechanisms and the tracking of the DID as an address attribute through the system bus switching fabric(s). As transactions reach the slave memory and peripheral controllers, the address is used to select the appropriate memory region descriptor or peripheral access control register. After the appropriate register is selected, the next function is the actual access evaluation.

For this step, the DID of the current transaction selects the appropriate 3-bit D*x*ACP (domain *x* access control policy) from the source register. The D*x*ACP field defines the access rights for the domain based on the four-level model combined with the transaction's access attributes (read = 0/write = 1, secure = 0/nonsecure = 1, user = 0/privileged = 1).

The four-level model is defined by the concatenation of two address attributes: secure/nonsecure and user/privileged. These two attribute signals (nonsecure, privileged) and the resulting access levels are defined as follows:

```
if {nonsecure, privileged} == 01b, then level = SecurePriv(ileged)
if {nonsecure, privileged} == 00b, then level = SecureUser
if {nonsecure, privileged} == 11b, then level = NonsecurePriv(ileged)
if {nonsecure, privileged} == 10b, then level = NonsecureUser
```

The resulting hierarchical access control model specifies the following:

```
SecurePriv(ileged) > SecureUser > NonsecurePriv(ileged) > NonsecureUser
```

Table 59 defines XRDC's evaluation of the access control policy.

Table 59. Domain *x* access control policy definition

Policy	Allowable accesses			
	Secure Priv	Secure User	Nonsecure Priv	Nonsecure User
111	r, w	r, w	r, w	r, w
110	r, w	r, w	r, w	none
101	r, w	r, w	r	r
100	r, w	r, w	r	none
011	r, w	r, w	none	none
010	r, w	none	none	none
001	r	r	none	none
000	none	none	none	none

Unimplemented domains default to the D*x*ACP field value of 000b in the above table, which provides no access rights.

After the D*x*ACP evaluation is complete and the access error signal is generated, the transaction is allowed to continue if it has sufficient access rights, else the access is aborted with the address and attribute information captured in the appropriate error registers.

XRDC optionally supports the inclusion of a hardware semaphore to dynamically alter the access control policy of a memory region or peripheral access. See Hardware semaphores and dynamic access rights for details.

15.5.4 Hardware semaphores and dynamic access rights

Both memory region descriptors and peripheral access control registers support the ability to optionally include a hardware semaphore in their access evaluations. This capability is provided to allow hardware enforcement of dynamic access rights based on the state of the semaphore for shared memory regions and/or shared peripherals.

If enabled (MRGD_W2_*n*[SE] = 1 or PDAC_W0_*n*[SE] = 1), the state of the specified hardware semaphore, as defined by MRGD_W2_*n*[SNUM] or PDAC_W0_*n*[SNUM]), is checked before the normal D*x*ACP evaluation. On a write transaction, if the semaphore is not idle, that is, the semaphore state is non-zero and not owned by the requesting domain, the access is aborted and error-terminated. Stated differently, writes into a semaphore-enabled address space are only allowed if the semaphore is idle, or is owned by the requesting domain. See the Semaphores2 (SEMA42) chapter for more information.

The semaphore check is detailed in lines 84-92 of the C code description contained in MRC C code description. The comparable function in the PAC C code in Peripheral access controller (PAC) is found in lines 38-45.

An additional note on this capability is that because the semaphore module only supports up to 15 domains, that is, DIDs [0-14], DID 15 cannot be used for this function.

15.5.5 Memory region controller (MRC)

The XRDC's memory region controller provides domain-based, hardware access control for all system bus references targeted at non-peripheral memory spaces. Using pre-programmed region descriptors that define memory spaces and their associated access rights per DID, the MRC concurrently monitors system bus transactions and evaluates the appropriateness of each transfer. Memory references that have sufficient access control rights are allowed to complete, while references that are not mapped to any region descriptor or have insufficient rights are terminated with an access error response.

The XRDC architectural framework supports up to 16 MRC instances, where each MRC instance can support {4, 8, 12, 16} region descriptors, concurrently monitoring up to eight slave memory ports. Note that monitoring an AXI protocol port counts as two ports since there are independent read address and write address channels. Conversely, an AHB connection counts as a single port.

A partial, simplified one-dimensional block diagram of an instance of the MRC module is shown in Figure 35. In this figure, a single slave bus port is shown. When the remaining slave ports are considered, the two-dimensional connection matrix of the MRC hardware is the resulting structure. The basic access evaluation macro is shown as the replicated submodule block. Each system bus slave port access is broadcast across the implemented memory region descriptors. In Figure 35, the system bus fabric slave port is shown on the left, the memory region descriptor registers are in the center, and the peripheral bus interface is on the right. The evaluation macro contains two magnitude comparators connected to the start and end address registers from each region descriptor as well as the combinational logic blocks to determine the region hit and access violations. These violations are based on the DID associated with the reference.

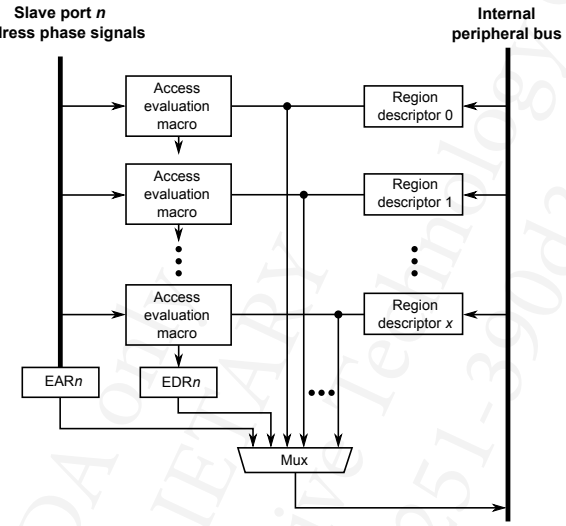


Figure 35. MRC access evaluation macro hardware structure

For each access, the MRC hardware performs a two-step evaluation process. First, the access is compared to all memory region descriptors to determine hit/miss status, and then, using the DID associated with the reference plus the address attributes {write, nonsecure, privileged}, the access rights are examined using the method detailed in DxACP evaluation.

15.5.5.1 Memory region descriptor hit determination

To determine if the current reference hits in a given region, two magnitude comparators are used in conjunction with the region's start and end addresses. The boolean equation for this portion of the hit determination is defined as follows:

```
region_hit_n =
   ((addr >= start_addr) & (addr <= end_addr)) & valid
```

Here, `addr` is the current system reference address, `start_addr]` and `end_addr` are the start and end addresses, respectively, and `valid` is the valid bit, all from memory region descriptor *n*.

You must ensure that end_addr ≥ start_addr. There are no hardware checks to verify this. The software must properly load appropriate values into these fields of the region descriptor.

Also, the region hit determination is based only on an address comparison of the first byte being accessed; that is, the MRC does *not* check the size of the access to make sure it entirely fits within the region.

15.5.5.2 Memory region D*x*ACP evaluation

For each region descriptor hit, the MRC logic then evaluates the access rights defined by the MRGD_W*m_n* registers. Specifically, the domainID attribute selects the appropriate D*x*ACP field from the MRGD_W{2,3}_*n* registers and then the {write, nonsecure,

privileged} address attributes are compared with the access policy defined by the selected D*x*ACP. See DxACP evaluation for additional details on this function.

When CR[GVLD] = 1, and for each bus slave port being monitored, MRC performs a reduction-AND of all the individual *(no_hit | error)* terms from each access evaluation macro. As specified in Memory region descriptor hit determination, an invalid MRGD_W*m_n* (MRGD_W3_*n*[VLD] = 0) forces the region_hit_n evaluation to be negated.

This expression then terminates the bus cycle with an error and reports an access error for three conditions:

- If the access does not hit a region descriptor
- If the access hits a single region descriptor and that region signals a domain violation
- If the access hits multiple (overlapping) regions and all regions signal violations

The third condition reflects that priority is given to *permission granting over access denying* for overlapping regions as this approach provides more flexibility to system software in the region descriptor assignments.

Unimplemented domain identifiers default to all-zero D*x*ACP fields providing no access rights. Additionally, these fields are RAZ/WI.

15.5.5.3 MRC C code description

The functionality of the MRC evaluation is described in the C code below. This code begins with the description of the data structures corresponding to the MRC-associated registers (lines 1-12). Note that the memory region descriptors are described as three-dimensional, unsigned 32-bit integer arrays indexed by the MRC instance number, the RGD number within the instance, and then the word offset within the region descriptor.

The access evaluation consists of four basic functions: a check of the XRDC's global valid bit (lines 43-45), the address hit/miss determination (lines 73-78), the selection of the D*x*ACP field and its evaluation (lines 80-96), and the combination of all the individual MRGD_W*m_n* hit and error indicators to determine the final state of the memory reference (lines 99-134).

```
 1  //  MRC  registers
 2 unsigned int      xrdc_hwcfg0;          // contains NMRC -> # of MRC's
 3 unsigned char     xrdc_mrcfg[16];       // defines NMRGD -> # of RGD's
 4
 5 #define RGD_ADDR_MASK   0xfffffffe0     // force sysAddr to modulo-32
 6 //
 7 // the memory region descriptors are modeled as a 3-dimensional array
 8 //  xrdc_mrgd[x][y][z] where:
 9 //      [x] = MRC instance number defined by xrdc_hwcfg0[nmrc]
10 //      [y] = RGDn within a given instance, defined by xrdc_mrcfgn[nmrgd]
11 //      [z] = Wordm within a given region descriptor
12 unsigned int      xrdc_mrgd[16][16][4]; // memory region descriptors
13
14 evaluate_MRC (MRCn, NMRGD, sysAddr, write, nonsecure, privileged, did)
15   unsigned int    MRCn;                 // MRCn instance number
16   unsigned int    NMRGD;                // number of RGDs for this MRC
17   unsigned int    sysAddr;              // system address
18   unsigned int    write;                // read   (=0), write      (=1)
19   unsigned int    nonsecure;            // secure (=0), nonsecure  (=1)
20   unsigned int    privileged;           // user   (=0), privileged (=1)
21   unsigned int    did;                  // domain identifier
22 {
23   unsigned int    mrgd_vld;             // local RGDn valid
24   unsigned int    mrgd_w2;              // local RGDn Word2
25   unsigned int    mrgd_w3;              // local RGDn Word3
26   unsigned int    mrgd_hit;             // calculated RGD "hit" signal
27   unsigned int    mrgd_error;           // calculated RGD "error" signal
28   unsigned int    mrgd_sema4;           // local copy of RGDn semaphore
29   unsigned int    mrc_hit;              // bitmask of RGDn hit signals
```

```
 30   unsigned int    mrc_error;            // bitmask of RGDn access errors
 31   unsigned int    final_error_state;    // final error state
 32   unsigned int    memory_access_error;  // final access error indicator

 38   unsigned int    i;                    // loop counter
 39
 40   mrc_hit   = 0;
 41   mrc_error = 0;
 42
 43   if (xrdc_cr_gvld == 0) {              // check global valid
 44       memory_access_error = 0;
 45   } else {
 46
 47       for (i = 0; i < xrdc_mrcfg[MRCn]; i++) {
 48           mrgd_hit    = 0;
 49           mrgd_error  = 0;
 50
 51           // extract the region descriptor valid indicator
 52           mrgd_vld = (xrdc_mrgd[MRCn][i][3] & 0x80000000) >> 31;
 53
 73           // first, determine if system address "hits" in the given MRDGn
 74           if (((sysAddr & RGD_ADDR_MASK) >=
 \                    (xrdc_mrgd[MRCn][i][0] & RGD_ADDR_MASK)) &&
 75               ((sysAddr & RGD_ADDR_MASK) <=
 \                    (xrdc_mrgd[MRCn][i][1] & RGD_ADDR_MASK)) &&
 76                (mrgd_vld == 1)) {
 77               mrgd_hit = 1;
 78           }
 79
 80           // second, if there was a hit, evaluate DxACP including any sema4
 81           if (mrgd_hit == 1) {
 82               mrgd_w2 = xrdc_mrgd[MRCn][i][2];
 83               mrgd_w3 = xrdc_mrgd[MRCn][i][3];
 84               // semaphore check using a local copy
 85               mrgd_sema4 = sema4[(mrgd_w2 & 0x3f000000) >> 24];
 86               // check if it is enabled on write and
 87              //       if not idle, nor equal to (did+1)
 88               if (((mrgd_w2 & 0x40000000) == 0x40000000) &&  // se
 89                    (write == 1)                          &&  // wt
 90                    (mrgd_sema4 != 0)                     &&  // sema4 !=idle
 91                    (mrgd_sema4 != (did+1)))              {   // sema4 !=did
 92                   mrgd_error = 1;
 93               } else {
 94                   // perform standard DxACP check
 95                   mrgd_error = evaluate_DxACP(mrgd_w2, mrgd_w3, did,
 \                                    nonsecure, privileged, write);
 96               }
 97           }
 98
 99           // accumulate the hit and error indicators
100           mrc_hit   = (mrc_hit   | (mrgd_hit   << i)) & 0xffff;
101           mrc_error = (mrc_error | (mrgd_error << i)) & 0xffff;
102       }
103
104
105       For each slave port monitored, MRC performs a reduction-AND of all
106       the individual (~hit | error) terms from each access evaluation.
107       This expression then terminates the bus cycle with an error & reports
108       an access error for three conditions:
109
```

```
110       1) If the access does not hit in *any* region descriptor,
111             then an access error is reported.
112       2) If the access hits in one region descriptor and
113          that region signals a violation,
114             then an access error is reported.
115       3) If the access hits in multiple (overlapping) regions and *all*
116          regions signal violations,
117             then an access error is reported.
118
119       The third condition reflects that priority is given to "permission
120       granting over access denying" for overlapping regions as this
121       approach provides more flexibility to system software in region
122       descriptor assignments.
123       */
124
125       final_error_state   = ((mrc_hit   ^ 0xffff)  // form ~mrc_hit
126                           |  mrc_error) & 0xffff;  // factor in error flags
127
128       if (final_error_state == 0xffff)
129           memory_access_error = 1;
130       else
131           memory_access_error = 0;
132    }
133    return memory_access_error;
134 }
```

15.5.6 Peripheral access controller (PAC)

The peripheral access controller performs an access control evaluation similar to that of the memory region controller (Memory region controller (MRC), but using dedicated memory region descriptors with a unique PDAC_W[0-1]_*n* register per slave peripheral slot. The presence of one region descriptor per slave peripheral address space slot in the PAC simplifies the required functionality compared to the MRC.

The functionality of the PAC evaluation is described in the C code shown below. This code begins with the description of the data structures corresponding to the PAC-associated registers (lines 1-3). It then performs a check of the global valid bit (CR[GVLD]) (lines 25-27), followed by the extraction of the peripheral slot number from the system address (line 32), and then followed by the access of the selected PDAC_W[0-1]_*n* registers (lines 33-34). Next, a check of the optional inclusion of a hardware semaphore and its current state is performed (lines 37-46). Finally, if the access is still allowed, an evaluation of the appropriate D*x*ACP permission field is done (line 48). This is detailed in DxACP evaluation and the state of the peripheral access error control signal is returned (line 52).

```
 1   // PAC registers
 2   unsigned int    xrdc_cr_gvld;
 3   unsigned int    xrdc_pdac[512][2];
 4
 5   evaluate_PAC (sysAddr, write, privileged, nonsecure, did)
 6       unsigned int    sysAddr;                  // system address
 7       unsigned int    write;                    // read/write attribute
 8       unsigned int    privileged;               // user/privileged attribute
 9       unsigned int    nonsecure;                // secure/nonsecure attribute
10       unsigned int    did;                      // domain identifier
11   {
12       unsigned int    ips_slot_number;          // f(system address)
13       unsigned int    pdacw0;                   // selected PAC_W0_n register
14       unsigned int    pdacw1;                   // selected PAC_W1_n register
15       unsigned int    pac_sema4;               // local copy of referenced sema4
16       unsigned int    peripheral_access_error;
17
```

```
18      // the evaluation of the PACn involves a 4-step process:
19      //   1) select the appropriate PDACn using the system address
20      //   2) using the DID for the access, select the appropriate DxACP field
21      //   3) evaluate the state of the (optional) SEMA4 for this access
22      //   4) using the {secure/nonsecure, user/priviledged} attributes, eval DxACP
23
24      // first, check the global XRDC valid bit
25      if (xrdc_cr_gvld == 0) {
26          peripheral_access_error = 0;
27          }
28
29      else {
30          // Extract the peripheral slot number from the system memory map.
31          // Example for a standard 4k slot assignment:
32          // sysAddr[20:12] = slot select [0-511]
33          ips_slot_number = (sysAddr & 0x001ff000) >> 12;
34          pdacw0 = xrdc_pdac[ips_slot_number][0];
35          pdacw1 = xrdc_pdac[ips_slot_number][1];
36          peripheral_access_error = 0;
37          if ((pdacw1 & 0x80000000) == 0x80000000) {         // PDACn is valid
38              // semaphore check
39              pac_sema4 = sema4[(pdacw0 & 0x3f000000) >> 24]; // local copy
40              // check if it is enabled on a write and if not idle, nor == (did+1)
41              if (((pdacw0 & 0x40000000) == 0x40000000)   &&  // se
42                   (write == 1)                           &&  // wt
43                   (pac_sema4 != 0)                        &&  // sema4 != idle
44                   (pac_sema4 != (did+1)))                {     // sema4 != did
45                  peripheral_access_error = 1;
46              } else {
47                  // perform the standard DxACP check
48                  peripheral_access_error = evaluate_DxACP(pdacw0, pdacw1, did,
\                                                  nonsecure, privileged, write);
49              }
50          }
51      }
52      return peripheral_access_error;
53  }
```

15.6 Initialization information

Out of reset, the value of CR[GVLD] is 0 and XRDC is disabled, allowing secure privileged startup code to configure the entire programming model. The initialization process typically is performed in three steps:

1. You read the various hardware configuration registers (HWCFG{0-3}, MDACFG*n*, and MRCFG*n*) to obtain the implemented hardware capabilities for the chip.
2. Using the information retrieved in step 1 coupled with the desired domain architecture, you program all the register data structures for domain assignment (MDA_W*m_n*), memory region descriptors (MRGD_W*m_n*), and peripheral access control registers (PDAC_W*m_n*). There are individual valid bits included in all these registers that are typically asserted. Additionally, there are lock mechanisms available if register settings need to be configured and marked as read-only.
3. After the XRDC programming model is loaded, the CR is written with GVLD asserted. At that point, the XRDC is fully operational.

In this context, XRDC being fully operational means the MDAC modules are generating the DIDs and the slave memory and peripheral responders (MRC, PAC) are checking the access rights. Because of the distributed design hierarchy and the pipelined nature of the hardware system bus fabric, it can take multiple cycles for a generated DID to propagate through the entire design. Any previous DID is based on the master's default DID, and depending on the programmed access policies, could generate an access error response.

If incorrect error responses are being generated, the following approaches can be used to minimize or eliminate these extraneous access errors:

- Minimize the amount of system bus traffic when the value of CR[GVLD] is 1.
- Guarantee that all addresses being accessed provide sufficient access rights for the default DID(s) as well as the newly programmed DID(s). After XRDC is fully operational, as confirmed by a read of HWCFG1, the permissions for the default DID(s) can be removed.
- Try to have the bus master that is programming and configuring the XRDC registers use the same DID, that is, its default DID, for both initialization and configuration as well as normal system operation. You do this when you define the DID assignments for the system.

15.7 Application information

As described in Introduction, the XRDC architecture is intended to provide a broad, highly capable framework for access control, system memory protection, and peripheral isolation. The resulting microarchitecture implementation is distributed throughout the core platform, and is highly configurable via hardware design parameters and software programmability. This considerable number of hardware design parameters allows the XRDC architecture to be a very scalable solution, capable of servicing a wide spectrum of chips from ultra-low end MCUs to multi-core, high-performance embedded microprocessors.

15.7.1 Master domain assignments

The typical use case related to master domain assignments is to include one (or more) processor core(s) in a single domain, possibly coupled with other bus master devices such as DMA. The definition of a domain may be static, based on the combination of a processor and other optional bus masters. Alternatively, the processor's operation may be configured to dynamically select between a very small number of domains. The maximum number of possible CPU domain definitions is limited by the MDACFG*n* value. In particular, the optional inclusion of PID value(s) can be used to create multiple classes of CPU with different domain values.

For example, critical tasks, whether safety-critical, performance-critical, and so on, can be grouped together in one domain and all other tasks into the second domain. Non-processor bus masters typically have a single MDA_W*m_n* configuration register associated with them. The domainID and {nonsecure, privileged} attributes are usually statically assigned, unless the module can inherit attributes from a processor programming it (from certain modules such as DMAs).

The MDA_W*m_n* registers, along with the memory region descriptors and the peripheral domain access control registers, have lock features that allow the register resources to be converted into read-only resources to protect the written configuration.

15.7.2 Cache coherency interconnect (CCI)

For multi-core clusters built with Cortex-A cores, the cluster and optional cache coherency interconnect (CCI) logic tags each system bus transaction with a read identifier (RID) or write identifier (WID), depending on the transaction type. The precise identification of the source processor core is complicated by the fact that there is a shared write buffer and transactions can be generated by the snoop control unit in the cluster. These complicating factors remove the ability to precisely associate certain write operations with a specific processor. For this chip, precise source identification of the system bus transaction is therefore limited to the cluster number, an SCU transaction.

Accordingly, there is logic associated with the Master Domain Assignment function that uses the read and write identifiers to dynamically select a domainID using the PID and PID mask functions in the MDA_W*m_n* registers.

For this chip, the {R,W}ID outputs from the core are remapped into a 2-bit encoded ID that is subsequently input to the XRDC MDAC modules to support generation of the domainID.

The following steps detail the hardware operation of using the core {R,W}IDs and the required software configuration to dynamically generate the domainID in the XRDC MDAC logic for the two dual-core Cortex-A53 clusters, cluster0 and cluster1:

1. The hardware remaps the core read and write IDs to the following 2-bit encoded values:

remapped_{r,w}id[1:0]	Description
00b	Signals a cluster0 access
01b	Signals a cluster1 access
10b	Signals an SCU or ACP access

2. The MDAC instances of the core clusters are configured to enable a PID input value, and the remapped 2-bit read and write IDs are connected to the PID_IN[3:0] input signal as follows:

```
PID_IN[3:0] = {remapped_wid[1:0], remapped_rid[1:0]}
```

That is, PID_IN[3:2] represents the remapped write ID and PID_IN[1:0] is the remapped read ID. In addition, PID_IN[4] is used to signal a read (PID_IN[4] = 0) or a write (PID_IN[4] = 1).

Each MDA_W*m_n* register associated with this system bus master port then calculates, on a cycle-by-cycle basis, a dynamic 6-bit PID (PID_dyn) value defined as follows:

```
PID_dyn[5:0] = {PID_REG[5], write, remapped_wid[1:0], remapped_rid[1:0]}
```

Here, PID_REG[5] is the register bit from the associated PID*n* register, and the low-order 5 bits are the concatenation of the write indicator and the remapped write and read identifiers, which are read-only-zero.

3. The dynamic PID_dyn[5:0] value is compared against the MDA_W*m_n*[PIDMDIDB] field, which provides a static PID and PID mask for the comparison. If the combined dynamic PID input matches the MDA_W*m_n*[PIDMDIDB] field, there is a register "hit" and the domainID is generated based on the remaining fields in the MDA_W*m_n* register.
4. From a software configuration perspective, six MDA_W*m_n* registers can be used to generate the required domain IDs: three registers define the domain IDs for reads and three for the ccwrite domain ID. As specific examples, consider the following hex values for these six MDA_W*m_n* registers:

Hex value	Description
MDA_W0_*n* = c000_2c8wh	Domain *w* for cluster0 reads (valid, locked)
MDA_W1_*n* = c001_2c8yh	Domain *y* for cluster1 reads (valid, locked)
MDA_W2_*n* = c002_2c8zh	Domain *z* for SCU reads (valid, locked)
MDA_W3_*n* = c010_238wh	Domain *w* for cluster0 writes (valid, locked)
MDA_W4_*n* = c014_238yh	Domain *y* for cluster1 writes (valid, locked)
MDA_W5_*n* = c018_278zh	Domain *z* for SCU writes (valid, locked)

In this case, the three domainIDs (*w*,*y*,*z*) are arbitrary within the supported [0-7] set.

The above MDA_W*m_n* register definitions can be further refined to utilize the DIDSPA = 2 (DFMT = 0) setting to include the processor number (where possible) as the low-order 2 bits of the domain ID. The lower-order two bits of the input DID to the MDAC instance are configured to generate the value "2" to indicate cp0, and the value "3" to indicate cp1. Then the following domain IDs are generated:

DID[3:0]	Description
2	cluster0, cp0
3	cluster0, cp1
6	cluster1, cp0
7	cluster1, cp1
5	SCU transactions

In this case, domains {*w*, *y*, *z*} = {2, 6, 5}. These values are selected as they align with the default DIDs.

The corresponding MDA_W*m*_*n* register values are specified in the following table:

MDA_W0_*n* = c000_2ca0h	Domains 2/3 for cluster0 {cp0, cp1} read
MDA_W1_*n* = c001_2ca4h	Domains 6/7 for cluster1 {cp0, cp1} read
MDA_W2_*n* = c002_2d85h	Domain 5 for SCU read
MDA_W3_*n* = c010_23a0h	Domains 2/3 for cluster0 cp*x* write
MDA_W4_*n* = c014_23a4h	Domains 6/7 for cluster1 cp*x* write
MDA_W5_*n* = c018_2785h	Domain 5 for SCU write

Finally, it should be noted that the cluster specifically identifies cp0 or cp1 on certain classes of transactions: writes to non-reorderable device memory, store exclusives, and barrier operations have the correct processor identifier. However, the most common processor system bus writes to normal memory or reorderable device memory do not precisely identify the source processor because of the presence of write buffers within the cluster.

15.7.3 Memory region descriptor management

There are two important concepts to consider in managing the memory region descriptors.

First, the association between each MRC*n* instance and the slave memory regions it monitors is chip-specific. This chip-specific configuration specifies the number of implemented memory region descriptors in a given instance and the specific port numbers associated with the slave memories being monitored.

Second, as detailed in Memory region DxACP evaluation, after the XRDC is enabled, every memory reference must hit one or more regions, else an access violation is reported. There are two other conditions for which access errors are reported:

- If the access hits a single region descriptor and that region signals a domain violation
- If the access hits multiple (overlapping) regions and all regions signal violations

15.7.4 Domain error capture management

When either a memory region controller or a peripheral access controller detects a domain access violation, the address and attribute information of the offending access is captured. The DERRLOC*n* read-only registers provide additional information by signaling the instance number of the submodule for which the access violation(s) occurred. Because the resulting exception handler needs the submodule instance to retrieve the captured address and attribute information from DERR_W0_*n* and DERR_W1_*n*, the DERRLOC*n* registers provide the instance number details.

These registers are organized as a word array, indexed by the faulting domain number, with the contents of each register providing a bitmap signaling the instance number(s) associated with all submodules containing captured error information for that domain.

It is recommended that the exception handler begins by reading the HWCFG1 register to determine its domainID. Next, it uses the just-retrieved domainID to index into the DERRLOC*n* array. The resulting DERRLOC*n* value is then examined to determine the instance number of the reporting MRC and/or PAC submodule.

There may be multiple access violations, across multiple instances, pending for a given domain. It is suggested that exception handlers use a "find first one" instruction (alternatively known as "count leading zeroes") to quickly and efficiently find the instance number containing access violation details. After the instance number is determined, the captured error address (DERR_W0_*n*) and attribute information (DERR_W1_*n*) can easily be retrieved from the domain error registers. The 2-bit DERR_W1_*n*[EST] field provides information signaling whether one or more errors have been detected by the submodule instance. A special write to DERR_W3_*n* is required to reset (and rearm) the EST error capture state machine.

This process is repeated until all errors associated with a given domainID have been processed.

A graphical representation of this two-step retrieval of the domain error address and attributes is shown in Figure 36.

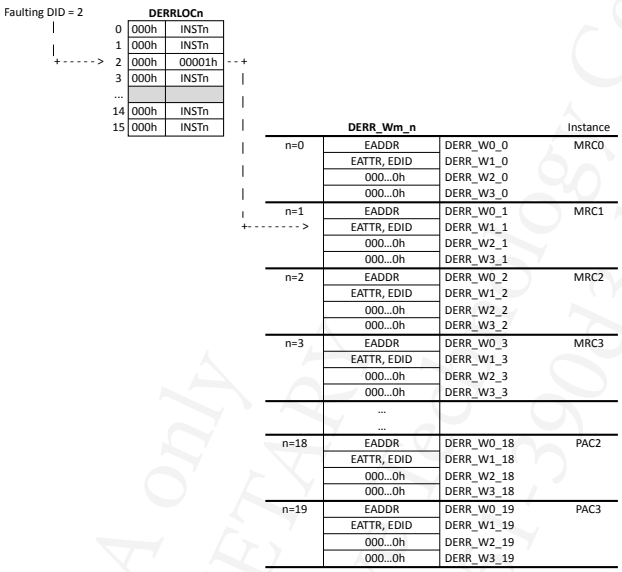


Figure 36. Two-step retrieval of domain error address and attributes

15.8 Glossary

ACP	Access Control Policy
DID	Domain Identifier
MDAC	Master Domain Assignment Controller
MGR	Manager
MRC	Memory Region Controller
PAC	Peripheral Access Controller

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b23f-3901ab624c1

Chapter 16
System Integration Unit Lite2 (SIUL2)

16.1 Chip-specific SIUL2 information

16.1.1 SIUL2 memory map

The SIUL2 memory map is split into two regions, one for SIUL2_0 and one for SIUL2_1. The global memory map space for the SIUL2 modules is specified in the attached IOMUX spreadsheet. That spreadsheet refers to SIUL2_0 and SIUL2_1 as SIUL_CC and SIUL_OFFCC, respectively.

16.1.2 Register configuration

SIUL2_0 is mapped to MSCRs 0–101 and 512–595. SIUL2_1 is mapped to MSCRs 112–190 and 631–1023. This means, for example, that the first SIUL2_1 MSCR register is MSCR112, and that has an offset of 1C0h (112d×4) relative to the MSCR0 address of SIUL2_1. Thus the MSCR112 address is at 400h (1C0h+240h) relative to the SIUL2_1 base address. The attached IOMUX spreadsheet defines the mapping of MSCRs to port pins. This spreadsheet also shows that the MSCR[RCVR], MSCR[DSE], MSCR[SRE], MSCR[INV], MSCR[ODE], and MSCR[OBE] fields are reserved for all GPI pads.These fields on PAD_010, PAD_036, PAD_038, and PAD_043 are writable, but writing any value to those fields has no impact on pad function. MSCR[SLEEP] is reserved for all pads.

NOTE

PAD 10, 36, 38, 43, 146, 148, 150, 152, 154, 156, 158, 160, 162, 164, 166, 168, 170, 172, 174, 176, 179, 182, and 183 on this chip are implemented as GPIs. To know the exact implementation, see the IO Signal Table tab in the attached IOMUX spreadsheet.

NOTE

CR values above 511 in IOMUX spreadsheet refer to IMCR registers. The equivalent IMCR number is CR number minus 512.

16.1.3 Interrupts and DMA requests

SIUL2_0 has no interrupt or DMA requests. SIUL2_1 has a single interrupt that is an OR of all EIRQ fields. After the interrupt has been triggered, the ISR can check the interrupt status flag to find which particular EIRQ pin triggered the interrupt. SIUL2_1 also has eight DMA requests that are mapped to EIRQ[0:7]. The SIUL2_1 DIRSR0 implements only fields EIRE0–EIRE7. Bits 8:31 are reserved and do not correspond to any valid DMA request.

16.1.4 MIDR1 and MIDR2 Reset Values

The reset values for the SIUL2 MCU ID registers (MIDR) will change with the product variant. The following table contains the reset values for each product varient.

Table 60. MIDR reset values for S32G product family

Product Variant	SIUL2_CC_MIDR1	SIUL2_CC_MIDR2	SIUL2_OFF_CC_MIDR 1	SIUL2_OFF_CC_MIDR 2
S32G274A	0x1D12_000X[1]	0x48BB_0000	0x4C20_0412	0x0X12_C000[2]
S32G254A	0x1CFE_000X[1]	0x48BB_0000	0x4C20_0412	0x0XFE_C000[3]
S32G233A	0x1CE9_000X[1]	0x48BB_0000	0x4C20_0411	0x0XE9_C000[3]
S32G234M	0x1CEA_000X[1]	0x48B8_0000	0x4C20_3412	0x0XEA_C000[3]

1. X= 2-bit MAJOR_MASK bit field and 2-bit MINOR_MASK bit field which will change depending upon part revision.

2. X=4'bnn01, where n is the 2-bit SUBMINOR_MASK field which will change depending upon part revision.
3. X=4'bnn00, where n is the 2-bit SUBMINOR_MASK field which will change depending upon part revision.

NOTE

The part revision information is contained in the following bit fields:

- SIUL2_0_MIDR1[MAJOR_MASK]
- SIUL2_0_MIDR1[MINOR_MASK]
- SIUL2_1_MIDR2[SUBMINOR_MASK]

The revision information follows the order: Revision MAJOR.MINOR.SUBMINOR

16.2 Introduction

16.2.1 Overview

SIUL2 provides control over all electrical pin controls and ports with 16 bits of bidirectional, general-purpose input and output signals. SIUL2 enables you to select the functions and electrical characteristics that appear on external chip pins. It also controls the multiplexing of internal signals from one module to another and controls chip I/O. It supports as many as 32 external interrupts with trigger event configuration.

The block diagram shows the functional blocks of the module and its interfaces to other system components.

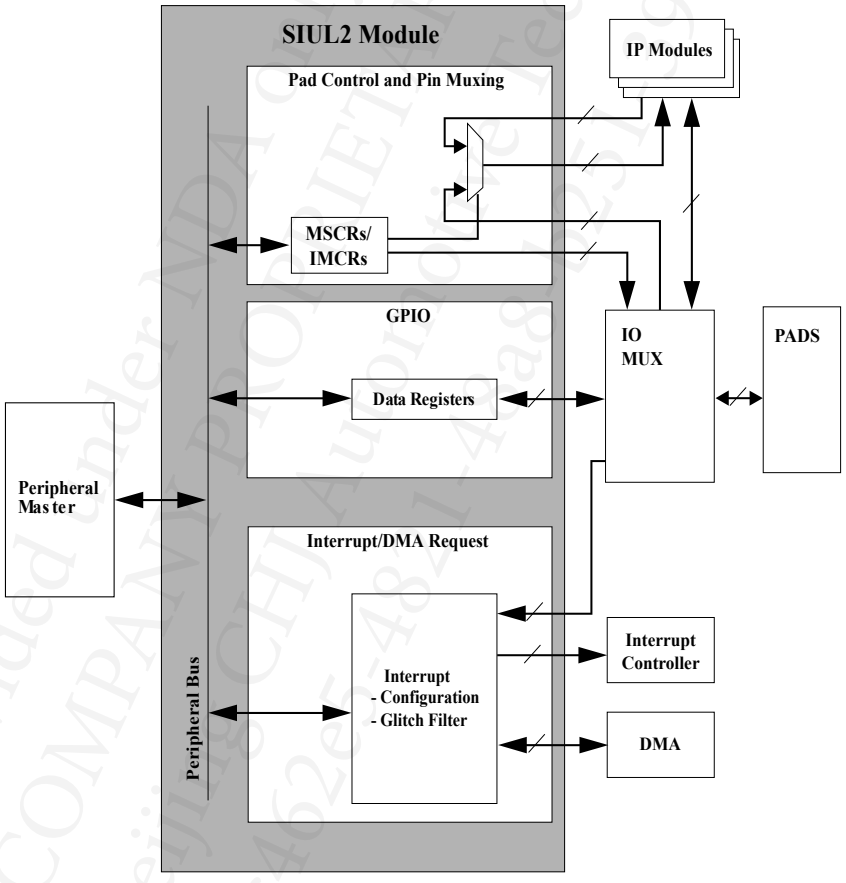


Figure 37. SIUL2 block diagram

SIUL2 provides dedicated pad control to general-purpose pads that can be configured as either inputs or outputs. It provides registers for you to read values from GPIO pads configured as inputs and to write values to GPIO pads configured as outputs:

- When configured as output, you can write to an internal register to control the state driven on the associated output pad.
- When configured as input, you can detect the state of the associated pad by reading the value from an internal register.
- When configured as input and output, the pad value can be read back to check if the written value appeared on the pad.

You can access GPIO data registers in various ways to allow port access and bit manipulation without read-modify-write operations:

- Access to two 16-bit ports in one access
- Read/write access to a single bit
- A 16-bit port write with a bit mask using single 32-bit access

You can configure external interrupt sources at the chip level to be used with any chip pad. You can configure interrupt sources to have a digital filter to reject short glitches on the inputs. The external interrupt/DMA requests map to the REQ pins in the chip packages.

16.2.2 Features

SIUL2 supports:

- One to 32 GPIO ports with data control
  - Drives data to as many as 16 independent I/O channels
  - Samples data from as many as 16 independent I/O channels
- Read or write of two 16-bit registers with one access for a 32-bit port
- External interrupt/DMA requests:
  - One to 12 programmable digital glitch filters, one for each REQ pin
  - Edge detection
- Multiplexed Signal Configuration Registers (MSCR) to configure the electrical parameters and settings for as many as 512 functional pads.

See the interrupt map file and the DMAMUX map file attached to this document for mapping of interrupt and DMA sources to interrupt vectors and DMA channels.

16.3 SIUL2_0 register descriptions

This section describes the SIUL2 registers.

NOTE

- Undocumented register spaces in the SIUL2 memory map, including addresses shown as blanks, are Reserved.
  - Reserved registers or spaces are read as 0.
  - Writes to Reserved registers or spaces generate a transfer error.
- Writes to read-only registers generate a transfer error.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-b251-390d3ab62fc1

NOTE

- For the array of 8-bit registers GPDO*n* and GPDI*n*:
  - An 8-bit access to an unimplemented address (a "hole") within the array region will generate a transfer error.
  - However, if you do a 16-bit or a 32-bit access and if any register instance is implemented within the accessed range, a transfer error will not be generated even if the range includes a hole.
- For the array of 16-bit registers PGPDO*n* and PGPDI*n*:
  - A 16-bit access to an unimplemented address (a "hole") within the array region will generate a transfer error.
  - However, if you do a 32-bit access and if a register instance is implemented on the other 16-bit range, a transfer error will not be generated even if the range includes a hole.

16.3.1 SIUL2_0 memory map

SIUL2_0 base address: 4009_C000h

Offset	Register	Width (In bits)	Access	Reset value
4h	SIUL2 MCU ID Register #1 (MIDR1)	32	RO	1D12_0000h
8h	SIUL2 MCU ID Register #2 (MIDR2)	32	RO	48BB_0000h
10h	SIUL2 DMA/Interrupt Status Flag Register0 (DISR0)	32	W1C	0000_0000h
18h	SIUL2 DMA/Interrupt Request Enable Register0 (DIRER0)	32	RW	0000_0000h
20h	SIUL2 DMA/Interrupt Request Select Register0 (DIRSR0)	32	RW	0000_0000h
28h	SIUL2 Interrupt Rising-Edge Event Enable Register 0 (IREER0)	32	RW	0000_0000h
30h	SIUL2 Interrupt Falling-Edge Event Enable Register 0 (IFEER0)	32	RW	0000_0000h
38h	SIUL2 Interrupt Filter Enable Register 0 (IFER0)	32	RW	0000_0000h
40h - 6Ch	SIUL2 Interrupt Filter Maximum Counter Register (IFMCR0 - IFMCR11)	32	RW	0000_0000h
C0h	SIUL2 Interrupt Filter Clock Prescaler Register (IFCPR)	32	RW	0000_0000h
240h - 3D4h	SIUL2 Multiplexed Signal Configuration Register (MSCR0 - MSCR101)	32	RW	See description
A40h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR0)	32	RW	0000_0000h
A44h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR1)	32	RW	0000_0000h
A4Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR3)	32	RW	0000_0000h
A50h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR4)	32	RW	0000_0000h
A54h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR5)	32	RW	0000_0000h
A58h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR6)	32	RW	0000_0000h
A5Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR7)	32	RW	0000_0000h
A60h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR8)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A64h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR9)	32	RW	0000_0000h
A68h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR10)	32	RW	0000_0000h
A6Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR11)	32	RW	0000_0000h
A70h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR12)	32	RW	0000_0000h
A74h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR13)	32	RW	0000_0000h
A78h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR14)	32	RW	0000_0000h
A7Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR15)	32	RW	0000_0000h
A80h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR16)	32	RW	0000_0000h
A84h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR17)	32	RW	0000_0000h
A88h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR18)	32	RW	0000_0000h
A8Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR19)	32	RW	0000_0000h
A90h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR20)	32	RW	0000_0000h
A94h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR21)	32	RW	0000_0000h
A98h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR22)	32	RW	0000_0000h
A9Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR23)	32	RW	0000_0000h
AA0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR24)	32	RW	0000_0000h
AA4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR25)	32	RW	0000_0000h
AA8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR26)	32	RW	0000_0000h
AACh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR27)	32	RW	0000_0000h
AB0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR28)	32	RW	0000_0000h
AB4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR29)	32	RW	0000_0000h
AB8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR30)	32	RW	0000_0000h
ABCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR31)	32	RW	0000_0000h
AC0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR32)	32	RW	0000_0000h
AC4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR33)	32	RW	0000_0000h
AC8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR34)	32	RW	0000_0000h
ACCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR35)	32	RW	0000_0000h
AD0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR36)	32	RW	0000_0000h
AD4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR37)	32	RW	0000_0000h
AD8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR38)	32	RW	0000_0000h
ADCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR39)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
AE0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR40)	32	RW	0000_0000h
AE4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR41)	32	RW	0000_0000h
AE8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR42)	32	RW	0000_0000h
AECh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR43)	32	RW	0000_0000h
AF0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR44)	32	RW	0000_0000h
AF4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR45)	32	RW	0000_0000h
AF8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR46)	32	RW	0000_0000h
AFCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR47)	32	RW	0000_0000h
B00h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR48)	32	RW	0000_0000h
B04h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR49)	32	RW	0000_0000h
B08h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR50)	32	RW	0000_0000h
B0Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR51)	32	RW	0000_0000h
B10h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR52)	32	RW	0000_0000h
B14h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR53)	32	RW	0000_0000h
B18h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR54)	32	RW	0000_0000h
B1Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR55)	32	RW	0000_0000h
B20h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR56)	32	RW	0000_0000h
B24h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR57)	32	RW	0000_0000h
B28h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR58)	32	RW	0000_0000h
B2Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR59)	32	RW	0000_0000h
B30h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR60)	32	RW	0000_0000h
B34h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR61)	32	RW	0000_0000h
B50h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR68)	32	RW	0000_0000h
B54h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR69)	32	RW	0000_0000h
B58h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR70)	32	RW	0000_0000h
B5Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR71)	32	RW	0000_0000h
B60h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR72)	32	RW	0000_0000h
B64h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR73)	32	RW	0000_0000h
B68h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR74)	32	RW	0000_0000h
B6Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR75)	32	RW	0000_0000h
B70h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR76)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
B74h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR77)	32	RW	0000_0000h
B78h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR78)	32	RW	0000_0000h
B7Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR79)	32	RW	0000_0000h
B80h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR80)	32	RW	0000_0000h
B84h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR81)	32	RW	0000_0000h
B88h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR82)	32	RW	0000_0000h
B8Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR83)	32	RW	0000_0000h
1300h	SIUL2 GPIO Pad Data Output Register (GPDO3)	8	RW	00h
1301h	SIUL2 GPIO Pad Data Output Register (GPDO2)	8	RW	00h
1302h	SIUL2 GPIO Pad Data Output Register (GPDO1)	8	RW	00h
1303h	SIUL2 GPIO Pad Data Output Register (GPDO0)	8	RW	00h
1304h	SIUL2 GPIO Pad Data Output Register (GPDO7)	8	RW	00h
1305h	SIUL2 GPIO Pad Data Output Register (GPDO6)	8	RW	00h
1306h	SIUL2 GPIO Pad Data Output Register (GPDO5)	8	RW	00h
1307h	SIUL2 GPIO Pad Data Output Register (GPDO4)	8	RW	00h
1308h	SIUL2 GPIO Pad Data Output Register (GPDO11)	8	RW	00h
1309h	SIUL2 GPIO Pad Data Output Register (GPDO10)	8	RW	00h
130Ah	SIUL2 GPIO Pad Data Output Register (GPDO9)	8	RW	00h
130Bh	SIUL2 GPIO Pad Data Output Register (GPDO8)	8	RW	00h
130Ch	SIUL2 GPIO Pad Data Output Register (GPDO15)	8	RW	00h
130Dh	SIUL2 GPIO Pad Data Output Register (GPDO14)	8	RW	00h
130Eh	SIUL2 GPIO Pad Data Output Register (GPDO13)	8	RW	00h
130Fh	SIUL2 GPIO Pad Data Output Register (GPDO12)	8	RW	00h
1310h	SIUL2 GPIO Pad Data Output Register (GPDO19)	8	RW	00h
1311h	SIUL2 GPIO Pad Data Output Register (GPDO18)	8	RW	00h
1312h	SIUL2 GPIO Pad Data Output Register (GPDO17)	8	RW	00h
1313h	SIUL2 GPIO Pad Data Output Register (GPDO16)	8	RW	00h
1314h	SIUL2 GPIO Pad Data Output Register (GPDO23)	8	RW	00h
1315h	SIUL2 GPIO Pad Data Output Register (GPDO22)	8	RW	00h
1316h	SIUL2 GPIO Pad Data Output Register (GPDO21)	8	RW	00h
1317h	SIUL2 GPIO Pad Data Output Register (GPDO20)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1318h	SIUL2 GPIO Pad Data Output Register (GPDO27)	8	RW	00h
1319h	SIUL2 GPIO Pad Data Output Register (GPDO26)	8	RW	00h
131Ah	SIUL2 GPIO Pad Data Output Register (GPDO25)	8	RW	00h
131Bh	SIUL2 GPIO Pad Data Output Register (GPDO24)	8	RW	00h
131Ch	SIUL2 GPIO Pad Data Output Register (GPDO31)	8	RW	00h
131Dh	SIUL2 GPIO Pad Data Output Register (GPDO30)	8	RW	00h
131Eh	SIUL2 GPIO Pad Data Output Register (GPDO29)	8	RW	00h
131Fh	SIUL2 GPIO Pad Data Output Register (GPDO28)	8	RW	00h
1320h	SIUL2 GPIO Pad Data Output Register (GPDO35)	8	RW	00h
1321h	SIUL2 GPIO Pad Data Output Register (GPDO34)	8	RW	00h
1322h	SIUL2 GPIO Pad Data Output Register (GPDO33)	8	RW	00h
1323h	SIUL2 GPIO Pad Data Output Register (GPDO32)	8	RW	00h
1324h	SIUL2 GPIO Pad Data Output Register (GPDO39)	8	RW	00h
1325h	SIUL2 GPIO Pad Data Output Register (GPDO38)	8	RW	00h
1326h	SIUL2 GPIO Pad Data Output Register (GPDO37)	8	RW	00h
1327h	SIUL2 GPIO Pad Data Output Register (GPDO36)	8	RW	00h
1328h	SIUL2 GPIO Pad Data Output Register (GPDO43)	8	RW	00h
1329h	SIUL2 GPIO Pad Data Output Register (GPDO42)	8	RW	00h
132Ah	SIUL2 GPIO Pad Data Output Register (GPDO41)	8	RW	00h
132Bh	SIUL2 GPIO Pad Data Output Register (GPDO40)	8	RW	00h
132Ch	SIUL2 GPIO Pad Data Output Register (GPDO47)	8	RW	00h
132Dh	SIUL2 GPIO Pad Data Output Register (GPDO46)	8	RW	00h
132Eh	SIUL2 GPIO Pad Data Output Register (GPDO45)	8	RW	00h
132Fh	SIUL2 GPIO Pad Data Output Register (GPDO44)	8	RW	00h
1330h	SIUL2 GPIO Pad Data Output Register (GPDO51)	8	RW	00h
1331h	SIUL2 GPIO Pad Data Output Register (GPDO50)	8	RW	00h
1332h	SIUL2 GPIO Pad Data Output Register (GPDO49)	8	RW	00h
1333h	SIUL2 GPIO Pad Data Output Register (GPDO48)	8	RW	00h
1334h	SIUL2 GPIO Pad Data Output Register (GPDO55)	8	RW	00h
1335h	SIUL2 GPIO Pad Data Output Register (GPDO54)	8	RW	00h
1336h	SIUL2 GPIO Pad Data Output Register (GPDO53)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1337h	SIUL2 GPIO Pad Data Output Register (GPDO52)	8	RW	00h
1338h	SIUL2 GPIO Pad Data Output Register (GPDO59)	8	RW	00h
1339h	SIUL2 GPIO Pad Data Output Register (GPDO58)	8	RW	00h
133Ah	SIUL2 GPIO Pad Data Output Register (GPDO57)	8	RW	00h
133Bh	SIUL2 GPIO Pad Data Output Register (GPDO56)	8	RW	00h
133Ch	SIUL2 GPIO Pad Data Output Register (GPDO63)	8	RW	00h
133Dh	SIUL2 GPIO Pad Data Output Register (GPDO62)	8	RW	00h
133Eh	SIUL2 GPIO Pad Data Output Register (GPDO61)	8	RW	00h
133Fh	SIUL2 GPIO Pad Data Output Register (GPDO60)	8	RW	00h
1340h	SIUL2 GPIO Pad Data Output Register (GPDO67)	8	RW	00h
1341h	SIUL2 GPIO Pad Data Output Register (GPDO66)	8	RW	00h
1342h	SIUL2 GPIO Pad Data Output Register (GPDO65)	8	RW	00h
1343h	SIUL2 GPIO Pad Data Output Register (GPDO64)	8	RW	00h
1344h	SIUL2 GPIO Pad Data Output Register (GPDO71)	8	RW	00h
1345h	SIUL2 GPIO Pad Data Output Register (GPDO70)	8	RW	00h
1346h	SIUL2 GPIO Pad Data Output Register (GPDO69)	8	RW	00h
1347h	SIUL2 GPIO Pad Data Output Register (GPDO68)	8	RW	00h
1348h	SIUL2 GPIO Pad Data Output Register (GPDO75)	8	RW	00h
1349h	SIUL2 GPIO Pad Data Output Register (GPDO74)	8	RW	00h
134Ah	SIUL2 GPIO Pad Data Output Register (GPDO73)	8	RW	00h
134Bh	SIUL2 GPIO Pad Data Output Register (GPDO72)	8	RW	00h
134Ch	SIUL2 GPIO Pad Data Output Register (GPDO79)	8	RW	00h
134Dh	SIUL2 GPIO Pad Data Output Register (GPDO78)	8	RW	00h
134Eh	SIUL2 GPIO Pad Data Output Register (GPDO77)	8	RW	00h
134Fh	SIUL2 GPIO Pad Data Output Register (GPDO76)	8	RW	00h
1350h	SIUL2 GPIO Pad Data Output Register (GPDO83)	8	RW	00h
1351h	SIUL2 GPIO Pad Data Output Register (GPDO82)	8	RW	00h
1352h	SIUL2 GPIO Pad Data Output Register (GPDO81)	8	RW	00h
1353h	SIUL2 GPIO Pad Data Output Register (GPDO80)	8	RW	00h
1354h	SIUL2 GPIO Pad Data Output Register (GPDO87)	8	RW	00h
1355h	SIUL2 GPIO Pad Data Output Register (GPDO86)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1356h	SIUL2 GPIO Pad Data Output Register (GPDO85)	8	RW	00h
1357h	SIUL2 GPIO Pad Data Output Register (GPDO84)	8	RW	00h
1358h	SIUL2 GPIO Pad Data Output Register (GPDO91)	8	RW	00h
1359h	SIUL2 GPIO Pad Data Output Register (GPDO90)	8	RW	00h
135Ah	SIUL2 GPIO Pad Data Output Register (GPDO89)	8	RW	00h
135Bh	SIUL2 GPIO Pad Data Output Register (GPDO88)	8	RW	00h
135Ch	SIUL2 GPIO Pad Data Output Register (GPDO95)	8	RW	00h
135Dh	SIUL2 GPIO Pad Data Output Register (GPDO94)	8	RW	00h
135Eh	SIUL2 GPIO Pad Data Output Register (GPDO93)	8	RW	00h
135Fh	SIUL2 GPIO Pad Data Output Register (GPDO92)	8	RW	00h
1360h	SIUL2 GPIO Pad Data Output Register (GPDO99)	8	RW	00h
1361h	SIUL2 GPIO Pad Data Output Register (GPDO98)	8	RW	00h
1362h	SIUL2 GPIO Pad Data Output Register (GPDO97)	8	RW	00h
1363h	SIUL2 GPIO Pad Data Output Register (GPDO96)	8	RW	00h
1366h	SIUL2 GPIO Pad Data Output Register (GPDO101)	8	RW	00h
1367h	SIUL2 GPIO Pad Data Output Register (GPDO100)	8	RW	00h
1500h	SIUL2 GPIO Pad Data Input Register (GPDI3)	8	RO	00h
1501h	SIUL2 GPIO Pad Data Input Register (GPDI2)	8	RO	00h
1502h	SIUL2 GPIO Pad Data Input Register (GPDI1)	8	RO	00h
1503h	SIUL2 GPIO Pad Data Input Register (GPDI0)	8	RO	01h
1504h	SIUL2 GPIO Pad Data Input Register (GPDI7)	8	RO	00h
1505h	SIUL2 GPIO Pad Data Input Register (GPDI6)	8	RO	00h
1506h	SIUL2 GPIO Pad Data Input Register (GPDI5)	8	RO	01h
1507h	SIUL2 GPIO Pad Data Input Register (GPDI4)	8	RO	00h
1508h	SIUL2 GPIO Pad Data Input Register (GPDI11)	8	RO	00h
1509h	SIUL2 GPIO Pad Data Input Register (GPDI10)	8	RO	00h
150Ah	SIUL2 GPIO Pad Data Input Register (GPDI9)	8	RO	00h
150Bh	SIUL2 GPIO Pad Data Input Register (GPDI8)	8	RO	00h
150Ch	SIUL2 GPIO Pad Data Input Register (GPDI15)	8	RO	00h
150Dh	SIUL2 GPIO Pad Data Input Register (GPDI14)	8	RO	00h
150Eh	SIUL2 GPIO Pad Data Input Register (GPDI13)	8	RO	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
150Fh	SIUL2 GPIO Pad Data Input Register (GPDI12)	8	RO	00h
1510h	SIUL2 GPIO Pad Data Input Register (GPDI19)	8	RO	00h
1511h	SIUL2 GPIO Pad Data Input Register (GPDI18)	8	RO	00h
1512h	SIUL2 GPIO Pad Data Input Register (GPDI17)	8	RO	00h
1513h	SIUL2 GPIO Pad Data Input Register (GPDI16)	8	RO	00h
1514h	SIUL2 GPIO Pad Data Input Register (GPDI23)	8	RO	00h
1515h	SIUL2 GPIO Pad Data Input Register (GPDI22)	8	RO	00h
1516h	SIUL2 GPIO Pad Data Input Register (GPDI21)	8	RO	00h
1517h	SIUL2 GPIO Pad Data Input Register (GPDI20)	8	RO	00h
1518h	SIUL2 GPIO Pad Data Input Register (GPDI27)	8	RO	00h
1519h	SIUL2 GPIO Pad Data Input Register (GPDI26)	8	RO	00h
151Ah	SIUL2 GPIO Pad Data Input Register (GPDI25)	8	RO	00h
151Bh	SIUL2 GPIO Pad Data Input Register (GPDI24)	8	RO	00h
151Ch	SIUL2 GPIO Pad Data Input Register (GPDI31)	8	RO	00h
151Dh	SIUL2 GPIO Pad Data Input Register (GPDI30)	8	RO	00h
151Eh	SIUL2 GPIO Pad Data Input Register (GPDI29)	8	RO	00h
151Fh	SIUL2 GPIO Pad Data Input Register (GPDI28)	8	RO	00h
1520h	SIUL2 GPIO Pad Data Input Register (GPDI35)	8	RO	00h
1521h	SIUL2 GPIO Pad Data Input Register (GPDI34)	8	RO	00h
1522h	SIUL2 GPIO Pad Data Input Register (GPDI33)	8	RO	00h
1523h	SIUL2 GPIO Pad Data Input Register (GPDI32)	8	RO	00h
1524h	SIUL2 GPIO Pad Data Input Register (GPDI39)	8	RO	00h
1525h	SIUL2 GPIO Pad Data Input Register (GPDI38)	8	RO	00h
1526h	SIUL2 GPIO Pad Data Input Register (GPDI37)	8	RO	00h
1527h	SIUL2 GPIO Pad Data Input Register (GPDI36)	8	RO	00h
1528h	SIUL2 GPIO Pad Data Input Register (GPDI43)	8	RO	00h
1529h	SIUL2 GPIO Pad Data Input Register (GPDI42)	8	RO	00h
152Ah	SIUL2 GPIO Pad Data Input Register (GPDI41)	8	RO	00h
152Bh	SIUL2 GPIO Pad Data Input Register (GPDI40)	8	RO	00h
152Ch	SIUL2 GPIO Pad Data Input Register (GPDI47)	8	RO	00h
152Dh	SIUL2 GPIO Pad Data Input Register (GPDI46)	8	RO	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
152Eh	SIUL2 GPIO Pad Data Input Register (GPDI45)	8	RO	00h
152Fh	SIUL2 GPIO Pad Data Input Register (GPDI44)	8	RO	00h
1530h	SIUL2 GPIO Pad Data Input Register (GPDI51)	8	RO	00h
1531h	SIUL2 GPIO Pad Data Input Register (GPDI50)	8	RO	00h
1532h	SIUL2 GPIO Pad Data Input Register (GPDI49)	8	RO	00h
1533h	SIUL2 GPIO Pad Data Input Register (GPDI48)	8	RO	00h
1534h	SIUL2 GPIO Pad Data Input Register (GPDI55)	8	RO	00h
1535h	SIUL2 GPIO Pad Data Input Register (GPDI54)	8	RO	00h
1536h	SIUL2 GPIO Pad Data Input Register (GPDI53)	8	RO	00h
1537h	SIUL2 GPIO Pad Data Input Register (GPDI52)	8	RO	00h
1538h	SIUL2 GPIO Pad Data Input Register (GPDI59)	8	RO	00h
1539h	SIUL2 GPIO Pad Data Input Register (GPDI58)	8	RO	00h
153Ah	SIUL2 GPIO Pad Data Input Register (GPDI57)	8	RO	00h
153Bh	SIUL2 GPIO Pad Data Input Register (GPDI56)	8	RO	00h
153Ch	SIUL2 GPIO Pad Data Input Register (GPDI63)	8	RO	00h
153Dh	SIUL2 GPIO Pad Data Input Register (GPDI62)	8	RO	00h
153Eh	SIUL2 GPIO Pad Data Input Register (GPDI61)	8	RO	00h
153Fh	SIUL2 GPIO Pad Data Input Register (GPDI60)	8	RO	00h
1540h	SIUL2 GPIO Pad Data Input Register (GPDI67)	8	RO	00h
1541h	SIUL2 GPIO Pad Data Input Register (GPDI66)	8	RO	00h
1542h	SIUL2 GPIO Pad Data Input Register (GPDI65)	8	RO	00h
1543h	SIUL2 GPIO Pad Data Input Register (GPDI64)	8	RO	00h
1544h	SIUL2 GPIO Pad Data Input Register (GPDI71)	8	RO	00h
1545h	SIUL2 GPIO Pad Data Input Register (GPDI70)	8	RO	00h
1546h	SIUL2 GPIO Pad Data Input Register (GPDI69)	8	RO	00h
1547h	SIUL2 GPIO Pad Data Input Register (GPDI68)	8	RO	00h
1548h	SIUL2 GPIO Pad Data Input Register (GPDI75)	8	RO	00h
1549h	SIUL2 GPIO Pad Data Input Register (GPDI74)	8	RO	00h
154Ah	SIUL2 GPIO Pad Data Input Register (GPDI73)	8	RO	00h
154Bh	SIUL2 GPIO Pad Data Input Register (GPDI72)	8	RO	00h
154Ch	SIUL2 GPIO Pad Data Input Register (GPDI79)	8	RO	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
154Dh	SIUL2 GPIO Pad Data Input Register (GPDI78)	8	RO	00h
154Eh	SIUL2 GPIO Pad Data Input Register (GPDI77)	8	RO	00h
154Fh	SIUL2 GPIO Pad Data Input Register (GPDI76)	8	RO	00h
1550h	SIUL2 GPIO Pad Data Input Register (GPDI83)	8	RO	00h
1551h	SIUL2 GPIO Pad Data Input Register (GPDI82)	8	RO	00h
1552h	SIUL2 GPIO Pad Data Input Register (GPDI81)	8	RO	00h
1553h	SIUL2 GPIO Pad Data Input Register (GPDI80)	8	RO	00h
1554h	SIUL2 GPIO Pad Data Input Register (GPDI87)	8	RO	00h
1555h	SIUL2 GPIO Pad Data Input Register (GPDI86)	8	RO	00h
1556h	SIUL2 GPIO Pad Data Input Register (GPDI85)	8	RO	00h
1557h	SIUL2 GPIO Pad Data Input Register (GPDI84)	8	RO	00h
1558h	SIUL2 GPIO Pad Data Input Register (GPDI91)	8	RO	00h
1559h	SIUL2 GPIO Pad Data Input Register (GPDI90)	8	RO	00h
155Ah	SIUL2 GPIO Pad Data Input Register (GPDI89)	8	RO	00h
155Bh	SIUL2 GPIO Pad Data Input Register (GPDI88)	8	RO	00h
155Ch	SIUL2 GPIO Pad Data Input Register (GPDI95)	8	RO	00h
155Dh	SIUL2 GPIO Pad Data Input Register (GPDI94)	8	RO	00h
155Eh	SIUL2 GPIO Pad Data Input Register (GPDI93)	8	RO	00h
155Fh	SIUL2 GPIO Pad Data Input Register (GPDI92)	8	RO	00h
1560h	SIUL2 GPIO Pad Data Input Register (GPDI99)	8	RO	00h
1561h	SIUL2 GPIO Pad Data Input Register (GPDI98)	8	RO	00h
1562h	SIUL2 GPIO Pad Data Input Register (GPDI97)	8	RO	00h
1563h	SIUL2 GPIO Pad Data Input Register (GPDI96)	8	RO	00h
1566h	SIUL2 GPIO Pad Data Input Register (GPDI101)	8	RO	00h
1567h	SIUL2 GPIO Pad Data Input Register (GPDI100)	8	RO	00h
1700h - 170Ah	SIUL2 Parallel GPIO Pad Data Out Register (PGPDO0 - PGPDO5)	16	RW	0000h
170Eh	SIUL2 Parallel GPIO Pad Data Out Register (PGPDO6)	16	RW	0000h
1740h - 174Ah	SIUL2 Parallel GPIO Pad Data In Register (PGPDI0 - PGPDI5)	16	RO	0000h
174Eh	SIUL2 Parallel GPIO Pad Data In Register (PGPDI6)	16	RO	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1780h - 1794h	SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO0 - MPGPDO5)	32	WORZ	0000_0000h
1798h	SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO6)	32	WORZ	0000_0000h

16.3.2 SIUL2 MCU ID Register #1 (MIDR1)

Offset

Register	Offset
MIDR1	4h

Function

This register holds identification information about the device.

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PRODUCT_LINE_LETTER						PART_NO									
W																
Reset	0	0	0	1	1	1	0	1	0	0	0	1	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CC_REVISION								MAJOR_MASK				MINOR_MASK			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 PRODUCT_LINE_LETTER	Product Line Letter Identified the ASCII character in MCU Part Number. This field specifies the part number suffix, and needs to be combined with MIDR1[PART_NO] to provide the full chip number. 0x01 A This value is set at the factory and cannot be changed. All other values are reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-16 PART_NO	MCU Part Number
15-8 CC_REVISION	CC Revision Revision number of common base architecture.
7-4 MAJOR_MASK	Major Mask Revision 0000b - Rev 1.x 0001b - Rev 2.x
3-0 MINOR_MASK	Minor Mask Revision 0000b - Rev x.0 0001b - Rev x.1

16.3.3 SIUL2 MCU ID Register #2 (MIDR2)

Offset

Register	Offset
MIDR2	8h

Function

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TECHNOLOGY			TEMPERATURE			PACKAGE						FREQUENCY			
W																
Reset	0	1	0	0	1	0	0	0	1	0	1	1	1	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FLASH_CODE		FLASH_DATA		FLASH_SIZE_DATA				FLASH_SIZE_CODE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHD Automotive Technology Co., Ltd.
c8c46205-4821-4888-b251-390d3ab624c1

Fields

Field	Function
31-29 TECHNOLOGY	Technology Identifies the silicon technology. 000b - C55FC 001b - C40EFS3 010b - 16FFC
28-26 TEMPERATURE	Temperature Identifies the ambient temperature range. 000b - C = 85C 010b - V = 105C 100b - M = 125C
25-20 PACKAGE	Package This field can by read by software to determine the package type that is used for the particular device: 001011b - 525-ball MAPBGA
19-16 FREQUENCY	Frequency Identifies maximum core frequency. Qualified by Product Line Letter to provide wider range of frequencies. 0001b - 64 MHz 0010b - 80 MHz 0011b - 120 MHz 0100b - 160 MHz 0101b - 240 MHz 0110b - 320 MHz 0111b - 340 MHz 1000b - 400 MHz 1001b - 600 MHz 1010b - 800 MHz 1011b - 1000 MHz 1100b - 1200 MHz
15-14 FLASH_CODE	Flash Code Identifies the location of Code Flash, if any, within the package. 00b - None 01b - SiP

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Monolithic
13-12 FLASH_DATA	Flash Data Identifies the location of Data Flash, if any, within the package. 00b - None 01b - AE 10b - Monolithic 11b - Configuralbe within Code Flash
11-8 FLASH_SIZE_DATA	Flash Size Data Identifies the Flash (EE) memory size. 0000b - 0KB 0001b - 128KB 0010b - 256KB 0011b - 384KB 0100b - 512KB 0101b - 640KB 0110b - 768KB 0111b - 896KB 1000b - 1024KB 1001b - 1152KB 1010b - 1280KB 1011b - 1408KB 1100b - 1536KB 1101b - 1664KB 1110b - 1792KB 1111b - 1920KB
7-0 FLASH_SIZE_CODE	Flash Size Code Identifies the Flash (code) memory size. Upper 6-Bit field defines Major size, Lower 2 bits define 0.25 MB intervals up to 32 MB, then 0.5 MB intervals up to 64 MB, then 4MB intervals. 00000000b - None 00000001b - 256kB 00000010b - 512kB 00000011b - 768kB

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00000100b - 1MB 00000101b - 1.25MB 00000110b - 1.50MB 00000111b - 1.75MB 00001000b - 2.00MB 00001100b - 3.00MB 00010000b - 4.00MB 00100000b - 8.00MB 01000000b - 16.00MB 10000000b - 32.00MB 10000100b - 34.00MB 10001000b - 36.00MB 11000000b - 64.00MB 11000100b - 80.00MB 11001000b - 96.00MB 11110100b - 272.00MB 11110101b - 280MB 11110110b - 300MB 11110111b - 320MB 11111000b - 360MB 11111001b - 400MB 11111010b - Reserved 11111011b - 512MB 11111111b - Reserved

16.3.4 SIUL2 DMA/Interrupt Status Flag Register0 (DISR0)

Offset

Register	Offset
DISR0	10h

Function

DISR0 contains flag bits that record an event on the external IRQ pins. When an event as defined in IREER0 and IFEER0 occurs, the corresponding flag bit is set. The IRQ flag bit is set regardless of the state of the corresponding DIRER0[EIRE*n*] bit. The IRQ flag bit remains set until you clear it or is cleared by servicing of a DMA request. The IRQ flag bits are cleared by writing a 1 to the bits. A write of 0 has no effect.

This register supports 8-, 16-, and 32-bit accesses.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	EIF11	EIF10	EIF9	EIF8	EIF7	EIF6	EIF5	EIF4	EIF3	EIF2	EIF1	EIF0
W					W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab62ec1

Table continued from the previous page...

Field	Function
—	
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 EIF11	External Interrupt Status Flag 11 If enabled (DIRERR11 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER11 and IFEER11 has occurred.
10 EIF10	External Interrupt Status Flag 10 If enabled (DIRERR10 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER10 and IFEER10 has occurred.
9 EIF9	External Interrupt Status Flag 9 If enabled (DIRERR9 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER9 and IFEER9 has occurred.
8 EIF8	External Interrupt Status Flag 8 If enabled (DIRERR8 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER8 and IFEER8 has occurred.
7 EIF7	External Interrupt Status Flag 7 If enabled (DIRERR7 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER7 and IFEER7 has occurred.
6 EIF6	External Interrupt Status Flag 6 If enabled (DIRERR6 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER6 and IFEER6 has occurred.
5 EIF5	External Interrupt Status Flag 5 If enabled (DIRERR5 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER5 and IFEER5 has occurred.
4 EIF4	External Interrupt Status Flag 4 If enabled (DIRERR4 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - An interrupt event as defined by IREER4 and IFEER4 has occurred.
3 EIF3	External Interrupt Status Flag 3 If enabled (DIRERR3 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER3 and IFEER3 has occurred.
2 EIF2	External Interrupt Status Flag 2 If enabled (DIRERR2 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER2 and IFEER2 has occurred.
1 EIF1	External Interrupt Status Flag 1 If enabled (DIRERR1 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER1 and IFEER1 has occurred.
0 EIF0	External Interrupt Status Flag 0 If enabled (DIRERR0 = 1) causes an interrupt. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER0 and IFEER0 has occurred.

16.3.5 SIUL2 DMA/Interrupt Request Enable Register0 (DIRER0)

Offset

Register	Offset
DIRER0	18h

Function

DIRER0 enables the assertion of DMA or interrupt request to the interrupt controller if the corresponding DISR0[EIF*n*] bit is set. The type of request is determined by the corresponding DIRSR0[DIRSR*n*] bit.

This register supports 8-, 16-, and 32-bit accesses.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca462e5-4821-42a8-b251-3901ab624c1

NOTE

DMA requests cannot be enabled or disabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	EIRE11	EIRE10	EIRE9	EIRE8	EIRE7	EIRE6	EIRE5	EIRE4	EIRE3	EIRE2	EIRE1	EIRE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab04fc1

Table continued from the previous page...

Field	Function
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 EIRE11	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
10 EIRE10	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
9 EIRE9	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
8 EIRE8	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
7 EIRE7	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
6 EIRE6	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
5 EIRE5	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
4 EIRE4	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
3 EIRE3	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
2 EIRE2	External Interrupt Request Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
1 EIRE1	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
0 EIRE0	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled

16.3.6 SIUL2 DMA/Interrupt Request Select Register0 (DIRSR0)

Offset

Register	Offset
DIRSR0	20h

Function

DIRSR0 selects between the DMA or interrupt request. The DIRSR0[DIRSR*n*] bit determines whether DMA or an interrupt request asserts the corresponding DISR0[EIF*n*] bit.

This register supports 8-, 16-, and 32-bit accesses.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	DIRSR 11	DIRSR 10	DIRSR 9	DIRSR 8	DIRSR 7	DIRSR 6	DIRSR 5	DIRSR 4	DIRSR 3	DIRSR 2	DIRSR 1	DIRSR 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEC Automotive Technology Co., Ltd.
c8c20e5-8321-488-b251-390d3ab624c1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 DIRSR11	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
10 DIRSR10	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
9 DIRSR9	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
8 DIRSR8	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 DIRSR7	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
6 DIRSR6	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
5 DIRSR5	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
4 DIRSR4	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
3 DIRSR3	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
2 DIRSR2	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved
1 DIRSR1	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt request 1b - Reserved
0 DIRSR0	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - Reserved

16.3.7 SIUL2 Interrupt Rising-Edge Event Enable Register 0 (IREER0)

Offset

Register	Offset
IREER0	28h

Function

IREER0 enables the rising-edge triggered events on the corresponding interrupt pads.

This register supports 8-, 16-, and 32-bit accesses.

NOTE

If both the IREE and IFEE bits are cleared for the same interrupt source, the interrupt status flag for the corresponding external interrupt will never be set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	IREE11	IREE10	IREE9	IREE8	IREE7	IREE6	IREE5	IREE4	IREE3	IREE2	IREE1	IREE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHIPSTAR Automotive Technology Co., Ltd.
c8c46765-4521-48a8-b251-390d3ab624c1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 IREE11	Enables rising-edge events to set DISR0[EIF11]. 0b - Disabled 1b - Enabled
10 IREE10	Enables rising-edge events to set DISR0[EIF10]. 0b - Disabled 1b - Enabled
9 IREE9	Enables rising-edge events to set DISR0[EIF9]. 0b - Disabled 1b - Enabled
8 IREE8	Enables rising-edge events to set DISR0[EIF8]. 0b - Disabled 1b - Enabled
7 IREE7	Enables rising-edge events to set DISR0[EIF7]. 0b - Disabled 1b - Enabled
6 IREE6	Enables rising-edge events to set DISR0[EIF6]. 0b - Disabled 1b - Enabled
5	Enables rising-edge events to set DISR0[EIF5].

Table continues on the next page...

Table continued from the previous page...

Field	Function
IREE5	0b - Disabled 1b - Enabled
4 IREE4	Enables rising-edge events to set DISR0[EIF4]. 0b - Disabled 1b - Enabled
3 IREE3	Enables rising-edge events to set DISR0[EIF3]. 0b - Disabled 1b - Enabled
2 IREE2	Enables rising-edge events to set DISR0[EIF2]. 0b - Disabled 1b - Enabled
1 IREE1	Enables rising-edge events to set DISR0[EIF1]. 0b - Disabled 1b - Enabled
0 IREE0	Enables rising-edge events to set DISR0[EIF0]. 0b - Disabled 1b - Enabled

16.3.8 SIUL2 Interrupt Falling-Edge Event Enable Register 0 (IFEER0)

Offset

Register	Offset
IFEER0	30h

Function

IFEER0 enables falling-edge triggered events on the corresponding interrupt pads.

This register supports 8-, 16-, and 32-bit accesses.

NOTE

If both the IREE and IFEE bits are cleared for the same interrupt source, the interrupt status flag for the corresponding external interrupt will never be set.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c40e5-4821-48a2-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	IFEE11	IFEE10	IFEE9	IFEE8	IFEE7	IFEE6	IFEE5	IFEE4	IFEE3	IFEE2	IFEE1	IFEE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 IFEE11	Enables falling-edge events to set DISR0[EIF11]. 0b - Disabled 1b - Enabled
10 IFEE10	Enables falling-edge events to set DISR0[EIF10]. 0b - Disabled 1b - Enabled
9 IFEE9	Enables falling-edge events to set DISR0[EIF9]. 0b - Disabled 1b - Enabled
8	Enables falling-edge events to set DISR0[EIF8].

Table continues on the next page...

Table continued from the previous page...

Field	Function
IFEE8	0b - Disabled 1b - Enabled
7 IFEE7	Enables falling-edge events to set DISR0[EIF7]. 0b - Disabled 1b - Enabled
6 IFEE6	Enables falling-edge events to set DISR0[EIF6]. 0b - Disabled 1b - Enabled
5 IFEE5	Enables falling-edge events to set DISR0[EIF5]. 0b - Disabled 1b - Enabled
4 IFEE4	Enables falling-edge events to set DISR0[EIF4]. 0b - Disabled 1b - Enabled
3 IFEE3	Enables falling-edge events to set DISR0[EIF3]. 0b - Disabled 1b - Enabled
2 IFEE2	Enables falling-edge events to set DISR0[EIF2]. 0b - Disabled 1b - Enabled
1 IFEE1	Enables falling-edge events to set DISR0[EIF1]. 0b - Disabled 1b - Enabled
0 IFEE0	Enables falling-edge events to set DISR0[EIF0]. 0b - Disabled 1b - Enabled

16.3.9 SIUL2 Interrupt Filter Enable Register 0 (IFER0)

Offset

Register	Offset
IFER0	38h

Provided by IHS Markit
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4d21-48a8-b251-390d3ab624c1

Function

IFER0 enables a digital filter counter on the corresponding interrupt pads to filter out glitches on the inputs.

This register supports 8-, 16-, and 32-bit accesses.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	IFE11	IFE10	IFE9	IFE8	IFE7	IFE6	IFE5	IFE4	IFE3	IFE2	IFE1	IFE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3db024c1

Table continued from the previous page...

Field	Function
—	
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 IFE11	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
10 IFE10	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 IFE9	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
8 IFE8	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
7 IFE7	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
6 IFE6	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
5 IFE5	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
4 IFE4	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
3 IFE3	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
2 IFE2	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
1 IFE1	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
0 IFE0	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-1821-48a8-b251-390d3ab624c1

16.3.10 SIUL2 Interrupt Filter Maximum Counter Register (IFMCR0 - IFMCR11)

Offset

For a = 0 to 11:

Register	Offset
IFMCRa	40h + (a × 4h)

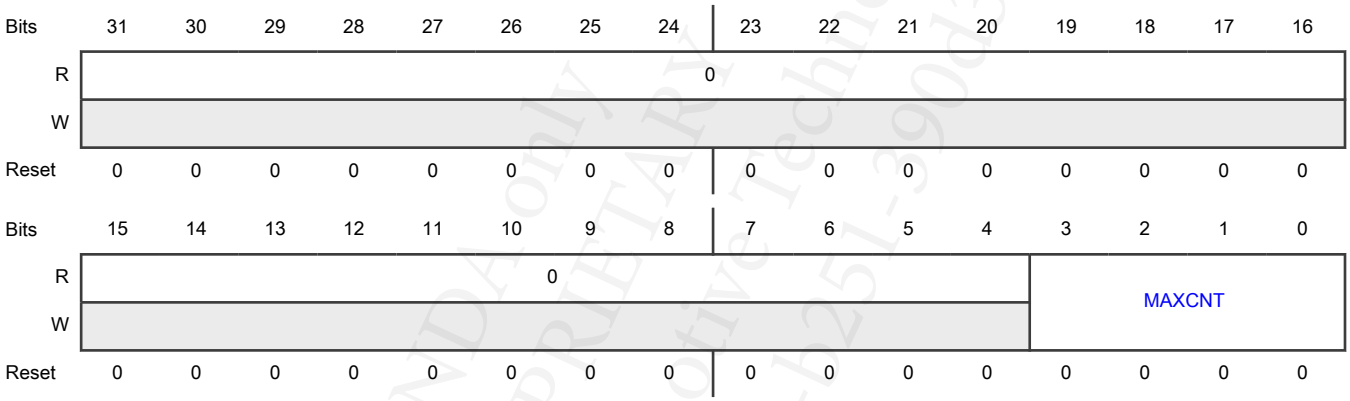
Function

IFMCR*n* registers configure the filter counter associated with each digital glitch filter.

NOTE

These registers support only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 MAXCNT	Maximum Interrupt Filter Counter setting MAXCNT can be 0d to 15d. • A value of 0d, 1d, or 2d sets the filter as an all pass filter. • A value of 3d to 15d sets the filter period to TCK × MAXCNT + *n* × TCK, where: — *n* is 0, 1, 2, 3, or 4. — TCK is the prescaled filter clock period, which is the IRC clock prescaled to the IFCP value specified in IFCPR.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e2e5-4821-48a8-b251-290d3ab624c1

16.3.11 SIUL2 Interrupt Filter Clock Prescaler Register (IFCPR)

Offset

Register	Offset
IFCPR	C0h

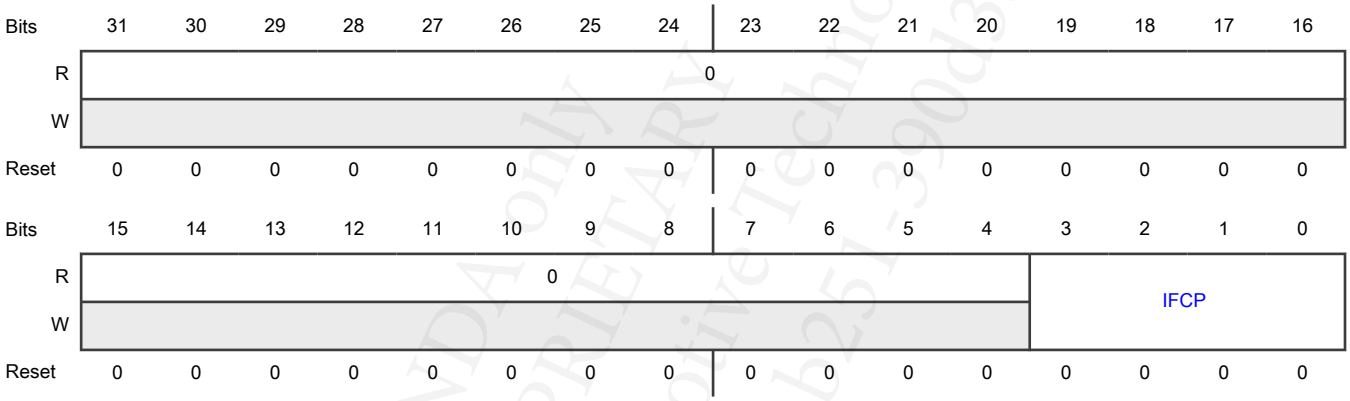
Function

IFCPR configures the clock prescaler which selects the clock for all digital filters. A prescaler is applied to the input clock to SIUL2, which is the peripheral clock counter in the SIUL2.

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 IFCP	Interrupt Filter Clock Prescaler setting Prescaled Filter Clock period is T_{IRC} × (IFCP + 1), where: • T_{IRC} is the internal oscillator period. • IFCP is 0 to 15.

16.3.12 SIUL2 Multiplexed Signal Configuration Register (MSCR0 - MSCR101)

Offset

For a = 0 to 101:

Register	Offset
MSCRa	240h + (a × 4h)

Function

MSCR*n* registers select the source signal connected to the register's associated destination, which is a chip output pin or a chip pin that can be configured as an output.

MSCR*n* also specifies the electrical properties of the associated pin.

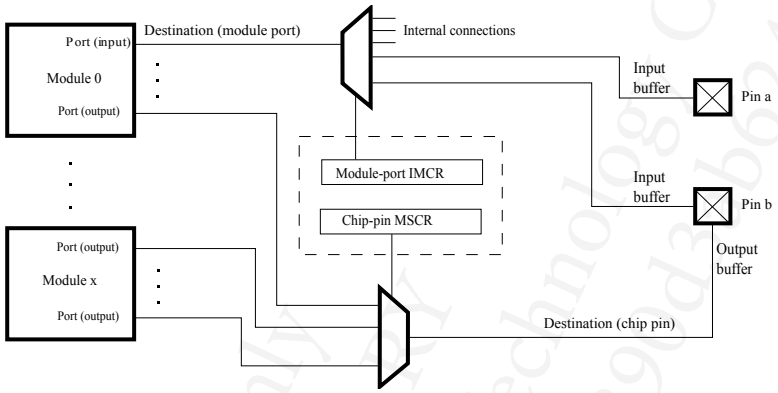


Figure 38. MSCR and IMCR port and pin connection

For chip-pin MSCR assignments and pin types, see the IOMUX file attached to this document.

NOTE

- MSCR*n* registers support only 32-bit accesses. Byte and half-word write accesses are not supported.
- MSCR*n* registers must be configured only during application initialization and must not be modified during application runtime.
- Accessing a reserved MSCR*n* register generates a transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0										OBE	ODE	IBE	Reserved		SRE
W																
Reset	See Register reset values.															

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SRE		PUE	PUS	Reserved	RCVR	0				SMC	0		SSS		
W																
Reset	See Register reset values.															

Register reset values

Register	Reset value
MSCR0	0009_3000h
MSCR1	0001_0021h
MSCR2–MSCR4	0009_2000h
MSCR5	0009_3021h
MSCR6–MSCR101	0001_0000h

Fields

Field	Function
31-22 —	Reserved
21 OBE	GPIO Output Buffer Enable Applies only to digital pins. Otherwise this bit is reserved. 0b - Output driver disabled 1b - Output driver enabled
20 ODE	Open Drain Enable **NOTE** To enable open drain both OBE and ODE bits need to be set. 0b - Open drain function disabled 1b - Open drain function enabled when OBE is also 1
19 IBE	Input Buffer Enable Used only when the associated destination is a chip pin. Enables the associated pin's input buffer. 0b - Disabled 1b - Enabled
18-17 —	Reserved Has no function.
16-14 SRE	Slew Rate Control • For "3.3 V/1.8 V" FAST pads: — 000b: Fmax= 208 MHz (at 1.8V), 166 MHz (at 3.3V) — 001b: Reserved — 010b: Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	— 011b: Reserved — 100b: Fmax=166 MHz (at 1.8V), 150 MHz (at 3.3V) — 101b: Fmax=150 MHz (at 1.8V), 133 MHz (at 3.3V) — 110b: Fmax=133 MHz(at 1.8V), 100 MHz (at 3.3V) — 111b, Fmax=83 MHz (at 1.8V), 63 MHz (at 3.3V) **NOTE** Rout increases with each SRE setting. • For "1.8 V" GPIO pads: — 000b: Fmax=208 MHz — 001b: Reserved — 010b: Reserved — 011b: Reserved — 100b: Fmax=166 MHz — 101b: Fmax=150 MHz — 110b: Fmax=133 MHz — 111b: Fmax=83 MHz **NOTE** Rout is fixed • For "3.3 V" GPIO pads: — 000b: Reserved — 001b: Reserved — 010b: Reserved — 011b: Reserved — 100b: Fmax=50 MHz — 101b: Fmax=50 MHz — 110b: Fmax=10 MHz — 111b: Fmax=1 MHz **NOTE** Rout increases with each SRE setting.
13 PUE	Pull Enable Enables the pull function. Used only when the associated destination is a chip pin. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 PUS	Pull Select Determines whether the pull function is a pullup or pulldown when the pull function is enabled by the PUE field. Used only when the associated destination is a chip pin. 0b - Pulldown 1b - Pullup
11 —	Reserved Has no function.
10 RCVR	Receiver Select **NOTE** 1833 and GPIO33 pad types do not support this feature. RCVR field should be set before configuring IBE field to avoid glitching of the receiver output in the design. 0b - Enables the differential vref based receiver. 1b - Enables the single ended receiver.
9-6 —	Reserved
5 SMC	Safe Mode Control Used only when the associated destination is a chip pin. Specifies whether the chip disables the pin's output buffer when FCCU enters in Fault state. 0b - Disable (The output buffer returns to its previous state when FCCU leaves Fault state.) 1b - Don't disable
4-3 —	Reserved
2-0 SSS	Source Signal Select Selects a function for the pad. Refer to "SSS" column of the 'IO Signal Table' tab of the IOMUX spreadsheet attachment.

16.3.13 SIUL2 Input Multiplexed Signal Configuration Register (IMCR0 - IMCR83)

Offset

Register	Offset
IMCR0	A40h
IMCR1	A44h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR3	A4Ch
IMCR4	A50h
IMCR5	A54h
IMCR6	A58h
IMCR7	A5Ch
IMCR8	A60h
IMCR9	A64h
IMCR10	A68h
IMCR11	A6Ch
IMCR12	A70h
IMCR13	A74h
IMCR14	A78h
IMCR15	A7Ch
IMCR16	A80h
IMCR17	A84h
IMCR18	A88h
IMCR19	A8Ch
IMCR20	A90h
IMCR21	A94h
IMCR22	A98h
IMCR23	A9Ch
IMCR24	AA0h
IMCR25	AA4h
IMCR26	AA8h
IMCR27	AACh
IMCR28	AB0h
IMCR29	AB4h
IMCR30	AB8h
IMCR31	ABCh
IMCR32	AC0h
IMCR33	AC4h
IMCR34	AC8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR35	ACCh
IMCR36	AD0h
IMCR37	AD4h
IMCR38	AD8h
IMCR39	ADCh
IMCR40	AE0h
IMCR41	AE4h
IMCR42	AE8h
IMCR43	AECh
IMCR44	AF0h
IMCR45	AF4h
IMCR46	AF8h
IMCR47	AFCh
IMCR48	B00h
IMCR49	B04h
IMCR50	B08h
IMCR51	B0Ch
IMCR52	B10h
IMCR53	B14h
IMCR54	B18h
IMCR55	B1Ch
IMCR56	B20h
IMCR57	B24h
IMCR58	B28h
IMCR59	B2Ch
IMCR60	B30h
IMCR61	B34h
IMCR68	B50h
IMCR69	B54h
IMCR70	B58h
IMCR71	B5Ch
IMCR72	B60h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR73	B64h
IMCR74	B68h
IMCR75	B6Ch
IMCR76	B70h
IMCR77	B74h
IMCR78	B78h
IMCR79	B7Ch
IMCR80	B80h
IMCR81	B84h
IMCR82	B88h
IMCR83	B8Ch

Function

IMCR*n* registers select the source signal connected to the register's associated destination, which is an internal module port that is an input port or can be configured as an input.

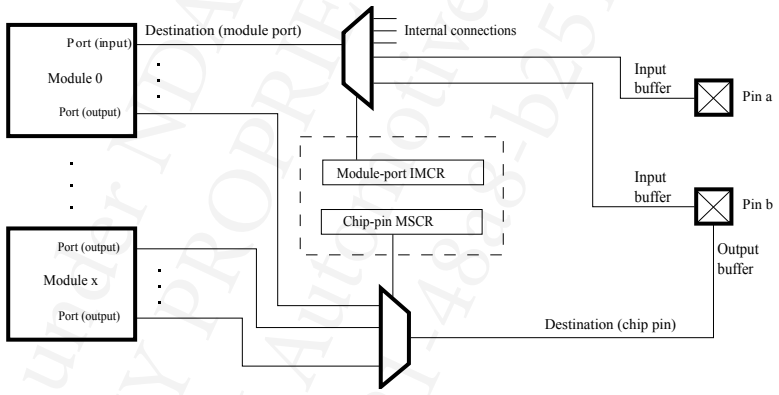


Figure 39. MSCR and IMCR port and pin connection

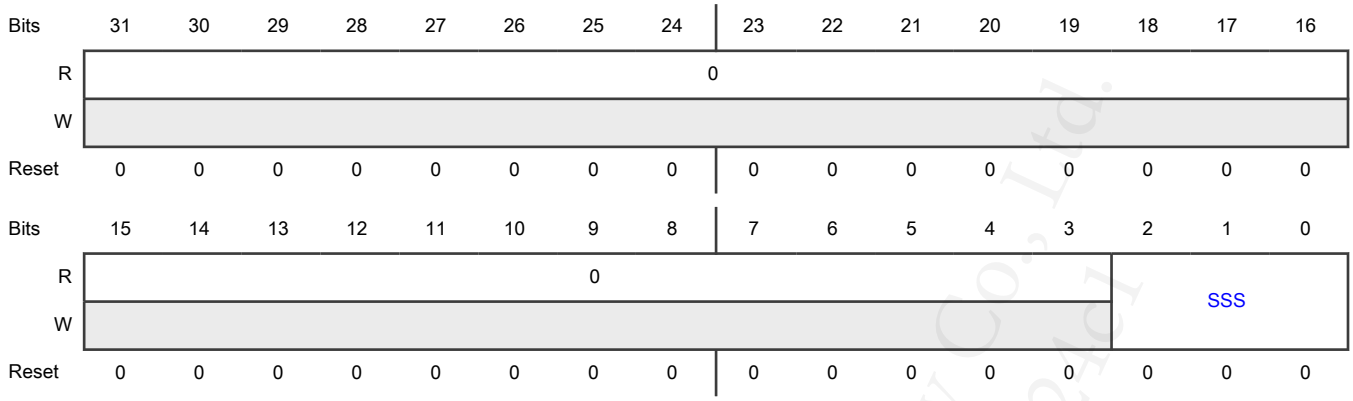
For IMCR assignments and field values, see the IOMUX file attached to this document.

NOTE

- IMCR*n* registers support only 32-bit accesses. Byte and half-word write accesses are not supported.
- IMCR*n* registers must be configured only during application initialization and must not be modified during application runtime.
- Accessing a reserved IMCR*n* register generates a transfer error.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c4625e-8214-4823-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 SSS	Source Signal Select Selects which source signal is connected to the associated destination (chip pin).

16.3.14 SIUL2 GPIO Pad Data Output Register (GPDO0 - GPDO101)

Offset

Register	Offset
GPDO3	1300h
GPDO2	1301h
GPDO1	1302h
GPDO0	1303h
GPDO7	1304h
GPDO6	1305h
GPDO5	1306h
GPDO4	1307h
GPDO11	1308h
GPDO10	1309h
GPDO9	130Ah
GPDO8	130Bh
GPDO15	130Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
GPDO14	130Dh
GPDO13	130Eh
GPDO12	130Fh
GPDO19	1310h
GPDO18	1311h
GPDO17	1312h
GPDO16	1313h
GPDO23	1314h
GPDO22	1315h
GPDO21	1316h
GPDO20	1317h
GPDO27	1318h
GPDO26	1319h
GPDO25	131Ah
GPDO24	131Bh
GPDO31	131Ch
GPDO30	131Dh
GPDO29	131Eh
GPDO28	131Fh
GPDO35	1320h
GPDO34	1321h
GPDO33	1322h
GPDO32	1323h
GPDO39	1324h
GPDO38	1325h
GPDO37	1326h
GPDO36	1327h
GPDO43	1328h
GPDO42	1329h
GPDO41	132Ah
GPDO40	132Bh
GPDO47	132Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
GPDO46	132Dh
GPDO45	132Eh
GPDO44	132Fh
GPDO51	1330h
GPDO50	1331h
GPDO49	1332h
GPDO48	1333h
GPDO55	1334h
GPDO54	1335h
GPDO53	1336h
GPDO52	1337h
GPDO59	1338h
GPDO58	1339h
GPDO57	133Ah
GPDO56	133Bh
GPDO63	133Ch
GPDO62	133Dh
GPDO61	133Eh
GPDO60	133Fh
GPDO67	1340h
GPDO66	1341h
GPDO65	1342h
GPDO64	1343h
GPDO71	1344h
GPDO70	1345h
GPDO69	1346h
GPDO68	1347h
GPDO75	1348h
GPDO74	1349h
GPDO73	134Ah
GPDO72	134Bh
GPDO79	134Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
GPDO78	134Dh
GPDO77	134Eh
GPDO76	134Fh
GPDO83	1350h
GPDO82	1351h
GPDO81	1352h
GPDO80	1353h
GPDO87	1354h
GPDO86	1355h
GPDO85	1356h
GPDO84	1357h
GPDO91	1358h
GPDO90	1359h
GPDO89	135Ah
GPDO88	135Bh
GPDO95	135Ch
GPDO94	135Dh
GPDO93	135Eh
GPDO92	135Fh
GPDO99	1360h
GPDO98	1361h
GPDO97	1362h
GPDO96	1363h
GPDO101	1366h
GPDO100	1367h

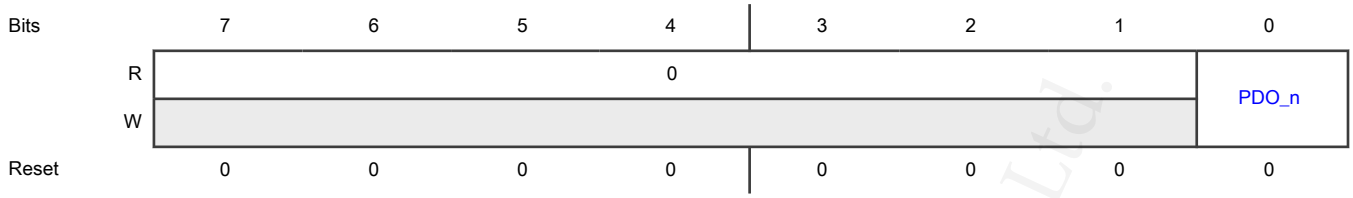
Function

Each GPDO*n* register sets or clears a single GPIO pad with a byte access.

These registers support 8-, 16-, and 32-bit accesses.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
7-1 —	Reserved
0 PDO_n	Pad Data Out Stores the data to be driven out on the external GPIO pad controlled by this register when the pad is configured as an output. PDO_*n* represents PDO[*n*], where *n* is the instance of the register. 0b - Logic low value 1b - Logic high value

16.3.15 SIUL2 GPIO Pad Data Input Register (GPDI0 - GPDI101)

Offset

Register	Offset
GPDI3	1500h
GPDI2	1501h
GPDI1	1502h
GPDI0	1503h
GPDI7	1504h
GPDI6	1505h
GPDI5	1506h
GPDI4	1507h
GPDI11	1508h
GPDI10	1509h
GPDI9	150Ah
GPDI8	150Bh
GPDI15	150Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
GPDI14	150Dh
GPDI13	150Eh
GPDI12	150Fh
GPDI19	1510h
GPDI18	1511h
GPDI17	1512h
GPDI16	1513h
GPDI23	1514h
GPDI22	1515h
GPDI21	1516h
GPDI20	1517h
GPDI27	1518h
GPDI26	1519h
GPDI25	151Ah
GPDI24	151Bh
GPDI31	151Ch
GPDI30	151Dh
GPDI29	151Eh
GPDI28	151Fh
GPDI35	1520h
GPDI34	1521h
GPDI33	1522h
GPDI32	1523h
GPDI39	1524h
GPDI38	1525h
GPDI37	1526h
GPDI36	1527h
GPDI43	1528h
GPDI42	1529h
GPDI41	152Ah
GPDI40	152Bh
GPDI47	152Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
GPDI46	152Dh
GPDI45	152Eh
GPDI44	152Fh
GPDI51	1530h
GPDI50	1531h
GPDI49	1532h
GPDI48	1533h
GPDI55	1534h
GPDI54	1535h
GPDI53	1536h
GPDI52	1537h
GPDI59	1538h
GPDI58	1539h
GPDI57	153Ah
GPDI56	153Bh
GPDI63	153Ch
GPDI62	153Dh
GPDI61	153Eh
GPDI60	153Fh
GPDI67	1540h
GPDI66	1541h
GPDI65	1542h
GPDI64	1543h
GPDI71	1544h
GPDI70	1545h
GPDI69	1546h
GPDI68	1547h
GPDI75	1548h
GPDI74	1549h
GPDI73	154Ah
GPDI72	154Bh
GPDI79	154Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
GPDI78	154Dh
GPDI77	154Eh
GPDI76	154Fh
GPDI83	1550h
GPDI82	1551h
GPDI81	1552h
GPDI80	1553h
GPDI87	1554h
GPDI86	1555h
GPDI85	1556h
GPDI84	1557h
GPDI91	1558h
GPDI90	1559h
GPDI89	155Ah
GPDI88	155Bh
GPDI95	155Ch
GPDI94	155Dh
GPDI93	155Eh
GPDI92	155Fh
GPDI99	1560h
GPDI98	1561h
GPDI97	1562h
GPDI96	1563h
GPDI101	1566h
GPDI100	1567h

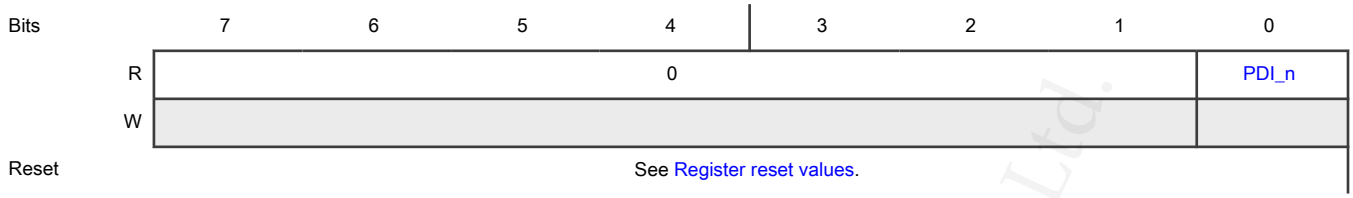
Function

Each GPDI*n* register reads the GPIO pad data with a byte access.

These registers support 8-, 16-, and 32-bit accesses.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Diagram



Register reset values

Register	Reset value
GPDI0	01h
GPDI1–GPDI4	00h
GPDI5	01h
GPDI6–GPDI101	00h
GPDI102–GPDI103	Register not supported

Fields

Field	Function
7-1 —	Reserved
0 PDI_n	Pad Data In Stores the value of the external GPIO pad associated with this register. PDI_*n* represents PDI[*n*], where *n* is the instance of the register. 0b - Logic low 1b - Logic high

16.3.16 SIUL2 Parallel GPIO Pad Data Out Register (PGPDO0 - PGPDO5)

Offset

Register	Offset
PGPDO1	1700h
PGPDO0	1702h
PGPDO3	1704h
PGPDO2	1706h
PGPDO5	1708h
PGPDO4	170Ah

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Function

Each PGPDO*n* register sets or clears the respective pads of the chip.

PGPDO*n* registers can set the values of all output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* registers set the value on a specific pin with byte writes.

Each bit writes or reads the data register that stores the value to be driven on the pad in output mode. Access to this register location is coherent with access to the bit-wise GPDO*n*.

For a given PGPDO*x*[PPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

PGPDO*n* registers access the same physical resource as the PDO and MPGPDO address locations.

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO
W	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15
14 PPDO14	Parallel Pad Data Out 14
13 PPDO13	Parallel Pad Data Out 13
12 PPDO12	Parallel Pad Data Out 12
11 PPDO11	Parallel Pad Data Out 11
10	Parallel Pad Data Out 10

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDO10	
9 PPDO9	Parallel Pad Data Out 9
8 PPDO8	Parallel Pad Data Out 8
7 PPDO7	Parallel Pad Data Out 7
6 PPDO6	Parallel Pad Data Out 6
5 PPDO5	Parallel Pad Data Out 5
4 PPDO4	Parallel Pad Data Out 4
3 PPDO3	Parallel Pad Data Out 3
2 PPDO2	Parallel Pad Data Out 2
1 PPDO1	Parallel Pad Data Out 1
0 PPDO0	Parallel Pad Data Out 0

16.3.17 SIUL2 Parallel GPIO Pad Data Out Register (PGPDO6)

Offset

Register	Offset
PGPDO6	170Eh

Function

Each PGPDO*n* register sets or clears the respective pads of the chip.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co., Ltd.
8c462e5-4821-48a8-b251-390d3ab624c1

PGPDO*n* registers can set the values of all output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* registers set the value on a specific pin with byte writes.

Each bit writes or reads the data register that stores the value to be driven on the pad in output mode. Access to this register location is coherent with access to the bit-wise GPDO*n*.

For a given PGPDO*x*[PPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

PGPDO*n* registers access the same physical resource as the PDO and MPGPDO address locations.

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDO 15	PPDO 14	PPDO 13	PPDO 12	PPDO 11	PPDO 10	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15
14 PPDO14	Parallel Pad Data Out 14
13 PPDO13	Parallel Pad Data Out 13
12 PPDO12	Parallel Pad Data Out 12
11 PPDO11	Parallel Pad Data Out 11
10 PPDO10	Parallel Pad Data Out 10
9	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Always write zero to this field.
8 —	Reserved Always write zero to this field.
7 —	Reserved Always write zero to this field.
6 —	Reserved Always write zero to this field.
5 —	Reserved Always write zero to this field.
4 —	Reserved Always write zero to this field.
3 —	Reserved Always write zero to this field.
2 —	Reserved Always write zero to this field.
1 —	Reserved Always write zero to this field.
0 —	Reserved Always write zero to this field.

16.3.18 SIUL2 Parallel GPIO Pad Data In Register (PGPDI0 - PGPDI5)

Offset

Register	Offset
PGPDI1	1740h
PGPDI0	1742h
PGPDI3	1744h
PGPDI2	1746h
PGPDI5	1748h
PGPDI4	174Ah

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462c5-4827-48a8-b251-390d3ab624c1

Function

PGPDI*n* registers hold the synchronized input value from the pads.

PGPDI*n* registers can read the values of all input pins assigned to a chip port with a single 16-bit register read, while the GPDI*n* registers read the value on a specific pin with a byte read.

Each bit reads the current pad value. Access to this register location is coherent with access to the bit-wise GPDI*n*.

For a given PGPDI*x*[PPDI*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] bit:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	PPDI9	PPDI8	PPDI7	PPDI6	PPDI5	PPDI4	PPDI3	PPDI2	PPDI1	PPDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data In 15
14 PPDI14	Parallel Pad Data In 14
13 PPDI13	Parallel Pad Data In 13
12 PPDI12	Parallel Pad Data In 12
11 PPDI11	Parallel Pad Data In 11
10 PPDI10	Parallel Pad Data In 10

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 PPDI9	Parallel Pad Data In 9
8 PPDI8	Parallel Pad Data In 8
7 PPDI7	Parallel Pad Data In 7
6 PPDI6	Parallel Pad Data In 6
5 PPDI5	Parallel Pad Data In 5
4 PPDI4	Parallel Pad Data In 4
3 PPDI3	Parallel Pad Data In 3
2 PPDI2	Parallel Pad Data In 2
1 PPDI1	Parallel Pad Data In 1
0 PPDI0	Parallel Pad Data In 0

16.3.19 SIUL2 Parallel GPIO Pad Data In Register (PGPDI6)

Offset

Register	Offset
PGPDI6	174Eh

Function

PGPDI*n* registers hold the synchronized input value from the pads.

PGPDI*n* registers can read the values of all input pins assigned to a chip port with a single 16-bit register read, while the GPDI*n* registers read the value on a specific pin with a byte read.

Each bit reads the current pad value. Access to this register location is coherent with access to the bit-wise GPDI*n*.

For a given PGPDI*x*[PPDI*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] bit:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data In 15
14 PPDI14	Parallel Pad Data In 14
13 PPDI13	Parallel Pad Data In 13
12 PPDI12	Parallel Pad Data In 12
11 PPDI11	Parallel Pad Data In 11
10 PPDI10	Parallel Pad Data In 10
9 —	Reserved
8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

16.3.20 SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO0 - MPGPDO5)

Offset

Register	Offset
MPGPDO0	1780h
MPGPDO1	1784h
MPGPDO2	1788h
MPGPDO3	178Ch
MPGPDO4	1790h
MPGPDO5	1794h

Function

Each MPGPDO*n* register selectively modifies the pad values associated with PGPDO*n*.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

NOTE

MPGPDO*n* registers must only be accessed with 32-bit writes. 8-bit or 16-bit writes will not modify any bits in the register and cause a transfer error. Read access will return 0.

NOTE

MPGPDO*n* registers support only 32-bit accesses. Byte and half-word accesses are not supported.

Accesses to each MPGPDO*n* register location is coherent with access to the bit-wise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]
- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	MASK 9	MASK 8	MASK 7	MASK 6	MASK 5	MASK 4	MASK 3	MASK 2	MASK 1	MASK 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	MPPDO9	MPPDO8	MPPDO7	MPPDO6	MPPDO5	MPPDO4	MPPDO3	MPPDO2	MPPDO1	MPPDO0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MASK15	Mask Field 15 Masks MPPDO15 in the same MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the same MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29	Mask Field 13

Table continues on the next page...

Table continued from the previous page...

Field	Function
MASK13	Masks MPPDO13 in the same MPGPDO*n* register instance. 0b - MPPDO13 is ignored 1b - MPPDO13 is written
28 MASK12	Mask Field 12 Masks MPPDO12 in the same MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written
27 MASK11	Mask Field 11 Masks MPPDO11 in the same MPGPDO*n* register instance. 0b - MPPDO11 is ignored 1b - MPPDO11 is written
26 MASK10	Mask Field 10 Masks MPPDO10 in the same MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 MASK9	Mask Field 9 Masks MPPDO9 in the same MPGPDO*n* register instance. 0b - MPPDO9 is ignored 1b - MPPDO9 is written
24 MASK8	Mask Field 8 Masks MPPDO8 in the same MPGPDO*n* register instance. 0b - MPPDO8 is ignored 1b - MPPDO8 is written
23 MASK7	Mask Field 7 Masks MPPDO7 in the same MPGPDO*n* register instance. 0b - MPPDO7 is ignored 1b - MPPDO7 is written
22 MASK6	Mask Field 6 Masks MPPDO6 in the same MPGPDO*n* register instance. 0b - MPPDO6 is ignored 1b - MPPDO6 is written

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 MASK5	Mask Field 5 Masks MPPDO5 in the same MPGPDO*n* register instance. 0b - MPPDO5 is ignored 1b - MPPDO5 is written
20 MASK4	Mask Field 4 Masks MPPDO4 in the same MPGPDO*n* register instance. 0b - MPPDO4 is ignored 1b - MPPDO4 is written
19 MASK3	Mask Field 3 Masks MPPDO3 in the same MPGPDO*n* register instance. 0b - MPPDO3 is ignored 1b - MPPDO3 is written
18 MASK2	Mask Field 2 Masks MPPDO2 in the same MPGPDO*n* register instance. 0b - MPPDO2 is ignored 1b - MPPDO2 is written
17 MASK1	Mask Field 1 Masks MPPDO1 in the same MPGPDO*n* register instance. 0b - MPPDO1 is ignored 1b - MPPDO1 is written
16 MASK0	Mask Field 0 Masks MPPDO0 in the same MPGPDO*n* register instance. 0b - MPPDO0 is ignored 1b - MPPDO0 is written
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes the data register that stores the value to be driven on the pad in output mode.
14 MPPDO14	Masked Parallel Pad Data Out 14 Writes the data register that stores the value to be driven on the pad in output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes the data register that stores the value to be driven on the pad in output mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 MPPDO12	Masked Parallel Pad Data Out 12 Writes the data register that stores the value to be driven on the pad in output mode.
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes the data register that stores the value to be driven on the pad in output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes the data register that stores the value to be driven on the pad in output mode.
9 MPPDO9	Masked Parallel Pad Data Out 9 Writes the data register that stores the value to be driven on the pad in output mode.
8 MPPDO8	Masked Parallel Pad Data Out 8 Writes the data register that stores the value to be driven on the pad in output mode.
7 MPPDO7	Masked Parallel Pad Data Out 7 Writes the data register that stores the value to be driven on the pad in output mode.
6 MPPDO6	Masked Parallel Pad Data Out 6 Writes the data register that stores the value to be driven on the pad in output mode.
5 MPPDO5	Masked Parallel Pad Data Out 5 Writes the data register that stores the value to be driven on the pad in output mode.
4 MPPDO4	Masked Parallel Pad Data Out 4 Writes the data register that stores the value to be driven on the pad in output mode.
3 MPPDO3	Masked Parallel Pad Data Out 3 Writes the data register that stores the value to be driven on the pad in output mode.
2 MPPDO2	Masked Parallel Pad Data Out 2 Writes the data register that stores the value to be driven on the pad in output mode.
1 MPPDO1	Masked Parallel Pad Data Out 1 Writes the data register that stores the value to be driven on the pad in output mode.
0 MPPDO0	Masked Parallel Pad Data Out 0 Writes the data register that stores the value to be driven on the pad in output mode.

Preliminary NDA Only
COMPANY PROPRIETARY
Beijing CEC Automotive Technology Co., Ltd.
c8c46e3-4821-48-8-251-5903a624c1

16.3.21 SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO6)

Offset

Register	Offset
MPGPDO6	1798h

Function

Each MPGPDO*n* register selectively modifies the pad values associated with PGPDO*n*.

NOTE

MPGPDO*n* registers must only be accessed with 32-bit writes. 8-bit or 16-bit writes will not modify any bits in the register and cause a transfer error. Read access will return 0.

NOTE

MPGPDO*n* registers support only 32-bit accesses. Byte and half-word accesses are not supported.

Accesses to each MPGPDO*n* register location is coherent with access to the bit-wise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]
- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Mask Field 15

Table continues on the next page...

Table continued from the previous page...

Field	Function
MASK15	Masks MPPDO15 in the same MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the same MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29 MASK13	Mask Field 13 Masks MPPDO13 in the same MPGPDO*n* register instance. 0b - MPPDO13 is ignored 1b - MPPDO13 is written
28 MASK12	Mask Field 12 Masks MPPDO12 in the same MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written
27 MASK11	Mask Field 11 Masks MPPDO11 in the same MPGPDO*n* register instance. 0b - MPPDO11 is ignored 1b - MPPDO11 is written
26 MASK10	Mask Field 10 Masks MPPDO10 in the same MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 —	Reserved Always write zero to this field.
24 —	Reserved Always write zero to this field.
23 —	Reserved Always write zero to this field.
22	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Always write zero to this field.
21 —	Reserved Always write zero to this field.
20 —	Reserved Always write zero to this field.
19 —	Reserved Always write zero to this field.
18 —	Reserved Always write zero to this field.
17 —	Reserved Always write zero to this field.
16 —	Reserved Always write zero to this field.
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes the data register that stores the value to be driven on the pad in output mode.
14 MPPDO14	Masked Parallel Pad Data Out 14 Writes the data register that stores the value to be driven on the pad in output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes the data register that stores the value to be driven on the pad in output mode.
12 MPPDO12	Masked Parallel Pad Data Out 12 Writes the data register that stores the value to be driven on the pad in output mode.
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes the data register that stores the value to be driven on the pad in output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes the data register that stores the value to be driven on the pad in output mode.
9 —	Reserved Always write zero to this field.
8 —	Reserved Always write zero to this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 —	Reserved Always write zero to this field.
6 —	Reserved Always write zero to this field.
5 —	Reserved Always write zero to this field.
4 —	Reserved Always write zero to this field.
3 —	Reserved Always write zero to this field.
2 —	Reserved Always write zero to this field.
1 —	Reserved Always write zero to this field.
0 —	Reserved Always write zero to this field.

16.4 SIUL2_1 register descriptions

This section describes the SIUL2 registers.

NOTE

- Undocumented register spaces in the SIUL2 memory map, including addresses shown as blanks, are Reserved.
  - Reserved registers or spaces are read as 0.
  - Writes to Reserved registers or spaces generate a transfer error.
- Writes to read-only registers generate a transfer error.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

NOTE

- For the array of 8-bit registers GPDO*n* and GPDI*n*:
  - An 8-bit access to an unimplemented address (a "hole") within the array region will generate a transfer error.
  - However, if you do a 16-bit or a 32-bit access and if any register instance is implemented within the accessed range, a transfer error will not be generated even if the range includes a hole.
- For the array of 16-bit registers PGPDO*n* and PGPDI*n*:
  - A 16-bit access to an unimplemented address (a "hole") within the array region will generate a transfer error.
  - However, if you do a 32-bit access and if a register instance is implemented on the other 16-bit range, a transfer error will not be generated even if the range includes a hole.

16.4.1 SIUL2_1 memory map

SIUL2_1 base address: 4401_0000h

Offset	Register	Width (In bits)	Access	Reset value
4h	SIUL2 MCU ID Register #1 (MIDR1)	32	RO	4C20_0412h
8h	SIUL2 MCU ID Register #2 (MIDR2)	32	RO	0112_C000h
10h	SIUL2 DMA/Interrupt Status Flag Register0 (DISR0)	32	W1C	0000_0000h
18h	SIUL2 DMA/Interrupt Request Enable Register0 (DIRER0)	32	RW	0000_0000h
20h	SIUL2 DMA/Interrupt Request Select Register0 (DIRSR0)	32	RW	0000_0000h
28h	SIUL2 Interrupt Rising-Edge Event Enable Register 0 (IREER0)	32	RW	0000_0000h
30h	SIUL2 Interrupt Falling-Edge Event Enable Register 0 (IFEER0)	32	RW	0000_0000h
38h	SIUL2 Interrupt Filter Enable Register 0 (IFER0)	32	RW	0000_0000h
40h - BCh	SIUL2 Interrupt Filter Maximum Counter Register (IFMCR0 - IFMCR31)	32	RW	0000_0000h
C0h	SIUL2 Interrupt Filter Clock Prescaler Register (IFCPR)	32	RW	0000_0000h
400h	SIUL2 Multiplexed Signal Configuration Register (MSCR112)	32	RW	See description
404h	SIUL2 Multiplexed Signal Configuration Register (MSCR113)	32	RW	See description
408h	SIUL2 Multiplexed Signal Configuration Register (MSCR114)	32	RW	See description
40Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR115)	32	RW	See description
410h	SIUL2 Multiplexed Signal Configuration Register (MSCR116)	32	RW	See description
414h	SIUL2 Multiplexed Signal Configuration Register (MSCR117)	32	RW	See description

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
418h	SIUL2 Multiplexed Signal Configuration Register (MSCR118)	32	RW	See description
41Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR119)	32	RW	See description
420h	SIUL2 Multiplexed Signal Configuration Register (MSCR120)	32	RW	See description
424h	SIUL2 Multiplexed Signal Configuration Register (MSCR121)	32	RW	See description
428h	SIUL2 Multiplexed Signal Configuration Register (MSCR122)	32	RW	See description
480h	SIUL2 Multiplexed Signal Configuration Register (MSCR144)	32	RW	See description
484h	SIUL2 Multiplexed Signal Configuration Register (MSCR145)	32	RW	See description
488h	SIUL2 Multiplexed Signal Configuration Register (MSCR146)	32	RW	See description
48Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR147)	32	RW	See description
490h	SIUL2 Multiplexed Signal Configuration Register (MSCR148)	32	RW	See description
494h	SIUL2 Multiplexed Signal Configuration Register (MSCR149)	32	RW	See description
498h	SIUL2 Multiplexed Signal Configuration Register (MSCR150)	32	RW	See description
49Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR151)	32	RW	See description
4A0h	SIUL2 Multiplexed Signal Configuration Register (MSCR152)	32	RW	See description
4A4h	SIUL2 Multiplexed Signal Configuration Register (MSCR153)	32	RW	See description
4A8h	SIUL2 Multiplexed Signal Configuration Register (MSCR154)	32	RW	See description
4ACh	SIUL2 Multiplexed Signal Configuration Register (MSCR155)	32	RW	See description
4B0h	SIUL2 Multiplexed Signal Configuration Register (MSCR156)	32	RW	See description

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4B4h	SIUL2 Multiplexed Signal Configuration Register (MSCR157)	32	RW	See description
4B8h	SIUL2 Multiplexed Signal Configuration Register (MSCR158)	32	RW	See description
4BCh	SIUL2 Multiplexed Signal Configuration Register (MSCR159)	32	RW	See description
4C0h	SIUL2 Multiplexed Signal Configuration Register (MSCR160)	32	RW	See description
4C4h	SIUL2 Multiplexed Signal Configuration Register (MSCR161)	32	RW	See description
4C8h	SIUL2 Multiplexed Signal Configuration Register (MSCR162)	32	RW	See description
4CCh	SIUL2 Multiplexed Signal Configuration Register (MSCR163)	32	RW	See description
4D0h	SIUL2 Multiplexed Signal Configuration Register (MSCR164)	32	RW	See description
4D4h	SIUL2 Multiplexed Signal Configuration Register (MSCR165)	32	RW	See description
4D8h	SIUL2 Multiplexed Signal Configuration Register (MSCR166)	32	RW	See description
4DCh	SIUL2 Multiplexed Signal Configuration Register (MSCR167)	32	RW	See description
4E0h	SIUL2 Multiplexed Signal Configuration Register (MSCR168)	32	RW	See description
4E4h	SIUL2 Multiplexed Signal Configuration Register (MSCR169)	32	RW	See description
4E8h	SIUL2 Multiplexed Signal Configuration Register (MSCR170)	32	RW	See description
4ECh	SIUL2 Multiplexed Signal Configuration Register (MSCR171)	32	RW	See description
4F0h	SIUL2 Multiplexed Signal Configuration Register (MSCR172)	32	RW	See description
4F4h	SIUL2 Multiplexed Signal Configuration Register (MSCR173)	32	RW	See description
4F8h	SIUL2 Multiplexed Signal Configuration Register (MSCR174)	32	RW	See description

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4FCh	SIUL2 Multiplexed Signal Configuration Register (MSCR175)	32	RW	See description
500h	SIUL2 Multiplexed Signal Configuration Register (MSCR176)	32	RW	See description
504h	SIUL2 Multiplexed Signal Configuration Register (MSCR177)	32	RW	See description
508h	SIUL2 Multiplexed Signal Configuration Register (MSCR178)	32	RW	See description
50Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR179)	32	RW	See description
510h	SIUL2 Multiplexed Signal Configuration Register (MSCR180)	32	RW	See description
514h	SIUL2 Multiplexed Signal Configuration Register (MSCR181)	32	RW	See description
518h	SIUL2 Multiplexed Signal Configuration Register (MSCR182)	32	RW	See description
51Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR183)	32	RW	See description
520h	SIUL2 Multiplexed Signal Configuration Register (MSCR184)	32	RW	See description
524h	SIUL2 Multiplexed Signal Configuration Register (MSCR185)	32	RW	See description
528h	SIUL2 Multiplexed Signal Configuration Register (MSCR186)	32	RW	See description
52Ch	SIUL2 Multiplexed Signal Configuration Register (MSCR187)	32	RW	See description
530h	SIUL2 Multiplexed Signal Configuration Register (MSCR188)	32	RW	See description
534h	SIUL2 Multiplexed Signal Configuration Register (MSCR189)	32	RW	See description
538h	SIUL2 Multiplexed Signal Configuration Register (MSCR190)	32	RW	See description
C1Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR119)	32	RW	0000_0000h
C20h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR120)	32	RW	0000_0000h
C24h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR121)	32	RW	0000_0000h
C40h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR128)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
C44h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR129)	32	RW	0000_0000h
C7Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR143)	32	RW	0000_0000h
C80h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR144)	32	RW	0000_0000h
C84h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR145)	32	RW	0000_0000h
C88h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR146)	32	RW	0000_0000h
C8Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR147)	32	RW	0000_0000h
C90h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR148)	32	RW	0000_0000h
C94h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR149)	32	RW	0000_0000h
C98h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR150)	32	RW	0000_0000h
C9Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR151)	32	RW	0000_0000h
CA4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR153)	32	RW	0000_0000h
CA8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR154)	32	RW	0000_0000h
CACh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR155)	32	RW	0000_0000h
CB0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR156)	32	RW	0000_0000h
CB4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR157)	32	RW	0000_0000h
CB8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR158)	32	RW	0000_0000h
CBCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR159)	32	RW	0000_0000h
CC0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR160)	32	RW	0000_0000h
CC4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR161)	32	RW	0000_0000h
D74h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR205)	32	RW	0000_0000h
D78h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR206)	32	RW	0000_0000h
D7Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR207)	32	RW	0000_0000h
D80h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR208)	32	RW	0000_0000h
D84h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR209)	32	RW	0000_0000h
D88h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR210)	32	RW	0000_0000h
D8Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR211)	32	RW	0000_0000h
D90h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR212)	32	RW	0000_0000h
DC0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR224)	32	RW	0000_0000h
DC4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR225)	32	RW	0000_0000h
DE4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR233)	32	RW	0000_0000h
DE8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR234)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
DECh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR235)	32	RW	0000_0000h
DF0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR236)	32	RW	0000_0000h
DF4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR237)	32	RW	0000_0000h
DF8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR238)	32	RW	0000_0000h
DFCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR239)	32	RW	0000_0000h
E00h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR240)	32	RW	0000_0000h
E04h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR241)	32	RW	0000_0000h
E08h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR242)	32	RW	0000_0000h
E0Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR243)	32	RW	0000_0000h
E10h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR244)	32	RW	0000_0000h
E14h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR245)	32	RW	0000_0000h
E18h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR246)	32	RW	0000_0000h
E1Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR247)	32	RW	0000_0000h
E20h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR248)	32	RW	0000_0000h
E84h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR273)	32	RW	0000_0000h
E88h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR274)	32	RW	0000_0000h
E98h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR278)	32	RW	0000_0000h
E9Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR279)	32	RW	0000_0000h
EA0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR280)	32	RW	0000_0000h
EA4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR281)	32	RW	0000_0000h
EACh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR283)	32	RW	0000_0000h
EB0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR284)	32	RW	0000_0000h
EB4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR285)	32	RW	0000_0000h
EB8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR286)	32	RW	0000_0000h
EC0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR288)	32	RW	0000_0000h
EC4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR289)	32	RW	0000_0000h
EC8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR290)	32	RW	0000_0000h
ECCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR291)	32	RW	0000_0000h
ED0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR292)	32	RW	0000_0000h
ED4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR293)	32	RW	0000_0000h
ED8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR294)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
EE0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR296)	32	RW	0000_0000h
EE4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR297)	32	RW	0000_0000h
EE8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR298)	32	RW	0000_0000h
EECh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR299)	32	RW	0000_0000h
EF0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR300)	32	RW	0000_0000h
EF4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR301)	32	RW	0000_0000h
EF8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR302)	32	RW	0000_0000h
F00h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR304)	32	RW	0000_0000h
F04h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR305)	32	RW	0000_0000h
F08h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR306)	32	RW	0000_0000h
F0Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR307)	32	RW	0000_0000h
F10h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR308)	32	RW	0000_0000h
F14h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR309)	32	RW	0000_0000h
F18h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR310)	32	RW	0000_0000h
F20h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR312)	32	RW	0000_0000h
F24h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR313)	32	RW	0000_0000h
F28h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR314)	32	RW	0000_0000h
F30h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR316)	32	RW	0000_0000h
F38h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR318)	32	RW	0000_0000h
F48h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR322)	32	RW	0000_0000h
F4Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR323)	32	RW	0000_0000h
F50h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR324)	32	RW	0000_0000h
F54h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR325)	32	RW	0000_0000h
F58h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR326)	32	RW	0000_0000h
F5Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR327)	32	RW	0000_0000h
F60h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR328)	32	RW	0000_0000h
F64h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR329)	32	RW	0000_0000h
F68h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR330)	32	RW	0000_0000h
F6Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR331)	32	RW	0000_0000h
F70h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR332)	32	RW	0000_0000h
F74h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR333)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
F78h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR334)	32	RW	0000_0000h
F7Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR335)	32	RW	0000_0000h
F80h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR336)	32	RW	0000_0000h
F84h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR337)	32	RW	0000_0000h
F88h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR338)	32	RW	0000_0000h
F8Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR339)	32	RW	0000_0000h
F90h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR340)	32	RW	0000_0000h
F9Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR343)	32	RW	0000_0000h
FA0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR344)	32	RW	0000_0000h
FA4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR345)	32	RW	0000_0000h
FA8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR346)	32	RW	0000_0000h
FACh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR347)	32	RW	0000_0000h
FB0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR348)	32	RW	0000_0000h
FB4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR349)	32	RW	0000_0000h
FB8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR350)	32	RW	0000_0000h
FBCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR351)	32	RW	0000_0000h
FC0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR352)	32	RW	0000_0000h
FC4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR353)	32	RW	0000_0000h
FC8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR354)	32	RW	0000_0000h
FCCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR355)	32	RW	0000_0000h
FD0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR356)	32	RW	0000_0000h
FD4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR357)	32	RW	0000_0000h
FD8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR358)	32	RW	0000_0000h
FDCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR359)	32	RW	0000_0000h
FE0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR360)	32	RW	0000_0000h
FECh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR363)	32	RW	0000_0000h
FF0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR364)	32	RW	0000_0000h
FF4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR365)	32	RW	0000_0000h
FF8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR366)	32	RW	0000_0000h
FFCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR367)	32	RW	0000_0000h
1000h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR368)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1004h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR369)	32	RW	0000_0000h
1008h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR370)	32	RW	0000_0000h
100Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR371)	32	RW	0000_0000h
1010h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR372)	32	RW	0000_0000h
1014h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR373)	32	RW	0000_0000h
1018h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR374)	32	RW	0000_0000h
101Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR375)	32	RW	0000_0000h
1020h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR376)	32	RW	0000_0000h
1024h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR377)	32	RW	0000_0000h
1028h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR378)	32	RW	0000_0000h
102Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR379)	32	RW	0000_0000h
1030h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR380)	32	RW	0000_0000h
103Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR383)	32	RW	0000_0000h
1040h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR384)	32	RW	0000_0000h
1044h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR385)	32	RW	0000_0000h
1048h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR386)	32	RW	0000_0000h
104Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR387)	32	RW	0000_0000h
1050h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR388)	32	RW	0000_0000h
1054h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR389)	32	RW	0000_0000h
1058h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR390)	32	RW	0000_0000h
105Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR391)	32	RW	0000_0000h
1060h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR392)	32	RW	0000_0000h
1064h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR393)	32	RW	0000_0000h
1078h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR398)	32	RW	0000_0000h
107Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR399)	32	RW	0000_0000h
1080h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR400)	32	RW	0000_0000h
1084h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR401)	32	RW	0000_0000h
1088h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR402)	32	RW	0000_0000h
108Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR403)	32	RW	0000_0000h
1090h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR404)	32	RW	0000_0000h
1094h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR405)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1098h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR406)	32	RW	0000_0000h
109Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR407)	32	RW	0000_0000h
10A0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR408)	32	RW	0000_0000h
10A4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR409)	32	RW	0000_0000h
10A8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR410)	32	RW	0000_0000h
10ACh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR411)	32	RW	0000_0000h
10B0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR412)	32	RW	0000_0000h
10B4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR413)	32	RW	0000_0000h
10B8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR414)	32	RW	0000_0000h
10BCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR415)	32	RW	0000_0000h
10C0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR416)	32	RW	0000_0000h
10C4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR417)	32	RW	0000_0000h
10C8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR418)	32	RW	0000_0000h
10CCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR419)	32	RW	0000_0000h
10D0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR420)	32	RW	0000_0000h
10D4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR421)	32	RW	0000_0000h
10D8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR422)	32	RW	0000_0000h
10DCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR423)	32	RW	0000_0000h
10E0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR424)	32	RW	0000_0000h
10E4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR425)	32	RW	0000_0000h
10E8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR426)	32	RW	0000_0000h
10ECh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR427)	32	RW	0000_0000h
10F0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR428)	32	RW	0000_0000h
10F4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR429)	32	RW	0000_0000h
10F8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR430)	32	RW	0000_0000h
10FCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR431)	32	RW	0000_0000h
1100h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR432)	32	RW	0000_0000h
1104h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR433)	32	RW	0000_0000h
118Ch	SIUL2 Input Multiplexed Signal Configuration Register (IMCR467)	32	RW	0000_0000h
1190h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR468)	32	RW	0000_0000h
1194h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR469)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1198h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR470)	32	RW	0000_0000h
11A4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR473)	32	RW	0000_0000h
11A8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR474)	32	RW	0000_0000h
11ACh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR475)	32	RW	0000_0000h
11B8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR478)	32	RW	0000_0000h
11BCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR479)	32	RW	0000_0000h
11C0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR480)	32	RW	0000_0000h
11CCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR483)	32	RW	0000_0000h
11D0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR484)	32	RW	0000_0000h
11D4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR485)	32	RW	0000_0000h
11E0h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR488)	32	RW	0000_0000h
11E4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR489)	32	RW	0000_0000h
11E8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR490)	32	RW	0000_0000h
11F4h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR493)	32	RW	0000_0000h
11F8h	SIUL2 Input Multiplexed Signal Configuration Register (IMCR494)	32	RW	0000_0000h
11FCh	SIUL2 Input Multiplexed Signal Configuration Register (IMCR495)	32	RW	0000_0000h
1370h	SIUL2 GPIO Pad Data Output Register (GPDO115)	8	RW	00h
1371h	SIUL2 GPIO Pad Data Output Register (GPDO114)	8	RW	00h
1372h	SIUL2 GPIO Pad Data Output Register (GPDO113)	8	RW	00h
1373h	SIUL2 GPIO Pad Data Output Register (GPDO112)	8	RW	00h
1374h	SIUL2 GPIO Pad Data Output Register (GPDO119)	8	RW	00h
1375h	SIUL2 GPIO Pad Data Output Register (GPDO118)	8	RW	00h
1376h	SIUL2 GPIO Pad Data Output Register (GPDO117)	8	RW	00h
1377h	SIUL2 GPIO Pad Data Output Register (GPDO116)	8	RW	00h
1379h	SIUL2 GPIO Pad Data Output Register (GPDO122)	8	RW	00h
137Ah	SIUL2 GPIO Pad Data Output Register (GPDO121)	8	RW	00h
137Bh	SIUL2 GPIO Pad Data Output Register (GPDO120)	8	RW	00h
1390h	SIUL2 GPIO Pad Data Output Register (GPDO147)	8	RW	00h
1391h	SIUL2 GPIO Pad Data Output Register (GPDO146)	8	RW	00h
1392h	SIUL2 GPIO Pad Data Output Register (GPDO145)	8	RW	00h
1393h	SIUL2 GPIO Pad Data Output Register (GPDO144)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1394h	SIUL2 GPIO Pad Data Output Register (GPDO151)	8	RW	00h
1395h	SIUL2 GPIO Pad Data Output Register (GPDO150)	8	RW	00h
1396h	SIUL2 GPIO Pad Data Output Register (GPDO149)	8	RW	00h
1397h	SIUL2 GPIO Pad Data Output Register (GPDO148)	8	RW	00h
1398h	SIUL2 GPIO Pad Data Output Register (GPDO155)	8	RW	00h
1399h	SIUL2 GPIO Pad Data Output Register (GPDO154)	8	RW	00h
139Ah	SIUL2 GPIO Pad Data Output Register (GPDO153)	8	RW	00h
139Bh	SIUL2 GPIO Pad Data Output Register (GPDO152)	8	RW	00h
139Ch	SIUL2 GPIO Pad Data Output Register (GPDO159)	8	RW	00h
139Dh	SIUL2 GPIO Pad Data Output Register (GPDO158)	8	RW	00h
139Eh	SIUL2 GPIO Pad Data Output Register (GPDO157)	8	RW	00h
139Fh	SIUL2 GPIO Pad Data Output Register (GPDO156)	8	RW	00h
13A0h	SIUL2 GPIO Pad Data Output Register (GPDO163)	8	RW	00h
13A1h	SIUL2 GPIO Pad Data Output Register (GPDO162)	8	RW	00h
13A2h	SIUL2 GPIO Pad Data Output Register (GPDO161)	8	RW	00h
13A3h	SIUL2 GPIO Pad Data Output Register (GPDO160)	8	RW	00h
13A4h	SIUL2 GPIO Pad Data Output Register (GPDO167)	8	RW	00h
13A5h	SIUL2 GPIO Pad Data Output Register (GPDO166)	8	RW	00h
13A6h	SIUL2 GPIO Pad Data Output Register (GPDO165)	8	RW	00h
13A7h	SIUL2 GPIO Pad Data Output Register (GPDO164)	8	RW	00h
13A8h	SIUL2 GPIO Pad Data Output Register (GPDO171)	8	RW	00h
13A9h	SIUL2 GPIO Pad Data Output Register (GPDO170)	8	RW	00h
13AAh	SIUL2 GPIO Pad Data Output Register (GPDO169)	8	RW	00h
13ABh	SIUL2 GPIO Pad Data Output Register (GPDO168)	8	RW	00h
13ACh	SIUL2 GPIO Pad Data Output Register (GPDO175)	8	RW	00h
13ADh	SIUL2 GPIO Pad Data Output Register (GPDO174)	8	RW	00h
13AEh	SIUL2 GPIO Pad Data Output Register (GPDO173)	8	RW	00h
13AFh	SIUL2 GPIO Pad Data Output Register (GPDO172)	8	RW	00h
13B0h	SIUL2 GPIO Pad Data Output Register (GPDO179)	8	RW	00h
13B1h	SIUL2 GPIO Pad Data Output Register (GPDO178)	8	RW	00h
13B2h	SIUL2 GPIO Pad Data Output Register (GPDO177)	8	RW	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
13B3h	SIUL2 GPIO Pad Data Output Register (GPDO176)	8	RW	00h
13B4h	SIUL2 GPIO Pad Data Output Register (GPDO183)	8	RW	00h
13B5h	SIUL2 GPIO Pad Data Output Register (GPDO182)	8	RW	00h
13B6h	SIUL2 GPIO Pad Data Output Register (GPDO181)	8	RW	00h
13B7h	SIUL2 GPIO Pad Data Output Register (GPDO180)	8	RW	00h
13B8h	SIUL2 GPIO Pad Data Output Register (GPDO187)	8	RW	00h
13B9h	SIUL2 GPIO Pad Data Output Register (GPDO186)	8	RW	00h
13BAh	SIUL2 GPIO Pad Data Output Register (GPDO185)	8	RW	00h
13BBh	SIUL2 GPIO Pad Data Output Register (GPDO184)	8	RW	00h
13BDh	SIUL2 GPIO Pad Data Output Register (GPDO190)	8	RW	00h
13BEh	SIUL2 GPIO Pad Data Output Register (GPDO189)	8	RW	00h
13BFh	SIUL2 GPIO Pad Data Output Register (GPDO188)	8	RW	00h
1570h	SIUL2 GPIO Pad Data Input Register (GPDI115)	8	RO	00h
1571h	SIUL2 GPIO Pad Data Input Register (GPDI114)	8	RO	00h
1572h	SIUL2 GPIO Pad Data Input Register (GPDI113)	8	RO	00h
1573h	SIUL2 GPIO Pad Data Input Register (GPDI112)	8	RO	00h
1574h	SIUL2 GPIO Pad Data Input Register (GPDI119)	8	RO	00h
1575h	SIUL2 GPIO Pad Data Input Register (GPDI118)	8	RO	00h
1576h	SIUL2 GPIO Pad Data Input Register (GPDI117)	8	RO	00h
1577h	SIUL2 GPIO Pad Data Input Register (GPDI116)	8	RO	00h
1579h	SIUL2 GPIO Pad Data Input Register (GPDI122)	8	RO	00h
157Ah	SIUL2 GPIO Pad Data Input Register (GPDI121)	8	RO	00h
157Bh	SIUL2 GPIO Pad Data Input Register (GPDI120)	8	RO	00h
1590h	SIUL2 GPIO Pad Data Input Register (GPDI147)	8	RO	00h
1591h	SIUL2 GPIO Pad Data Input Register (GPDI146)	8	RO	00h
1592h	SIUL2 GPIO Pad Data Input Register (GPDI145)	8	RO	00h
1593h	SIUL2 GPIO Pad Data Input Register (GPDI144)	8	RO	00h
1594h	SIUL2 GPIO Pad Data Input Register (GPDI151)	8	RO	00h
1595h	SIUL2 GPIO Pad Data Input Register (GPDI150)	8	RO	00h
1596h	SIUL2 GPIO Pad Data Input Register (GPDI149)	8	RO	00h
1597h	SIUL2 GPIO Pad Data Input Register (GPDI148)	8	RO	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1598h	SIUL2 GPIO Pad Data Input Register (GPDI155)	8	RO	00h
1599h	SIUL2 GPIO Pad Data Input Register (GPDI154)	8	RO	00h
159Ah	SIUL2 GPIO Pad Data Input Register (GPDI153)	8	RO	00h
159Bh	SIUL2 GPIO Pad Data Input Register (GPDI152)	8	RO	00h
159Ch	SIUL2 GPIO Pad Data Input Register (GPDI159)	8	RO	00h
159Dh	SIUL2 GPIO Pad Data Input Register (GPDI158)	8	RO	00h
159Eh	SIUL2 GPIO Pad Data Input Register (GPDI157)	8	RO	00h
159Fh	SIUL2 GPIO Pad Data Input Register (GPDI156)	8	RO	00h
15A0h	SIUL2 GPIO Pad Data Input Register (GPDI163)	8	RO	00h
15A1h	SIUL2 GPIO Pad Data Input Register (GPDI162)	8	RO	00h
15A2h	SIUL2 GPIO Pad Data Input Register (GPDI161)	8	RO	00h
15A3h	SIUL2 GPIO Pad Data Input Register (GPDI160)	8	RO	00h
15A4h	SIUL2 GPIO Pad Data Input Register (GPDI167)	8	RO	00h
15A5h	SIUL2 GPIO Pad Data Input Register (GPDI166)	8	RO	00h
15A6h	SIUL2 GPIO Pad Data Input Register (GPDI165)	8	RO	00h
15A7h	SIUL2 GPIO Pad Data Input Register (GPDI164)	8	RO	00h
15A8h	SIUL2 GPIO Pad Data Input Register (GPDI171)	8	RO	00h
15A9h	SIUL2 GPIO Pad Data Input Register (GPDI170)	8	RO	00h
15AAh	SIUL2 GPIO Pad Data Input Register (GPDI169)	8	RO	00h
15ABh	SIUL2 GPIO Pad Data Input Register (GPDI168)	8	RO	00h
15ACh	SIUL2 GPIO Pad Data Input Register (GPDI175)	8	RO	00h
15ADh	SIUL2 GPIO Pad Data Input Register (GPDI174)	8	RO	00h
15AEh	SIUL2 GPIO Pad Data Input Register (GPDI173)	8	RO	00h
15AFh	SIUL2 GPIO Pad Data Input Register (GPDI172)	8	RO	00h
15B0h	SIUL2 GPIO Pad Data Input Register (GPDI179)	8	RO	00h
15B1h	SIUL2 GPIO Pad Data Input Register (GPDI178)	8	RO	00h
15B2h	SIUL2 GPIO Pad Data Input Register (GPDI177)	8	RO	00h
15B3h	SIUL2 GPIO Pad Data Input Register (GPDI176)	8	RO	00h
15B4h	SIUL2 GPIO Pad Data Input Register (GPDI183)	8	RO	00h
15B5h	SIUL2 GPIO Pad Data Input Register (GPDI182)	8	RO	00h
15B6h	SIUL2 GPIO Pad Data Input Register (GPDI181)	8	RO	00h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
15B7h	SIUL2 GPIO Pad Data Input Register (GPDI180)	8	RO	00h
15B8h	SIUL2 GPIO Pad Data Input Register (GPDI187)	8	RO	00h
15B9h	SIUL2 GPIO Pad Data Input Register (GPDI186)	8	RO	00h
15BAh	SIUL2 GPIO Pad Data Input Register (GPDI185)	8	RO	00h
15BBh	SIUL2 GPIO Pad Data Input Register (GPDI184)	8	RO	00h
15BDh	SIUL2 GPIO Pad Data Input Register (GPDI190)	8	RO	00h
15BEh	SIUL2 GPIO Pad Data Input Register (GPDI189)	8	RO	00h
15BFh	SIUL2 GPIO Pad Data Input Register (GPDI188)	8	RO	00h
170Ch	SIUL2 Parallel GPIO Pad Data Out Register (PGPDO7)	16	RW	0000h
1710h	SIUL2 Parallel GPIO Pad Data Out Register (PGPDO9)	16	RW	0000h
1714h	SIUL2 Parallel GPIO Pad Data Out Register (PGPDO11)	16	RW	0000h
1716h	SIUL2 Parallel GPIO Pad Data Out Register (PGPDO10)	16	RW	0000h
174Ch	SIUL2 Parallel GPIO Pad Data In Register (PGPDI7)	16	RO	0000h
1750h	SIUL2 Parallel GPIO Pad Data In Register (PGPDI9)	16	RO	0000h
1754h	SIUL2 Parallel GPIO Pad Data In Register (PGPDI11)	16	RO	0000h
1756h	SIUL2 Parallel GPIO Pad Data In Register (PGPDI10)	16	RO	0000h
179Ch	SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO7)	32	WORZ	0000_0000h
17A4h - 17A8h	SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO9 - MPGPDO10)	32	WORZ	0000_0000h
17ACh	SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO11)	32	WORZ	0000_0000h

16.4.2 SIUL2 MCU ID Register #1 (MIDR1)

Offset

Register	Offset
MIDR1	4h

Function

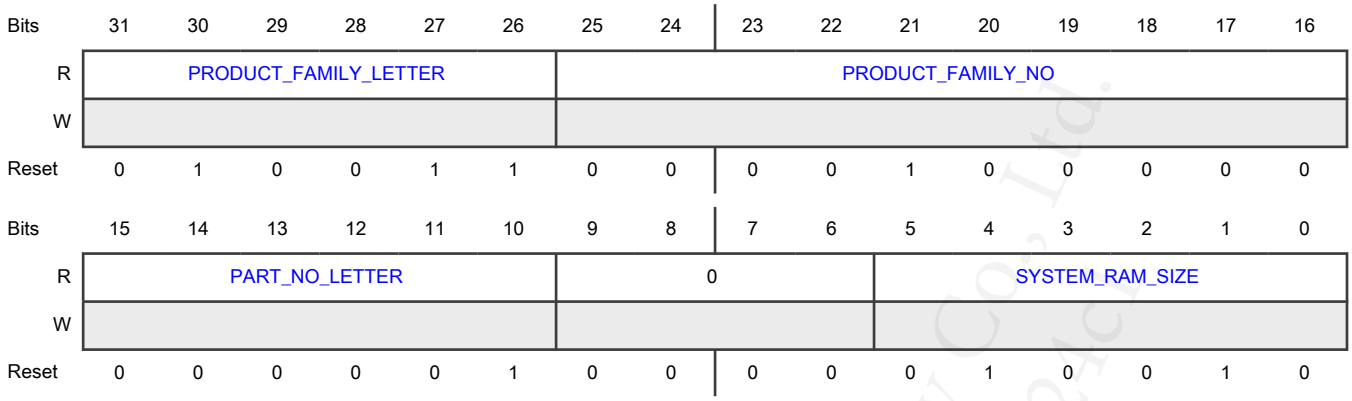
This register holds identification information about the device.

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

provided under NXP CONFIDENTIAL Beijing CHJ Automotive Technology Co.,Ltd. ca462e5-4821-48ab-b251-390daba6a61

Diagram



Fields

Field	Function
31-26 PRODUCT_FAMILY_LETTER	Product Family Letter Identifies the Product Family Letter. 010011b - S
25-16 PRODUCT_FAMILY_NO	Product Family Number 0000100000b - 32
15-10 PART_NO_LETTER	Part Number Letter Identifies the Part Number Letter. 000001b - A 000011b - C 001101b - M
9-6 —	Reserved
5-0 SYSTEM_RAM_SIZE	System RAM Size Identifies the system RAM size. 000010b - 128 kB 000011b - 192 kB 000100b - 256 kB 000101b - 384 kB 000110b - 512 kB 000111b - 768 kB 001000b - 1 MB

Table continues on the next page...

Table continued from the previous page...

Field	Function
	001001b - 1.25 MB 001010b - 1.5 MB 001011b - 1.75 MB 001100b - 2 MB 001101b - 2.5 MB 001110b - 3 MB 001111b - 3.5 MB 010000b - 4 MB 010001b - 6 MB 010010b - 8 MB 010011b - 10 MB 011100b - 12 MB 011101b - 14 MB 011110b - 16 MB 011111b - 18 MB

16.4.3 SIUL2 MCU ID Register #2 (MIDR2)

Offset

Register	Offset
MIDR2	8h

Function

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4871-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FLASH_VENDOR		FLASH_CONFIG		SUBMINOR_MASK		COMPUTE_DIE_PART_NUMBER									
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SERDES	OTA	HPBF	LAX	0										AE1	AE0
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 FLASH_VENDOR	Flash Vendor Identifies flash die vendor. 00b - None 01b - Macronix
29-28 FLASH_CONFIG	Flash Configuration Identifies flash die default I/O configuration. 00b - x1 I/O 01b - x8 I/O
27-26 SUBMINOR_MASK	Subminor Mask Revision 00b - Rev x.x.0 01b - Rev x.x.1
25-16 COMPUTE_DIE_PART_NUMBER	Compute Die Part Number
15 SERDES	SerDes subsystem Identifies whether the SerDes subsystem is present. 0b - No 1b - Yes
14 OTA	Over The Air Identifies if Over The Air is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No 1b - Yes
13 HPBF	High Performance Buffered Flash Identifies if High Performance Buffered Flash is enabled. 0b - No 1b - Yes
12 LAX	Linear Algebraic Accelerator Identifies if Linear Algebraic Accelerator is present. 0b - No 1b - Yes
11-2 —	Reserved
1 AE1	Application Extension 1 Identifies if Application Extension 1 is present. 0b - No 1b - Yes
0 AE0	Application Extension 0 Identifies if Application Extension 0 is present. 0b - No 1b - Yes

16.4.4 SIUL2 DMA/Interrupt Status Flag Register0 (DISR0)

Offset

Register	Offset
DISR0	10h

Function

DISR0 contains flag bits that record an event on the external IRQ pins. When an event as defined in IREER0 and IFEER0 occurs, the corresponding flag bit is set. The IRQ flag bit is set regardless of the state of the corresponding DIRER0[EIRE*n*] bit. The IRQ flag bit remains set until you clear it or is cleared by servicing of a DMA request. The IRQ flag bits are cleared by writing a 1 to the bits. A write of 0 has no effect.

This register supports 8-, 16-, and 32-bit accesses.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
86 62e5 4821 48a8 b251 390d 3ab6 24c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EIF31	EIF30	EIF29	EIF28	EIF27	EIF26	EIF25	EIF24	EIF23	EIF22	EIF21	EIF20	EIF19	EIF18	EIF17	EIF16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EIF15	EIF14	EIF13	EIF12	EIF11	EIF10	EIF9	EIF8	EIF7	EIF6	EIF5	EIF4	EIF3	EIF2	EIF1	EIF0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EIF31	External Interrupt Status Flag 31 If enabled (DIRERR31 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER31 and IFEER31 has occurred.
30 EIF30	External Interrupt Status Flag 30 If enabled (DIRERR30 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER30 and IFEER30 has occurred.
29 EIF29	External Interrupt Status Flag 29 If enabled (DIRERR29 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER29 and IFEER29 has occurred.
28 EIF28	External Interrupt Status Flag 28 If enabled (DIRERR28 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER28 and IFEER28 has occurred.
27 EIF27	External Interrupt Status Flag 27 If enabled (DIRERR27 = 1) causes an interrupt or DMA request.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER27 and IFEER27 has occurred.
26 EIF26	External Interrupt Status Flag 26 If enabled (DIRERR26 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER26 and IFEER26 has occurred.
25 EIF25	External Interrupt Status Flag 25 If enabled (DIRERR25 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER25 and IFEER25 has occurred.
24 EIF24	External Interrupt Status Flag 24 If enabled (DIRERR24 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER24 and IFEER24 has occurred.
23 EIF23	External Interrupt Status Flag 23 If enabled (DIRERR23 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER23 and IFEER23 has occurred.
22 EIF22	External Interrupt Status Flag 22 If enabled (DIRERR22 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER22 and IFEER22 has occurred.
21 EIF21	External Interrupt Status Flag 21 If enabled (DIRERR21 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER21 and IFEER21 has occurred.
20 EIF20	External Interrupt Status Flag 20 If enabled (DIRERR20 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER20 and IFEER20 has occurred.
19 EIF19	External Interrupt Status Flag 19 If enabled (DIRERR19 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER19 and IFEER19 has occurred.
18 EIF18	External Interrupt Status Flag 18 If enabled (DIRERR18 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER18 and IFEER18 has occurred.
17 EIF17	External Interrupt Status Flag 17 If enabled (DIRERR17 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER17 and IFEER17 has occurred.
16 EIF16	External Interrupt Status Flag 16 If enabled (DIRERR16 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER16 and IFEER16 has occurred.
15 EIF15	External Interrupt Status Flag 15 If enabled (DIRERR15 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - An interrupt event as defined by IREER15 and IFEER15 has occurred.
14 EIF14	External Interrupt Status Flag 14 If enabled (DIRERR14 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER14 and IFEER14 has occurred.
13 EIF13	External Interrupt Status Flag 13 If enabled (DIRERR13 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER13 and IFEER13 has occurred.
12 EIF12	External Interrupt Status Flag 12 If enabled (DIRERR12 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER12 and IFEER12 has occurred.
11 EIF11	External Interrupt Status Flag 11 If enabled (DIRERR11 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER11 and IFEER11 has occurred.
10 EIF10	External Interrupt Status Flag 10 If enabled (DIRERR10 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER10 and IFEER10 has occurred.
9 EIF9	External Interrupt Status Flag 9 If enabled (DIRERR9 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER9 and IFEER9 has occurred.

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 EIF8	External Interrupt Status Flag 8 If enabled (DIRERR8 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER8 and IFEER8 has occurred.
7 EIF7	External Interrupt Status Flag 7 If enabled (DIRERR7 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER7 and IFEER7 has occurred.
6 EIF6	External Interrupt Status Flag 6 If enabled (DIRERR6 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER6 and IFEER6 has occurred.
5 EIF5	External Interrupt Status Flag 5 If enabled (DIRERR5 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER5 and IFEER5 has occurred.
4 EIF4	External Interrupt Status Flag 4 If enabled (DIRERR4 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER4 and IFEER4 has occurred.
3 EIF3	External Interrupt Status Flag 3 If enabled (DIRERR3 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER3 and IFEER3 has occurred.
2	External Interrupt Status Flag 2 If enabled (DIRERR2 = 1) causes an interrupt or DMA request.

Table continues on the next page...

Table continued from the previous page...

Field	Function
EIF2	This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER2 and IFEER2 has occurred.
1 EIF1	External Interrupt Status Flag 1 If enabled (DIRERR1 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER1 and IFEER1 has occurred.
0 EIF0	External Interrupt Status Flag 0 If enabled (DIRERR0 = 1) causes an interrupt or DMA request. This flag can be cleared only by writing 1. Writing 0 has no effect. 0b - No interrupt event has occurred on the pad. 1b - An interrupt event as defined by IREER0 and IFEER0 has occurred.

16.4.5 SIUL2 DMA/Interrupt Request Enable Register0 (DIRER0)

Offset

Register	Offset
DIRER0	18h

Function

DIRER0 enables the assertion of DMA or interrupt request to the interrupt controller if the corresponding DISR0[EIF*n*] bit is set. The type of request is determined by the corresponding DIRSR0[DIRSR*n*] bit.

This register supports 8-, 16-, and 32-bit accesses.

NOTE

DMA requests cannot be enabled or disabled.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-e214-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EIRE31	EIRE30	EIRE29	EIRE28	EIRE27	EIRE26	EIRE25	EIRE24	EIRE23	EIRE22	EIRE21	EIRE20	EIRE19	EIRE18	EIRE17	EIRE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EIRE15	EIRE14	EIRE13	EIRE12	EIRE11	EIRE10	EIRE9	EIRE8	EIRE7	EIRE6	EIRE5	EIRE4	EIRE3	EIRE2	EIRE1	EIRE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EIRE31	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
30 EIRE30	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
29 EIRE29	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
28 EIRE28	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
27 EIRE27	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
26 EIRE26	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
25 EIRE25	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
24 EIRE24	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
23 EIRE23	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
22 EIRE22	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
21 EIRE21	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
20 EIRE20	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
19 EIRE19	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
18	External Interrupt Request Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
EIRE18	Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
17 EIRE17	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
16 EIRE16	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
15 EIRE15	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
14 EIRE14	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
13 EIRE13	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
12 EIRE12	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
11 EIRE11	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 EIRE10	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
9 EIRE9	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
8 EIRE8	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
7 EIRE7	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
6 EIRE6	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
5 EIRE5	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
4 EIRE4	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
3 EIRE3	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
84662e5_4821_48a8_b251_390d3ab624c1

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
2 EIRE2	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
1 EIRE1	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled
0 EIRE0	External Interrupt Request Enable Enables or disables interrupt requests from the corresponding pin. 0b - Disabled 1b - Enabled

16.4.6 SIUL2 DMA/Interrupt Request Select Register0 (DIRSR0)

Offset

Register	Offset
DIRSR0	20h

Function

DIRSR0 selects between the DMA or interrupt request. The DIRSR0[DIRSR*n*] bit determines whether DMA or an interrupt request asserts the corresponding DISR0[EIF*n*] bit.

This register supports 8-, 16-, and 32-bit accesses.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR
W	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR	DIRSR
W	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DIRSR31	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
30 DIRSR30	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
29 DIRSR29	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
28 DIRSR28	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
27 DIRSR27	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt request 1b - DMA request
26 DIRSR26	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
25 DIRSR25	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
24 DIRSR24	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
23 DIRSR23	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
22 DIRSR22	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
21 DIRSR21	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
20	DMA/Interrupt Request Select Register

Table continues on the next page...

Table continued from the previous page...

Field	Function
DIRSR20	Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
19 DIRSR19	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
18 DIRSR18	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
17 DIRSR17	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
16 DIRSR16	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
15 DIRSR15	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
14 DIRSR14	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 DIRSR13	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
12 DIRSR12	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
11 DIRSR11	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
10 DIRSR10	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
9 DIRSR9	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
8 DIRSR8	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
7 DIRSR7	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt request 1b - DMA request
6 DIRSR6	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
5 DIRSR5	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
4 DIRSR4	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
3 DIRSR3	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
2 DIRSR2	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
1 DIRSR1	DMA/Interrupt Request Select Register Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request
0	DMA/Interrupt Request Select Register

Table continues on the next page...

Table continued from the previous page...

Field	Function
DIRSR0	Selects between DMA request or external interrupt request when an edge-triggered event occurs on the corresponding pin. 0b - Interrupt request 1b - DMA request

16.4.7 SIUL2 Interrupt Rising-Edge Event Enable Register 0 (IREER0)

Offset

Register	Offset
IREER0	28h

Function

IREER0 enables the rising-edge triggered events on the corresponding interrupt pads.

This register supports 8-, 16-, and 32-bit accesses.

NOTE

If both the IREE and IFEE bits are cleared for the same interrupt source, the interrupt status flag for the corresponding external interrupt will never be set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IREE31	IREE30	IREE29	IREE28	IREE27	IREE26	IREE25	IREE24	IREE23	IREE22	IREE21	IREE20	IREE19	IREE18	IREE17	IREE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IREE15	IREE14	IREE13	IREE12	IREE11	IREE10	IREE9	IREE8	IREE7	IREE6	IREE5	IREE4	IREE3	IREE2	IREE1	IREE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 IREE31	Enables rising-edge events to set DISR0[EIF31]. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 IREE30	Enables rising-edge events to set DISR0[EIF30]. 0b - Disabled 1b - Enabled
29 IREE29	Enables rising-edge events to set DISR0[EIF29]. 0b - Disabled 1b - Enabled
28 IREE28	Enables rising-edge events to set DISR0[EIF28]. 0b - Disabled 1b - Enabled
27 IREE27	Enables rising-edge events to set DISR0[EIF27]. 0b - Disabled 1b - Enabled
26 IREE26	Enables rising-edge events to set DISR0[EIF26]. 0b - Disabled 1b - Enabled
25 IREE25	Enables rising-edge events to set DISR0[EIF25]. 0b - Disabled 1b - Enabled
24 IREE24	Enables rising-edge events to set DISR0[EIF24]. 0b - Disabled 1b - Enabled
23 IREE23	Enables rising-edge events to set DISR0[EIF23]. 0b - Disabled 1b - Enabled
22 IREE22	Enables rising-edge events to set DISR0[EIF22]. 0b - Disabled 1b - Enabled
21 IREE21	Enables rising-edge events to set DISR0[EIF21]. 0b - Disabled 1b - Enabled
20 IREE20	Enables rising-edge events to set DISR0[EIF20].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
19 IREE19	Enables rising-edge events to set DISR0[EIF19]. 0b - Disabled 1b - Enabled
18 IREE18	Enables rising-edge events to set DISR0[EIF18]. 0b - Disabled 1b - Enabled
17 IREE17	Enables rising-edge events to set DISR0[EIF17]. 0b - Disabled 1b - Enabled
16 IREE16	Enables rising-edge events to set DISR0[EIF16]. 0b - Disabled 1b - Enabled
15 IREE15	Enables rising-edge events to set DISR0[EIF15]. 0b - Disabled 1b - Enabled
14 IREE14	Enables rising-edge events to set DISR0[EIF14]. 0b - Disabled 1b - Enabled
13 IREE13	Enables rising-edge events to set DISR0[EIF13]. 0b - Disabled 1b - Enabled
12 IREE12	Enables rising-edge events to set DISR0[EIF12]. 0b - Disabled 1b - Enabled
11 IREE11	Enables rising-edge events to set DISR0[EIF11]. 0b - Disabled 1b - Enabled
10 IREE10	Enables rising-edge events to set DISR0[EIF10]. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 IREE9	Enables rising-edge events to set DISR0[EIF9]. 0b - Disabled 1b - Enabled
8 IREE8	Enables rising-edge events to set DISR0[EIF8]. 0b - Disabled 1b - Enabled
7 IREE7	Enables rising-edge events to set DISR0[EIF7]. 0b - Disabled 1b - Enabled
6 IREE6	Enables rising-edge events to set DISR0[EIF6]. 0b - Disabled 1b - Enabled
5 IREE5	Enables rising-edge events to set DISR0[EIF5]. 0b - Disabled 1b - Enabled
4 IREE4	Enables rising-edge events to set DISR0[EIF4]. 0b - Disabled 1b - Enabled
3 IREE3	Enables rising-edge events to set DISR0[EIF3]. 0b - Disabled 1b - Enabled
2 IREE2	Enables rising-edge events to set DISR0[EIF2]. 0b - Disabled 1b - Enabled
1 IREE1	Enables rising-edge events to set DISR0[EIF1]. 0b - Disabled 1b - Enabled
0 IREE0	Enables rising-edge events to set DISR0[EIF0]. 0b - Disabled 1b - Enabled

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

16.4.8 SIUL2 Interrupt Falling-Edge Event Enable Register 0 (IFEER0)

Offset

Register	Offset
IFEER0	30h

Function

IFEER0 enables falling-edge triggered events on the corresponding interrupt pads.

This register supports 8-, 16-, and 32-bit accesses.

NOTE

If both the IREE and IFEE bits are cleared for the same interrupt source, the interrupt status flag for the corresponding external interrupt will never be set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IFEE31	IFEE30	IFEE29	IFEE28	IFEE27	IFEE26	IFEE25	IFEE24	IFEE23	IFEE22	IFEE21	IFEE20	IFEE19	IFEE18	IFEE17	IFEE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IFEE15	IFEE14	IFEE13	IFEE12	IFEE11	IFEE10	IFEE9	IFEE8	IFEE7	IFEE6	IFEE5	IFEE4	IFEE3	IFEE2	IFEE1	IFEE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 IFEE31	Enables falling-edge events to set DISR0[EIF31]. 0b - Disabled 1b - Enabled
30 IFEE30	Enables falling-edge events to set DISR0[EIF30]. 0b - Disabled 1b - Enabled
29 IFEE29	Enables falling-edge events to set DISR0[EIF29]. 0b - Disabled 1b - Enabled
28	Enables falling-edge events to set DISR0[EIF28].

Table continues on the next page...

Table continued from the previous page...

Field	Function
IFEE28	0b - Disabled 1b - Enabled
27 IFEE27	Enables falling-edge events to set DISR0[EIF27]. 0b - Disabled 1b - Enabled
26 IFEE26	Enables falling-edge events to set DISR0[EIF26]. 0b - Disabled 1b - Enabled
25 IFEE25	Enables falling-edge events to set DISR0[EIF25]. 0b - Disabled 1b - Enabled
24 IFEE24	Enables falling-edge events to set DISR0[EIF24]. 0b - Disabled 1b - Enabled
23 IFEE23	Enables falling-edge events to set DISR0[EIF23]. 0b - Disabled 1b - Enabled
22 IFEE22	Enables falling-edge events to set DISR0[EIF22]. 0b - Disabled 1b - Enabled
21 IFEE21	Enables falling-edge events to set DISR0[EIF21]. 0b - Disabled 1b - Enabled
20 IFEE20	Enables falling-edge events to set DISR0[EIF20]. 0b - Disabled 1b - Enabled
19 IFEE19	Enables falling-edge events to set DISR0[EIF19]. 0b - Disabled 1b - Enabled
18 IFEE18	Enables falling-edge events to set DISR0[EIF18]. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 IFEE17	Enables falling-edge events to set DISR0[EIF17]. 0b - Disabled 1b - Enabled
16 IFEE16	Enables falling-edge events to set DISR0[EIF16]. 0b - Disabled 1b - Enabled
15 IFEE15	Enables falling-edge events to set DISR0[EIF15]. 0b - Disabled 1b - Enabled
14 IFEE14	Enables falling-edge events to set DISR0[EIF14]. 0b - Disabled 1b - Enabled
13 IFEE13	Enables falling-edge events to set DISR0[EIF13]. 0b - Disabled 1b - Enabled
12 IFEE12	Enables falling-edge events to set DISR0[EIF12]. 0b - Disabled 1b - Enabled
11 IFEE11	Enables falling-edge events to set DISR0[EIF11]. 0b - Disabled 1b - Enabled
10 IFEE10	Enables falling-edge events to set DISR0[EIF10]. 0b - Disabled 1b - Enabled
9 IFEE9	Enables falling-edge events to set DISR0[EIF9]. 0b - Disabled 1b - Enabled
8 IFEE8	Enables falling-edge events to set DISR0[EIF8]. 0b - Disabled 1b - Enabled
7 IFEE7	Enables falling-edge events to set DISR0[EIF7].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
6 IFEE6	Enables falling-edge events to set DISR0[EIF6]. 0b - Disabled 1b - Enabled
5 IFEE5	Enables falling-edge events to set DISR0[EIF5]. 0b - Disabled 1b - Enabled
4 IFEE4	Enables falling-edge events to set DISR0[EIF4]. 0b - Disabled 1b - Enabled
3 IFEE3	Enables falling-edge events to set DISR0[EIF3]. 0b - Disabled 1b - Enabled
2 IFEE2	Enables falling-edge events to set DISR0[EIF2]. 0b - Disabled 1b - Enabled
1 IFEE1	Enables falling-edge events to set DISR0[EIF1]. 0b - Disabled 1b - Enabled
0 IFEE0	Enables falling-edge events to set DISR0[EIF0]. 0b - Disabled 1b - Enabled

16.4.9 SIUL2 Interrupt Filter Enable Register 0 (IFER0)

Offset

Register	Offset
IFER0	38h

Function

IFER0 enables a digital filter counter on the corresponding interrupt pads to filter out glitches on the inputs.

This register supports 8-, 16-, and 32-bit accesses.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	IFE31	IFE30	IFE29	IFE28	IFE27	IFE26	IFE25	IFE24	IFE23	IFE22	IFE21	IFE20	IFE19	IFE18	IFE17	IFE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IFE15	IFE14	IFE13	IFE12	IFE11	IFE10	IFE9	IFE8	IFE7	IFE6	IFE5	IFE4	IFE3	IFE2	IFE1	IFE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 IFE31	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
30 IFE30	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
29 IFE29	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
28 IFE28	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
27 IFE27	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
26 IFE26	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
25 IFE25	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
24	Enables digital glitch filter on the interrupt pad input.

Table continues on the next page...

Table continued from the previous page...

Field	Function
IFE24	0b - Disabled 1b - Enabled
23 IFE23	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
22 IFE22	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
21 IFE21	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
20 IFE20	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
19 IFE19	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
18 IFE18	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
17 IFE17	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
16 IFE16	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
15 IFE15	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
14 IFE14	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 IFE13	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
12 IFE12	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
11 IFE11	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
10 IFE10	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
9 IFE9	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
8 IFE8	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
7 IFE7	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
6 IFE6	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
5 IFE5	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
4 IFE4	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
3 IFE3	Enables digital glitch filter on the interrupt pad input.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
2 IFE2	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
1 IFE1	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled
0 IFE0	Enables digital glitch filter on the interrupt pad input. 0b - Disabled 1b - Enabled

16.4.10 SIUL2 Interrupt Filter Maximum Counter Register (IFMCR0 - IFMCR31)

Offset

For a = 0 to 31:

Register	Offset
IFMCRa	40h + (a × 4h)

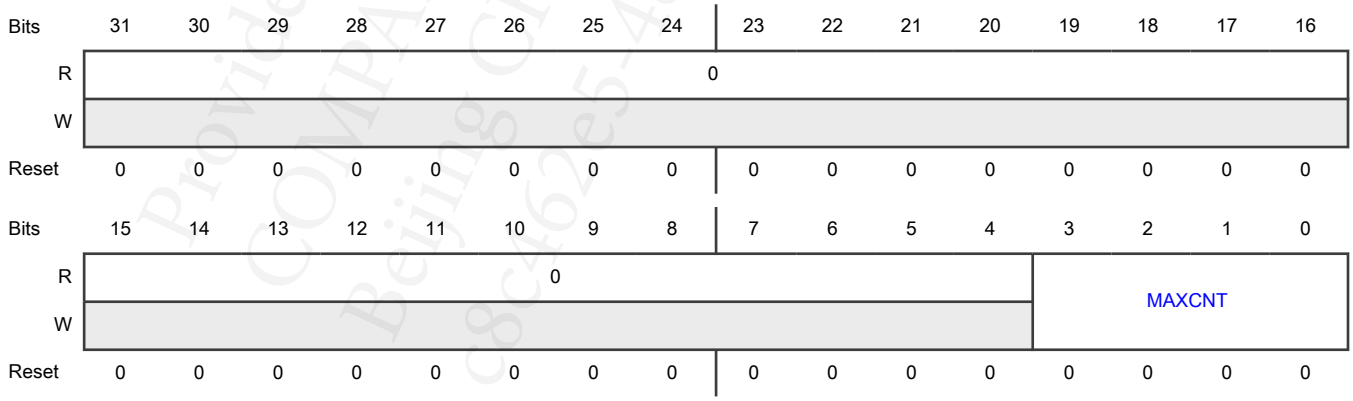
Function

IFMCR*n* registers configure the filter counter associated with each digital glitch filter.

NOTE

These registers support only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
08c462e5-4821-4a8-b251-300d3ab624c1

Fields

Field	Function
31-4 —	Reserved
3-0 MAXCNT	Maximum Interrupt Filter Counter setting MAXCNT can be 0d to 15d. • A value of 0d, 1d, or 2d sets the filter as an all pass filter. • A value of 3d to 15d sets the filter period to TCK × MAXCNT + *n* × TCK, where: — *n* is 0, 1, 2, 3, or 4. — TCK is the prescaled filter clock period, which is the IRC clock prescaled to the IFCP value specified in IFCPR.

16.4.11 SIUL2 Interrupt Filter Clock Prescaler Register (IFCPR)

Offset

Register	Offset
IFCPR	C0h

Function

IFCPR configures the clock prescaler which selects the clock for all digital filters. A prescaler is applied to the input clock to SIUL2, which is the peripheral clock counter in the SIUL2.

NOTE

This register supports only 32-bit accesses. Byte and half-word accesses are not supported.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												IFCP			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4824-4888-b251-390d3ab624c1

Fields

Field	Function
31-4 —	Reserved
3-0 IFCP	Interrupt Filter Clock Prescaler setting Prescaled Filter Clock period is T_{IRC} × (IFCP + 1), where: • T_{IRC} is the internal oscillator period. • IFCP is 0 to 15.

16.4.12 SIUL2 Multiplexed Signal Configuration Register (MSCR112 - MSCR190)

Offset

Register	Offset
MSCR112	400h
MSCR113	404h
MSCR114	408h
MSCR115	40Ch
MSCR116	410h
MSCR117	414h
MSCR118	418h
MSCR119	41Ch
MSCR120	420h
MSCR121	424h
MSCR122	428h
MSCR144	480h
MSCR145	484h
MSCR146	488h
MSCR147	48Ch
MSCR148	490h
MSCR149	494h
MSCR150	498h
MSCR151	49Ch
MSCR152	4A0h
MSCR153	4A4h

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Register	Offset
MSCR154	4A8h
MSCR155	4ACh
MSCR156	4B0h
MSCR157	4B4h
MSCR158	4B8h
MSCR159	4BCh
MSCR160	4C0h
MSCR161	4C4h
MSCR162	4C8h
MSCR163	4CCh
MSCR164	4D0h
MSCR165	4D4h
MSCR166	4D8h
MSCR167	4DCh
MSCR168	4E0h
MSCR169	4E4h
MSCR170	4E8h
MSCR171	4ECh
MSCR172	4F0h
MSCR173	4F4h
MSCR174	4F8h
MSCR175	4FCh
MSCR176	500h
MSCR177	504h
MSCR178	508h
MSCR179	50Ch
MSCR180	510h
MSCR181	514h
MSCR182	518h
MSCR183	51Ch
MSCR184	520h
MSCR185	524h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
MSCR186	528h
MSCR187	52Ch
MSCR188	530h
MSCR189	534h
MSCR190	538h

Function

MSCR*n* registers select the source signal connected to the register's associated destination, which is a chip output pin or a chip pin that can be configured as an output.

MSCR*n* also specifies the electrical properties of the associated pin.

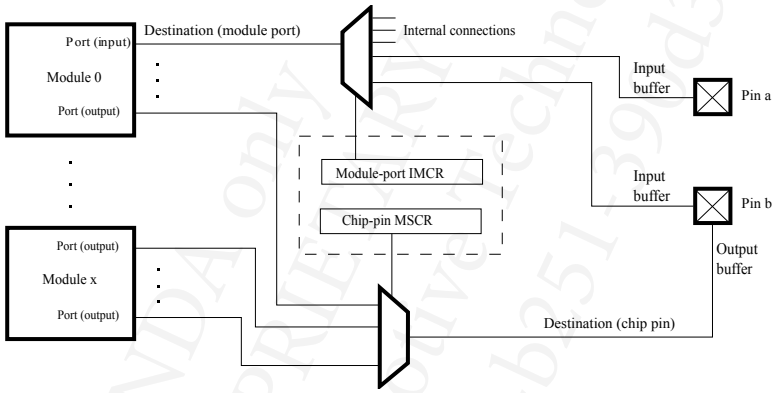


Figure 40. MSCR and IMCR port and pin connection

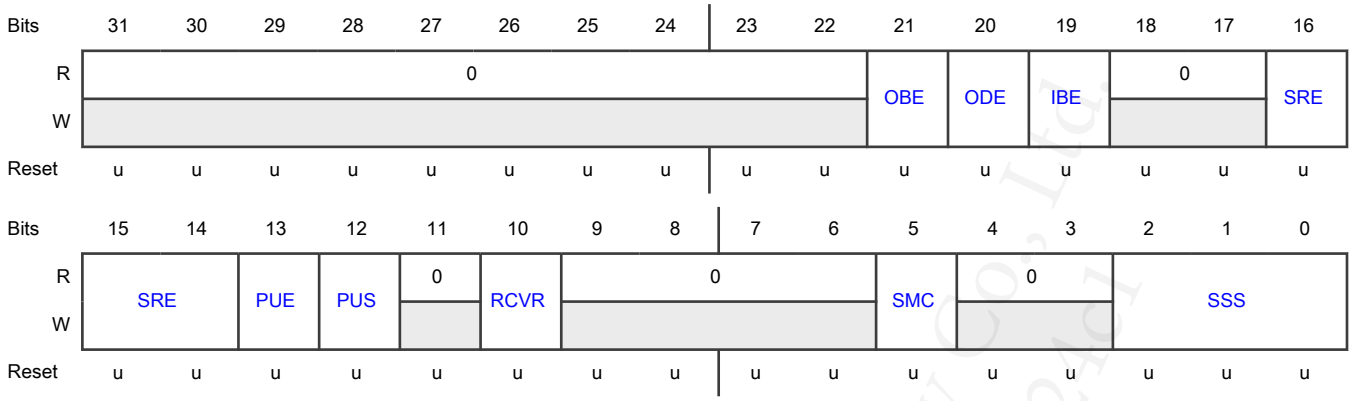
For chip-pin MSCR assignments and pin types, see the IOMUX file attached to this document.

NOTE

- MSCR*n* registers support only 32-bit accesses. Byte and half-word write accesses are not supported.
- MSCR*n* registers must be configured only during application initialization and must not be modified during application runtime.
- Accessing a reserved MSCR*n* register generates a transfer error.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4e21-48a8-b513-9d3ab624c1

Diagram



Fields

Field	Function
31-22 —	Reserved
21 OBE	GPIO Output Buffer Enable Applies only to digital pins. Otherwise this bit is reserved. 0b - Output driver disabled 1b - Output driver enabled
20 ODE	Open Drain Enable **NOTE** To enable open drain both OBE and ODE bits need to be set. 0b - Open drain function disabled 1b - Open drain function enabled when OBE is also 1
19 IBE	Input Buffer Enable Used only when the associated destination is a chip pin. Enables the associated pin's input buffer. 0b - Disabled 1b - Enabled
18-17 —	Reserved
16-14 SRE	Slew Rate Control 000b - Fastest setting, 208MHz 100b - 150MHz or lower 101b - 100MHz or lower

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - 50MHz or lower 111b - Slowest setting, 25MHz, for EMC applications
13 PUE	Pull Enable Enables the pull function. Used only when the associated destination is a chip pin. 0b - Disabled 1b - Enabled
12 PUS	Pull Select Determines whether the pull function is a pullup or pulldown when the pull function is enabled by the PUE field. Used only when the associated destination is a chip pin. 0b - Pull down 1b - Pull up
11 —	Reserved
10 RCVR	Receiver Select 0b - Enables the differential vref based receiver. 1b - Enables the single ended receiver.
9-6 —	Reserved
5 SMC	Safe Mode Control Used only when the associated destination is a chip pin. Specifies whether the chip disables the pin's output buffer when the chip enters Safe mode. 0b - Disable (The output buffer returns to its previous state when the chip leaves Safe mode.) 1b - Don't disable
4-3 —	Reserved
2-0 SSS	Source Signal Select Selects a function for the pad. Refer to “SSS” column of the ‘IO Signal Table' tab of the IOMUX spreadsheet attachment.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-4a8-b251-390d3a1624c1

16.4.13 SIUL2 Input Multiplexed Signal Configuration Register (IMCR119 - IMCR495)

Offset

Register	Offset
IMCR119	C1Ch
IMCR120	C20h
IMCR121	C24h
IMCR128	C40h
IMCR129	C44h
IMCR143	C7Ch
IMCR144	C80h
IMCR145	C84h
IMCR146	C88h
IMCR147	C8Ch
IMCR148	C90h
IMCR149	C94h
IMCR150	C98h
IMCR151	C9Ch
IMCR153	CA4h
IMCR154	CA8h
IMCR155	CACh
IMCR156	CB0h
IMCR157	CB4h
IMCR158	CB8h
IMCR159	CBCh
IMCR160	CC0h
IMCR161	CC4h
IMCR205	D74h
IMCR206	D78h
IMCR207	D7Ch
IMCR208	D80h
IMCR209	D84h
IMCR210	D88h
IMCR211	D8Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR212	D90h
IMCR224	DC0h
IMCR225	DC4h
IMCR233	DE4h
IMCR234	DE8h
IMCR235	DECh
IMCR236	DF0h
IMCR237	DF4h
IMCR238	DF8h
IMCR239	DFCh
IMCR240	E00h
IMCR241	E04h
IMCR242	E08h
IMCR243	E0Ch
IMCR244	E10h
IMCR245	E14h
IMCR246	E18h
IMCR247	E1Ch
IMCR248	E20h
IMCR273	E84h
IMCR274	E88h
IMCR278	E98h
IMCR279	E9Ch
IMCR280	EA0h
IMCR281	EA4h
IMCR283	EACh
IMCR284	EB0h
IMCR285	EB4h
IMCR286	EB8h
IMCR288	EC0h
IMCR289	EC4h
IMCR290	EC8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR291	ECCh
IMCR292	ED0h
IMCR293	ED4h
IMCR294	ED8h
IMCR296	EE0h
IMCR297	EE4h
IMCR298	EE8h
IMCR299	EECh
IMCR300	EF0h
IMCR301	EF4h
IMCR302	EF8h
IMCR304	F00h
IMCR305	F04h
IMCR306	F08h
IMCR307	F0Ch
IMCR308	F10h
IMCR309	F14h
IMCR310	F18h
IMCR312	F20h
IMCR313	F24h
IMCR314	F28h
IMCR316	F30h
IMCR318	F38h
IMCR322	F48h
IMCR323	F4Ch
IMCR324	F50h
IMCR325	F54h
IMCR326	F58h
IMCR327	F5Ch
IMCR328	F60h
IMCR329	F64h
IMCR330	F68h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR331	F6Ch
IMCR332	F70h
IMCR333	F74h
IMCR334	F78h
IMCR335	F7Ch
IMCR336	F80h
IMCR337	F84h
IMCR338	F88h
IMCR339	F8Ch
IMCR340	F90h
IMCR343	F9Ch
IMCR344	FA0h
IMCR345	FA4h
IMCR346	FA8h
IMCR347	FACh
IMCR348	FB0h
IMCR349	FB4h
IMCR350	FB8h
IMCR351	FBCh
IMCR352	FC0h
IMCR353	FC4h
IMCR354	FC8h
IMCR355	FCCh
IMCR356	FD0h
IMCR357	FD4h
IMCR358	FD8h
IMCR359	FDCh
IMCR360	FE0h
IMCR363	FECh
IMCR364	FF0h
IMCR365	FF4h
IMCR366	FF8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR367	FFCh
IMCR368	1000h
IMCR369	1004h
IMCR370	1008h
IMCR371	100Ch
IMCR372	1010h
IMCR373	1014h
IMCR374	1018h
IMCR375	101Ch
IMCR376	1020h
IMCR377	1024h
IMCR378	1028h
IMCR379	102Ch
IMCR380	1030h
IMCR383	103Ch
IMCR384	1040h
IMCR385	1044h
IMCR386	1048h
IMCR387	104Ch
IMCR388	1050h
IMCR389	1054h
IMCR390	1058h
IMCR391	105Ch
IMCR392	1060h
IMCR393	1064h
IMCR398	1078h
IMCR399	107Ch
IMCR400	1080h
IMCR401	1084h
IMCR402	1088h
IMCR403	108Ch
IMCR404	1090h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR405	1094h
IMCR406	1098h
IMCR407	109Ch
IMCR408	10A0h
IMCR409	10A4h
IMCR410	10A8h
IMCR411	10ACh
IMCR412	10B0h
IMCR413	10B4h
IMCR414	10B8h
IMCR415	10BCh
IMCR416	10C0h
IMCR417	10C4h
IMCR418	10C8h
IMCR419	10CCh
IMCR420	10D0h
IMCR421	10D4h
IMCR422	10D8h
IMCR423	10DCh
IMCR424	10E0h
IMCR425	10E4h
IMCR426	10E8h
IMCR427	10ECh
IMCR428	10F0h
IMCR429	10F4h
IMCR430	10F8h
IMCR431	10FCh
IMCR432	1100h
IMCR433	1104h
IMCR467	118Ch
IMCR468	1190h
IMCR469	1194h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
IMCR470	1198h
IMCR473	11A4h
IMCR474	11A8h
IMCR475	11ACh
IMCR478	11B8h
IMCR479	11BCh
IMCR480	11C0h
IMCR483	11CCh
IMCR484	11D0h
IMCR485	11D4h
IMCR488	11E0h
IMCR489	11E4h
IMCR490	11E8h
IMCR493	11F4h
IMCR494	11F8h
IMCR495	11FCh

Function

IMCR*n* registers select the source signal connected to the register's associated destination, which is an internal module port that is an input port or can be configured as an input.

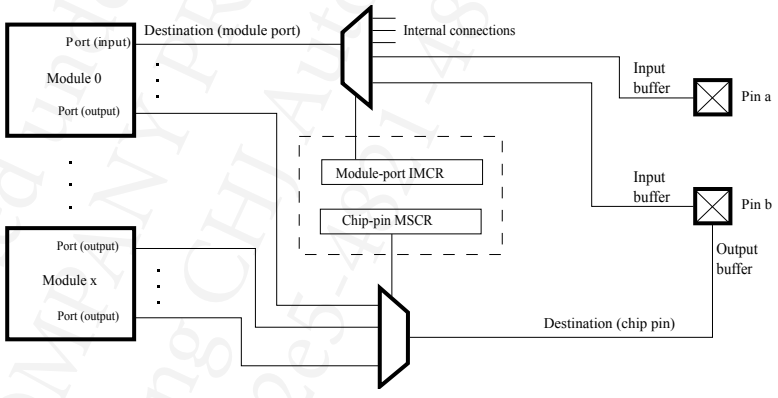


Figure 41. MSCR and IMCR port and pin connection

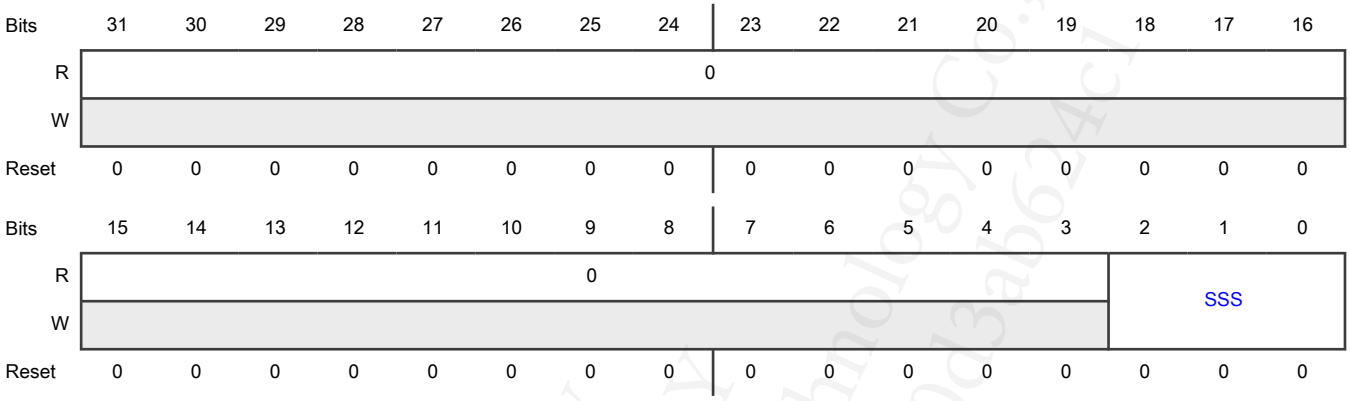
For IMCR assignments and field values, see the IOMUX file attached to this document.

Provided under NDA only
CONFIDENTIAL PROPRIETARY
Beijing CHIAUTO Automotive Technology Co., Ltd.
c8c0e25-4b1-4a8c-b251-390d3ab624c1

NOTE

- IMCR*n* registers support only 32-bit accesses. Byte and half-word write accesses are not supported.
- IMCR*n* registers must be configured only during application initialization and must not be modified during application runtime.
- Accessing a reserved IMCR*n* register generates a transfer error.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 SSS	Source Signal Select Selects which source signal is connected to the associated destination (chip pin).

16.4.14 SIUL2 GPIO Pad Data Output Register (GPDO115 - GPDO188)

Offset

Register	Offset
GPDO115	1370h
GPDO114	1371h
GPDO113	1372h
GPDO112	1373h
GPDO119	1374h
GPDO118	1375h
GPDO117	1376h
GPDO116	1377h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
GPDO122	1379h
GPDO121	137Ah
GPDO120	137Bh
GPDO147	1390h
GPDO146	1391h
GPDO145	1392h
GPDO144	1393h
GPDO151	1394h
GPDO150	1395h
GPDO149	1396h
GPDO148	1397h
GPDO155	1398h
GPDO154	1399h
GPDO153	139Ah
GPDO152	139Bh
GPDO159	139Ch
GPDO158	139Dh
GPDO157	139Eh
GPDO156	139Fh
GPDO163	13A0h
GPDO162	13A1h
GPDO161	13A2h
GPDO160	13A3h
GPDO167	13A4h
GPDO166	13A5h
GPDO165	13A6h
GPDO164	13A7h
GPDO171	13A8h
GPDO170	13A9h
GPDO169	13AAh
GPDO168	13ABh
GPDO175	13ACh

Table continues on the next page...

Table continued from the previous page...

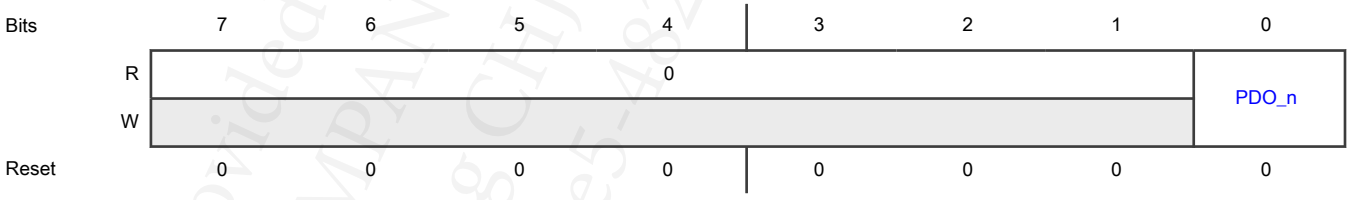
Register	Offset
GPDO174	13ADh
GPDO173	13AEh
GPDO172	13AFh
GPDO179	13B0h
GPDO178	13B1h
GPDO177	13B2h
GPDO176	13B3h
GPDO183	13B4h
GPDO182	13B5h
GPDO181	13B6h
GPDO180	13B7h
GPDO187	13B8h
GPDO186	13B9h
GPDO185	13BAh
GPDO184	13BBh
GPDO190	13BDh
GPDO189	13BEh
GPDO188	13BFh

Function

Each GPDO*n* register sets or clears a single GPIO pad with a byte access.

These registers support 8-, 16-, and 32-bit accesses.

Diagram



Fields

Field	Function
7-1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 PDO_n	Pad Data Out Stores the data to be driven out on the external GPIO pad controlled by this register when the pad is configured as an output. PDO_*n* represents PDO[*n*], where *n* is the instance of the register. 0b - Logic low value 1b - Logic high value

16.4.15 SIUL2 GPIO Pad Data Input Register (GPDI115 - GPDI188)

Offset

Register	Offset
GPDI115	1570h
GPDI114	1571h
GPDI113	1572h
GPDI112	1573h
GPDI119	1574h
GPDI118	1575h
GPDI117	1576h
GPDI116	1577h
GPDI122	1579h
GPDI121	157Ah
GPDI120	157Bh
GPDI147	1590h
GPDI146	1591h
GPDI145	1592h
GPDI144	1593h
GPDI151	1594h
GPDI150	1595h
GPDI149	1596h
GPDI148	1597h
GPDI155	1598h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
GPDI154	1599h
GPDI153	159Ah
GPDI152	159Bh
GPDI159	159Ch
GPDI158	159Dh
GPDI157	159Eh
GPDI156	159Fh
GPDI163	15A0h
GPDI162	15A1h
GPDI161	15A2h
GPDI160	15A3h
GPDI167	15A4h
GPDI166	15A5h
GPDI165	15A6h
GPDI164	15A7h
GPDI171	15A8h
GPDI170	15A9h
GPDI169	15AAh
GPDI168	15ABh
GPDI175	15ACh
GPDI174	15ADh
GPDI173	15AEh
GPDI172	15AFh
GPDI179	15B0h
GPDI178	15B1h
GPDI177	15B2h
GPDI176	15B3h
GPDI183	15B4h
GPDI182	15B5h
GPDI181	15B6h
GPDI180	15B7h
GPDI187	15B8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
GPDI186	15B9h
GPDI185	15BAh
GPDI184	15BBh
GPDI190	15BDh
GPDI189	15BEh
GPDI188	15BFh

Function

Each GPDI*n* register reads the GPIO pad data with a byte access.

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	7	6	5	4	3	2	1	0
R	0							PDI_n
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-1 —	Reserved
0 PDI_n	Pad Data In Stores the value of the external GPIO pad associated with this register. PDI_*n* represents PDI[*n*], where *n* is the instance of the register. 0b - Logic low 1b - Logic high

16.4.16 SIUL2 Parallel GPIO Pad Data Out Register (PGPDO7)

Offset

Register	Offset
PGPDO7	170Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c46265-482-48a8-b251-590d3ab624c1

Function

Each PGPDO*n* register sets or clears the respective pads of the chip.

PGPDO*n* registers can set the values of all output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* registers set the value on a specific pin with byte writes.

Each bit writes or reads the data register that stores the value to be driven on the pad in output mode. Access to this register location is coherent with access to the bit-wise GPDO*n*.

For a given PGPDO*x*[PPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

PGPDO*n* registers access the same physical resource as the PDO and MPGPDO address locations.

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDO 15	PPDO 14	PPDO 13	PPDO 12	PPDO 11	PPDO 10	PPDO 9	PPDO 8	PPDO 7	PPDO 6	PPDO 5	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15
14 PPDO14	Parallel Pad Data Out 14
13 PPDO13	Parallel Pad Data Out 13
12 PPDO12	Parallel Pad Data Out 12
11 PPDO11	Parallel Pad Data Out 11
10	Parallel Pad Data Out 10

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDO10	
9 PPDO9	Parallel Pad Data Out 9
8 PPDO8	Parallel Pad Data Out 8
7 PPDO7	Parallel Pad Data Out 7
6 PPDO6	Parallel Pad Data Out 6
5 PPDO5	Parallel Pad Data Out 5
4 —	Reserved Always write zero to this field.
3 —	Reserved Always write zero to this field.
2 —	Reserved Always write zero to this field.
1 —	Reserved Always write zero to this field.
0 —	Reserved Always write zero to this field.

16.4.17 SIUL2 Parallel GPIO Pad Data Out Register (PGPDO9)

Offset

Register	Offset
PGPDO9	1710h

Function

Each PGPDO*n* register sets or clears the respective pads of the chip.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co., Ltd.
8c462e5-4821-48a8-b251-390d3ab624c1

PGPDO*n* registers can set the values of all output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* registers set the value on a specific pin with byte writes.

Each bit writes or reads the data register that stores the value to be driven on the pad in output mode. Access to this register location is coherent with access to the bit-wise GPDO*n*.

For a given PGPDO*x*[PPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

PGPDO*n* registers access the same physical resource as the PDO and MPGPDO address locations.

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO	PPDO
W	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15
14 PPDO14	Parallel Pad Data Out 14
13 PPDO13	Parallel Pad Data Out 13
12 PPDO12	Parallel Pad Data Out 12
11 PPDO11	Parallel Pad Data Out 11
10 PPDO10	Parallel Pad Data Out 10
9	Parallel Pad Data Out 9

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDO9	
8 PPDO8	Parallel Pad Data Out 8
7 PPDO7	Parallel Pad Data Out 7
6 PPDO6	Parallel Pad Data Out 6
5 PPDO5	Parallel Pad Data Out 5
4 PPDO4	Parallel Pad Data Out 4
3 PPDO3	Parallel Pad Data Out 3
2 PPDO2	Parallel Pad Data Out 2
1 PPDO1	Parallel Pad Data Out 1
0 PPDO0	Parallel Pad Data Out 0

16.4.18 SIUL2 Parallel GPIO Pad Data Out Register (PGPDO11)

Offset

Register	Offset
PGPDO11	1714h

Function

Each PGPDO*n* register sets or clears the respective pads of the chip.

PGPDO*n* registers can set the values of all output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* registers set the value on a specific pin with byte writes.

Each bit writes or reads the data register that stores the value to be driven on the pad in output mode. Access to this register location is coherent with access to the bit-wise GPDO*n*.

For a given PGPDO*x*[PPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

PGPDO*n* registers access the same physical resource as the PDO and MPGPDO address locations.

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDO 15	PPDO 14	PPDO 13	PPDO 12	PPDO 11	PPDO 10	PPDO 9	PPDO 8	PPDO 7	PPDO 6	PPDO 5	PPDO 4	PPDO 3	PPDO 2	PPDO 1	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15
14 PPDO14	Parallel Pad Data Out 14
13 PPDO13	Parallel Pad Data Out 13
12 PPDO12	Parallel Pad Data Out 12
11 PPDO11	Parallel Pad Data Out 11
10 PPDO10	Parallel Pad Data Out 10
9 PPDO9	Parallel Pad Data Out 9
8 PPDO8	Parallel Pad Data Out 8

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 PPDO7	Parallel Pad Data Out 7
6 PPDO6	Parallel Pad Data Out 6
5 PPDO5	Parallel Pad Data Out 5
4 PPDO4	Parallel Pad Data Out 4
3 PPDO3	Parallel Pad Data Out 3
2 PPDO2	Parallel Pad Data Out 2
1 PPDO1	Parallel Pad Data Out 1
0 —	Reserved Always write zero to this field.

16.4.19 SIUL2 Parallel GPIO Pad Data Out Register (PGPDO10)

Offset

Register	Offset
PGPDO10	1716h

Function

Each PGPDO*n* register sets or clears the respective pads of the chip.

PGPDO*n* registers can set the values of all output pins assigned to a chip port with a single 16-bit register write, while the GPDO*n* registers set the value on a specific pin with byte writes.

Each bit writes or reads the data register that stores the value to be driven on the pad in output mode. Access to this register location is coherent with access to the bit-wise GPDO*n*.

For a given PGPDO*x*[PPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

PGPDO*x*[PPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDO0[PPDO15] = GPDO0[PDO_0]
- PGPDO2[PPDO15] = GPDO32[PDO_32]
- PGPDO31[PPDO0] = GPDO511[PDO_511]

PGPDO*n* registers access the same physical resource as the PDO and MPGPDO address locations.

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PPDO 15	PPDO 14	PPDO 13	PPDO 12	PPDO 11	PPDO 10	PPDO 9	PPDO 8	PPDO 7	PPDO 6	PPDO 5	PPDO 4	PPDO 3	PPDO 2	PPDO 1	PPDO 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDO15	Parallel Pad Data Out 15
14 PPDO14	Parallel Pad Data Out 14
13 PPDO13	Parallel Pad Data Out 13
12 PPDO12	Parallel Pad Data Out 12
11 PPDO11	Parallel Pad Data Out 11
10 PPDO10	Parallel Pad Data Out 10
9 PPDO9	Parallel Pad Data Out 9
8 PPDO8	Parallel Pad Data Out 8
7 PPDO7	Parallel Pad Data Out 7
6	Parallel Pad Data Out 6

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDO6	
5 PPDO5	Parallel Pad Data Out 5
4 PPDO4	Parallel Pad Data Out 4
3 PPDO3	Parallel Pad Data Out 3
2 PPDO2	Parallel Pad Data Out 2
1 PPDO1	Parallel Pad Data Out 1
0 PPDO0	Parallel Pad Data Out 0

16.4.20 SIUL2 Parallel GPIO Pad Data In Register (PGPDI7)

Offset

Register	Offset
PGPDI7	174Ch

Function

PGPDI*n* registers hold the synchronized input value from the pads.

PGPDI*n* registers can read the values of all input pins assigned to a chip port with a single 16-bit register read, while the GPDI*n* registers read the value on a specific pin with a byte read.

Each bit reads the current pad value. Access to this register location is coherent with access to the bit-wise GPDI*n*.

For a given PGPDI*x*[PPDI*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] bit:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

These registers support 8-, 16-, and 32-bit accesses.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	PPDI9	PPDI8	PPDI7	PPDI6	PPDI5	Reserved	Reserved	Reserved	Reserved	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data In 15
14 PPDI14	Parallel Pad Data In 14
13 PPDI13	Parallel Pad Data In 13
12 PPDI12	Parallel Pad Data In 12
11 PPDI11	Parallel Pad Data In 11
10 PPDI10	Parallel Pad Data In 10
9 PPDI9	Parallel Pad Data In 9
8 PPDI8	Parallel Pad Data In 8
7 PPDI7	Parallel Pad Data In 7
6 PPDI6	Parallel Pad Data In 6
5 PPDI5	Parallel Pad Data In 5
4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

16.4.21 SIUL2 Parallel GPIO Pad Data In Register (PGPDI9)

Offset

Register	Offset
PGPDI9	1750h

Function

PGPDI*n* registers hold the synchronized input value from the pads.

PGPDI*n* registers can read the values of all input pins assigned to a chip port with a single 16-bit register read, while the GPDI*n* registers read the value on a specific pin with a byte read.

Each bit reads the current pad value. Access to this register location is coherent with access to the bit-wise GPDI*n*.

For a given PGPDI*x*[PPDI*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] bit:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

These registers support 8-, 16-, and 32-bit accesses.

Preliminary Information
COMPANY CONFIDENTIAL
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4818-b251-390a3ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	PPDI9	PPDI8	PPDI7	PPDI6	PPDI5	PPDI4	PPDI3	PPDI2	PPDI1	PPDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data In 15
14 PPDI14	Parallel Pad Data In 14
13 PPDI13	Parallel Pad Data In 13
12 PPDI12	Parallel Pad Data In 12
11 PPDI11	Parallel Pad Data In 11
10 PPDI10	Parallel Pad Data In 10
9 PPDI9	Parallel Pad Data In 9
8 PPDI8	Parallel Pad Data In 8
7 PPDI7	Parallel Pad Data In 7
6 PPDI6	Parallel Pad Data In 6
5 PPDI5	Parallel Pad Data In 5
4	Parallel Pad Data In 4

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDI4	
3 PPDI3	Parallel Pad Data In 3
2 PPDI2	Parallel Pad Data In 2
1 PPDI1	Parallel Pad Data In 1
0 PPDI0	Parallel Pad Data In 0

16.4.22 SIUL2 Parallel GPIO Pad Data In Register (PGPDI11)

Offset

Register	Offset
PGPDI11	1754h

Function

PGPDI*n* registers hold the synchronized input value from the pads.

PGPDI*n* registers can read the values of all input pins assigned to a chip port with a single 16-bit register read, while the GPDI*n* registers read the value on a specific pin with a byte read.

Each bit reads the current pad value. Access to this register location is coherent with access to the bit-wise GPDI*n*.

For a given PGPDI*x*[PPDI*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] bit:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

These registers support 8-, 16-, and 32-bit accesses.

Preliminary Information
COMPANY CONFIDENTIAL
Beijing CHI Automotive Technology Co., Ltd.
c8c462e5-4821-4888-b251-390a3ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	PPDI9	PPDI8	PPDI7	PPDI6	PPDI5	PPDI4	PPDI3	PPDI2	PPDI1	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data In 15
14 PPDI14	Parallel Pad Data In 14
13 PPDI13	Parallel Pad Data In 13
12 PPDI12	Parallel Pad Data In 12
11 PPDI11	Parallel Pad Data In 11
10 PPDI10	Parallel Pad Data In 10
9 PPDI9	Parallel Pad Data In 9
8 PPDI8	Parallel Pad Data In 8
7 PPDI7	Parallel Pad Data In 7
6 PPDI6	Parallel Pad Data In 6
5 PPDI5	Parallel Pad Data In 5
4	Parallel Pad Data In 4

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDI4	
3 PPDI3	Parallel Pad Data In 3
2 PPDI2	Parallel Pad Data In 2
1 PPDI1	Parallel Pad Data In 1
0 —	Reserved

16.4.23 SIUL2 Parallel GPIO Pad Data In Register (PGPDI10)

Offset

Register	Offset
PGPDI10	1756h

Function

PGPDI*n* registers hold the synchronized input value from the pads.

PGPDI*n* registers can read the values of all input pins assigned to a chip port with a single 16-bit register read, while the GPDI*n* registers read the value on a specific pin with a byte read.

Each bit reads the current pad value. Access to this register location is coherent with access to the bit-wise GPDI*n*.

For a given PGPDI*x*[PPDI*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDI*n*[PDI_*n*] bit:

PGPDI*x*[PPDI*y*] = GPDI(*x* × 16) + (15 - *y*)[PDI_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- PGPDI0[PPDI15] = GPDI0[PDI_0]
- PGPDI2[PPDI15] = GPDI32[PDI_32]
- PGPDI31[PPDI0] = GPDI511[PDI_511]

These registers support 8-, 16-, and 32-bit accesses.

Preliminary Information
COMPANY CONFIDENTIAL
Beijing CHI Automotive Technology Co., Ltd.
c8c462e5-4821-4888-b251-390a3ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PPDI15	PPDI14	PPDI13	PPDI12	PPDI11	PPDI10	PPDI9	PPDI8	PPDI7	PPDI6	PPDI5	PPDI4	PPDI3	PPDI2	PPDI1	PPDI0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 PPDI15	Parallel Pad Data In 15
14 PPDI14	Parallel Pad Data In 14
13 PPDI13	Parallel Pad Data In 13
12 PPDI12	Parallel Pad Data In 12
11 PPDI11	Parallel Pad Data In 11
10 PPDI10	Parallel Pad Data In 10
9 PPDI9	Parallel Pad Data In 9
8 PPDI8	Parallel Pad Data In 8
7 PPDI7	Parallel Pad Data In 7
6 PPDI6	Parallel Pad Data In 6
5 PPDI5	Parallel Pad Data In 5
4	Parallel Pad Data In 4

Table continues on the next page...

Table continued from the previous page...

Field	Function
PPDI4	
3 PPDI3	Parallel Pad Data In 3
2 PPDI2	Parallel Pad Data In 2
1 PPDI1	Parallel Pad Data In 1
0 PPDI0	Parallel Pad Data In 0

16.4.24 SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO7)

Offset

Register	Offset
MPGPDO7	179Ch

Function

Each MPGPDO*n* register selectively modifies the pad values associated with PGPDO*n*.

NOTE

MPGPDO*n* registers must only be accessed with 32-bit writes. 8-bit or 16-bit writes will not modify any bits in the register and cause a transfer error. Read access will return 0.

NOTE

MPGPDO*n* registers support only 32-bit accesses. Byte and half-word accesses are not supported.

Accesses to each MPGPDO*n* register location is coherent with access to the bit-wise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]
- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Freescale NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e9-a021-4a8-b251-390a3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	MASK 9	MASK 8	MASK 7	MASK 6	MASK 5	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	MPPDO9	MPPDO8	MPPDO7	MPPDO6	MPPDO5	Reserved	Reserved	Reserved	Reserved	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MASK15	Mask Field 15 Masks MPPDO15 in the same MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the same MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29 MASK13	Mask Field 13 Masks MPPDO13 in the same MPGPDO*n* register instance. 0b - MPPDO13 is ignored 1b - MPPDO13 is written
28 MASK12	Mask Field 12 Masks MPPDO12 in the same MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written
27 MASK11	Mask Field 11 Masks MPPDO11 in the same MPGPDO*n* register instance. 0b - MPPDO11 is ignored 1b - MPPDO11 is written
26	Mask Field 10

Table continues on the next page...

Table continued from the previous page...

Field	Function
MASK10	Masks MPPDO10 in the same MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 MASK9	Mask Field 9 Masks MPPDO9 in the same MPGPDO*n* register instance. 0b - MPPDO9 is ignored 1b - MPPDO9 is written
24 MASK8	Mask Field 8 Masks MPPDO8 in the same MPGPDO*n* register instance. 0b - MPPDO8 is ignored 1b - MPPDO8 is written
23 MASK7	Mask Field 7 Masks MPPDO7 in the same MPGPDO*n* register instance. 0b - MPPDO7 is ignored 1b - MPPDO7 is written
22 MASK6	Mask Field 6 Masks MPPDO6 in the same MPGPDO*n* register instance. 0b - MPPDO6 is ignored 1b - MPPDO6 is written
21 MASK5	Mask Field 5 Masks MPPDO5 in the same MPGPDO*n* register instance. 0b - MPPDO5 is ignored 1b - MPPDO5 is written
20 —	Reserved Always write zero to this field.
19 —	Reserved Always write zero to this field.
18 —	Reserved Always write zero to this field.
17	Reserved Always write zero to this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
16 —	Reserved Always write zero to this field.
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes the data register that stores the value to be driven on the pad in output mode.
14 MPPDO14	Masked Parallel Pad Data Out 14 Writes the data register that stores the value to be driven on the pad in output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes the data register that stores the value to be driven on the pad in output mode.
12 MPPDO12	Masked Parallel Pad Data Out 12 Writes the data register that stores the value to be driven on the pad in output mode.
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes the data register that stores the value to be driven on the pad in output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes the data register that stores the value to be driven on the pad in output mode.
9 MPPDO9	Masked Parallel Pad Data Out 9 Writes the data register that stores the value to be driven on the pad in output mode.
8 MPPDO8	Masked Parallel Pad Data Out 8 Writes the data register that stores the value to be driven on the pad in output mode.
7 MPPDO7	Masked Parallel Pad Data Out 7 Writes the data register that stores the value to be driven on the pad in output mode.
6 MPPDO6	Masked Parallel Pad Data Out 6 Writes the data register that stores the value to be driven on the pad in output mode.
5 MPPDO5	Masked Parallel Pad Data Out 5 Writes the data register that stores the value to be driven on the pad in output mode.
4 —	Reserved Always write zero to this field.
3 —	Reserved Always write zero to this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 —	Reserved Always write zero to this field.
1 —	Reserved Always write zero to this field.
0 —	Reserved Always write zero to this field.

16.4.25 SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO9 - MPGPDO10)

Offset

Register	Offset
MPGPDO9	17A4h
MPGPDO10	17A8h

Function

Each MPGPDO*n* register selectively modifies the pad values associated with PGPDO*n*.

NOTE

MPGPDO*n* registers must only be accessed with 32-bit writes. 8-bit or 16-bit writes will not modify any bits in the register and cause a transfer error. Read access will return 0.

NOTE

MPGPDO*n* registers support only 32-bit accesses. Byte and half-word accesses are not supported.

Accesses to each MPGPDO*n* register location is coherent with access to the bit-wise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]
- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co., Ltd.
c8c462e5-4621-4ba8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	MASK 9	MASK 8	MASK 7	MASK 6	MASK 5	MASK 4	MASK 3	MASK 2	MASK 1	MASK 0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	MPPDO9	MPPDO8	MPPDO7	MPPDO6	MPPDO5	MPPDO4	MPPDO3	MPPDO2	MPPDO1	MPPDO0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MASK15	Mask Field 15 Masks MPPDO15 in the same MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the same MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29 MASK13	Mask Field 13 Masks MPPDO13 in the same MPGPDO*n* register instance. 0b - MPPDO13 is ignored 1b - MPPDO13 is written
28 MASK12	Mask Field 12 Masks MPPDO12 in the same MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written
27 MASK11	Mask Field 11 Masks MPPDO11 in the same MPGPDO*n* register instance. 0b - MPPDO11 is ignored 1b - MPPDO11 is written
26	Mask Field 10

Table continues on the next page...

Table continued from the previous page...

Field	Function
MASK10	Masks MPPDO10 in the same MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 MASK9	Mask Field 9 Masks MPPDO9 in the same MPGPDO*n* register instance. 0b - MPPDO9 is ignored 1b - MPPDO9 is written
24 MASK8	Mask Field 8 Masks MPPDO8 in the same MPGPDO*n* register instance. 0b - MPPDO8 is ignored 1b - MPPDO8 is written
23 MASK7	Mask Field 7 Masks MPPDO7 in the same MPGPDO*n* register instance. 0b - MPPDO7 is ignored 1b - MPPDO7 is written
22 MASK6	Mask Field 6 Masks MPPDO6 in the same MPGPDO*n* register instance. 0b - MPPDO6 is ignored 1b - MPPDO6 is written
21 MASK5	Mask Field 5 Masks MPPDO5 in the same MPGPDO*n* register instance. 0b - MPPDO5 is ignored 1b - MPPDO5 is written
20 MASK4	Mask Field 4 Masks MPPDO4 in the same MPGPDO*n* register instance. 0b - MPPDO4 is ignored 1b - MPPDO4 is written
19 MASK3	Mask Field 3 Masks MPPDO3 in the same MPGPDO*n* register instance. 0b - MPPDO3 is ignored 1b - MPPDO3 is written

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 MASK2	Mask Field 2 Masks MPPDO2 in the same MPGPDO*n* register instance. 0b - MPPDO2 is ignored 1b - MPPDO2 is written
17 MASK1	Mask Field 1 Masks MPPDO1 in the same MPGPDO*n* register instance. 0b - MPPDO1 is ignored 1b - MPPDO1 is written
16 MASK0	Mask Field 0 Masks MPPDO0 in the same MPGPDO*n* register instance. 0b - MPPDO0 is ignored 1b - MPPDO0 is written
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes the data register that stores the value to be driven on the pad in output mode.
14 MPPDO14	Masked Parallel Pad Data Out 14 Writes the data register that stores the value to be driven on the pad in output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes the data register that stores the value to be driven on the pad in output mode.
12 MPPDO12	Masked Parallel Pad Data Out 12 Writes the data register that stores the value to be driven on the pad in output mode.
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes the data register that stores the value to be driven on the pad in output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes the data register that stores the value to be driven on the pad in output mode.
9 MPPDO9	Masked Parallel Pad Data Out 9 Writes the data register that stores the value to be driven on the pad in output mode.
8 MPPDO8	Masked Parallel Pad Data Out 8 Writes the data register that stores the value to be driven on the pad in output mode.
7 MPPDO7	Masked Parallel Pad Data Out 7 Writes the data register that stores the value to be driven on the pad in output mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 MPPDO6	Masked Parallel Pad Data Out 6 Writes the data register that stores the value to be driven on the pad in output mode.
5 MPPDO5	Masked Parallel Pad Data Out 5 Writes the data register that stores the value to be driven on the pad in output mode.
4 MPPDO4	Masked Parallel Pad Data Out 4 Writes the data register that stores the value to be driven on the pad in output mode.
3 MPPDO3	Masked Parallel Pad Data Out 3 Writes the data register that stores the value to be driven on the pad in output mode.
2 MPPDO2	Masked Parallel Pad Data Out 2 Writes the data register that stores the value to be driven on the pad in output mode.
1 MPPDO1	Masked Parallel Pad Data Out 1 Writes the data register that stores the value to be driven on the pad in output mode.
0 MPPDO0	Masked Parallel Pad Data Out 0 Writes the data register that stores the value to be driven on the pad in output mode.

16.4.26 SIUL2 Masked Parallel GPIO Pad Data Out Register (MPGPDO11)

Offset

Register	Offset
MPGPDO11	17ACh

Function

Each MPGPDO*n* register selectively modifies the pad values associated with PGPDO*n*.

NOTE

MPGPDO*n* registers must only be accessed with 32-bit writes. 8-bit or 16-bit writes will not modify any bits in the register and cause a transfer error. Read access will return 0.

NOTE

MPGPDO*n* registers support only 32-bit accesses. Byte and half-word accesses are not supported.

Accesses to each MPGPDO*n* register location is coherent with access to the bit-wise GPDO*n*.

For a given MPGPDO*x*[MPPDO*y*] where *x* is the register instance index and *y* is the field index, the following equation shows the equivalent GPDO*n*[PDO_*n*] bit:

MPGPDO*x*[MPPDO*y*] = GPDO(*x* × 16) + (15 - *y*)[PDO_(*x* × 16) + (15 - *y*)]

Some examples of the mapping:

- MPGPDO0[MPPDO15] = GPDO0[PDO_0]
- MPGPDO2[MPPDO15] = GPDO32[PDO_32]
- MPGPDO31[MPPDO0] = GPDO511[PDO_511]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MASK 15	MASK 14	MASK 13	MASK 12	MASK 11	MASK 10	MASK 9	MASK 8	MASK 7	MASK 6	MASK 5	MASK 4	MASK 3	MASK 2	MASK 1	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	MPPDO15	MPPDO14	MPPDO13	MPPDO12	MPPDO11	MPPDO10	MPPDO9	MPPDO8	MPPDO7	MPPDO6	MPPDO5	MPPDO4	MPPDO3	MPPDO2	MPPDO1	Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MASK15	Mask Field 15 Masks MPPDO15 in the same MPGPDO*n* register instance. 0b - MPPDO15 is ignored 1b - MPPDO15 is written
30 MASK14	Mask Field 14 Masks MPPDO14 in the same MPGPDO*n* register instance. 0b - MPPDO14 is ignored 1b - MPPDO14 is written
29 MASK13	Mask Field 13 Masks MPPDO13 in the same MPGPDO*n* register instance. 0b - MPPDO13 is ignored 1b - MPPDO13 is written
28 MASK12	Mask Field 12 Masks MPPDO12 in the same MPGPDO*n* register instance. 0b - MPPDO12 is ignored 1b - MPPDO12 is written
27 MASK11	Mask Field 11 Masks MPPDO11 in the same MPGPDO*n* register instance.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MPPDO11 is ignored 1b - MPPDO11 is written
26 MASK10	Mask Field 10 Masks MPPDO10 in the same MPGPDO*n* register instance. 0b - MPPDO10 is ignored 1b - MPPDO10 is written
25 MASK9	Mask Field 9 Masks MPPDO9 in the same MPGPDO*n* register instance. 0b - MPPDO9 is ignored 1b - MPPDO9 is written
24 MASK8	Mask Field 8 Masks MPPDO8 in the same MPGPDO*n* register instance. 0b - MPPDO8 is ignored 1b - MPPDO8 is written
23 MASK7	Mask Field 7 Masks MPPDO7 in the same MPGPDO*n* register instance. 0b - MPPDO7 is ignored 1b - MPPDO7 is written
22 MASK6	Mask Field 6 Masks MPPDO6 in the same MPGPDO*n* register instance. 0b - MPPDO6 is ignored 1b - MPPDO6 is written
21 MASK5	Mask Field 5 Masks MPPDO5 in the same MPGPDO*n* register instance. 0b - MPPDO5 is ignored 1b - MPPDO5 is written
20 MASK4	Mask Field 4 Masks MPPDO4 in the same MPGPDO*n* register instance. 0b - MPPDO4 is ignored 1b - MPPDO4 is written
19	Mask Field 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
MASK3	Masks MPPDO3 in the same MPGPDO*n* register instance. 0b - MPPDO3 is ignored 1b - MPPDO3 is written
18 MASK2	Mask Field 2 Masks MPPDO2 in the same MPGPDO*n* register instance. 0b - MPPDO2 is ignored 1b - MPPDO2 is written
17 MASK1	Mask Field 1 Masks MPPDO1 in the same MPGPDO*n* register instance. 0b - MPPDO1 is ignored 1b - MPPDO1 is written
16 —	Reserved Always write zero to this field.
15 MPPDO15	Masked Parallel Pad Data Out 15 Writes the data register that stores the value to be driven on the pad in output mode.
14 MPPDO14	Masked Parallel Pad Data Out 14 Writes the data register that stores the value to be driven on the pad in output mode.
13 MPPDO13	Masked Parallel Pad Data Out 13 Writes the data register that stores the value to be driven on the pad in output mode.
12 MPPDO12	Masked Parallel Pad Data Out 12 Writes the data register that stores the value to be driven on the pad in output mode.
11 MPPDO11	Masked Parallel Pad Data Out 11 Writes the data register that stores the value to be driven on the pad in output mode.
10 MPPDO10	Masked Parallel Pad Data Out 10 Writes the data register that stores the value to be driven on the pad in output mode.
9 MPPDO9	Masked Parallel Pad Data Out 9 Writes the data register that stores the value to be driven on the pad in output mode.
8 MPPDO8	Masked Parallel Pad Data Out 8 Writes the data register that stores the value to be driven on the pad in output mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 MPPDO7	Masked Parallel Pad Data Out 7 Writes the data register that stores the value to be driven on the pad in output mode.
6 MPPDO6	Masked Parallel Pad Data Out 6 Writes the data register that stores the value to be driven on the pad in output mode.
5 MPPDO5	Masked Parallel Pad Data Out 5 Writes the data register that stores the value to be driven on the pad in output mode.
4 MPPDO4	Masked Parallel Pad Data Out 4 Writes the data register that stores the value to be driven on the pad in output mode.
3 MPPDO3	Masked Parallel Pad Data Out 3 Writes the data register that stores the value to be driven on the pad in output mode.
2 MPPDO2	Masked Parallel Pad Data Out 2 Writes the data register that stores the value to be driven on the pad in output mode.
1 MPPDO1	Masked Parallel Pad Data Out 1 Writes the data register that stores the value to be driven on the pad in output mode.
0 —	Reserved Always write zero to this field.

16.5 Functional description

16.5.1 Pad Control

SIUL2 can control the electrical characteristics of as many as 512 pads. It provides a consistent interface for all pads, both on a by-port and a by-bit basis.

The setting of each pad out of reset is fixed per chip but can be configured individually. This way you can select special pull settings or peripheral pad ownership.

You can configure each pad independently of all other pads on the chip or other pads grouped within a single port. Thus you can group different pad types together in ports and operate the pads individually. Grouping the various functions for each pad into a single register allows you to configure each pad with a single write to a register, which further allows you to duplicate software for similar pads with index changes.

16.5.2 General Purpose Input and Output pads (GPIO)

SIUL2 allows each pad to be configured as either of the following:

- General Purpose Input or Output pad (GPIO)
- A pad for one or more alternate functions (input or output) determined by the peripheral that uses the pad

GPIO pads can also be implemented without any alternate function.

SIUL2 can manage as many as 512 GPIO pads organized as ports that can be accessed for data reads and writes as 8-bit, 16-bit, or 32-bit.

All port accesses are identical, with each read or write performed only at a different location to access a different port width:

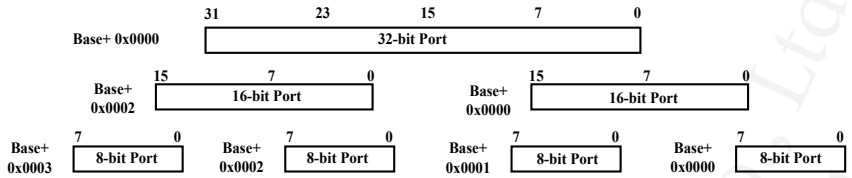


Figure 42. Data Port example arrangement showing configuration for different port width accesses

SIUL2 has separate data input and data output registers for all pads. You can thus directly read back an input or output value of a pad to validate what is actually present on the pad instead of confirming the value that was written to the data input registers.

- The data output registers support both read and write operations.
- The data input registers support read access only.

When a pad is configured to use one of its alternate functions, the data input values reflect the respective value of the pad. If a write operation is performed to the data output register for a pad configured as an alternate function (non-GPIO), this write will not be reflected by the pad value until re-configured to GPIO. All general purpose pads are implemented as bidirectional.

If bidirectional operation impacts performance or is not required for a pad function, you can limit the functionality of the pad to input only.

16.5.3 External interrupts/DMA requests (REQ Pins)

SIUL2 supports one to 32 external interrupts allocated to pads by the chip.

See the interrupt map file and the DMAMUX map file attached to this document for mapping of interrupt and DMA sources to interrupt vectors and DMA channels.

SIUL2 supports one to four interrupt vectors to the interrupt controller of the chip. Each interrupt vector can support as many as eight external interrupt sources from the chip pads.

All the external interrupt pads within a single group have equal priority. You are responsible for searching through the group of sources in the appropriate way for the application.

NOTE

Glitch filters appplied to external interrupts require a running internal oscillator clock. If such a clock is not available, enabling the glitch filter on an external interrupt will disable the interrupt.

16.5.3.1 External interrupt initialization

Follow these steps to enable external interrupts:

NOTE

If you do not follow these steps you may get a false interrupt flag during interrupt initialization.

1. Clear the DIRER0[EIRE*n*] bits to mask interrupts.
2. Set the appropriate IREER0[IREE*n*] bits and IFEER0[IFEE*n*] bits as needed to select the pin polarity.
3. Configure the appropriate bits in the MSCR register instances for the external interrupt pin(s):
    a. Clear the OBE and ODE bits to disable output.
    b. Set the IBE bit to enable the input buffer of the pin.
    c. If you are using the internal pullup or pulldown, configure the appropriate PUE and PUS fields.

4. Write the appropriate DIRSR0[DIRS*n*] bits to select a request between DMA or interrupt.
5. Select the desired glitch filter setup for the pins:
    a. Write the appropriate value to IFMCR*n*[MAXCNT] register for the respective external interrupt to the filter counter.
    b. Write the appropriate value to IFCPR[IFCP] to set the filter clock prescaler.
    c. Set the appropriate IFER0[IFE*n*] bits to enable the glitch filter for the external interrupt pins.
6. Write to the appropriate DISR0[EIF*n*] bits to clear any flags.
7. Set the appropriate DIRER0[EIRE*n*] bits to enable the interrupt pins.

16.5.3.2 External interrupt management

You can enable or disable each interrupt independently using a single rolled up register, DIRER0.

You can configure a pad defined as an external interrupt to recognize interrupts with an active rising edge, an active falling edge, or both, using the IREER and IFEER registers.

NOTE

You must not disable both edge events of a given interrupt.

Each external interrupt has an individual flag, held in the DISR0 register. DISR0 is a clear-by-write-1 register, which prevents inadvertent overwriting of other flags in the same register.

This figure provides an overview of the external interrupt implementation:

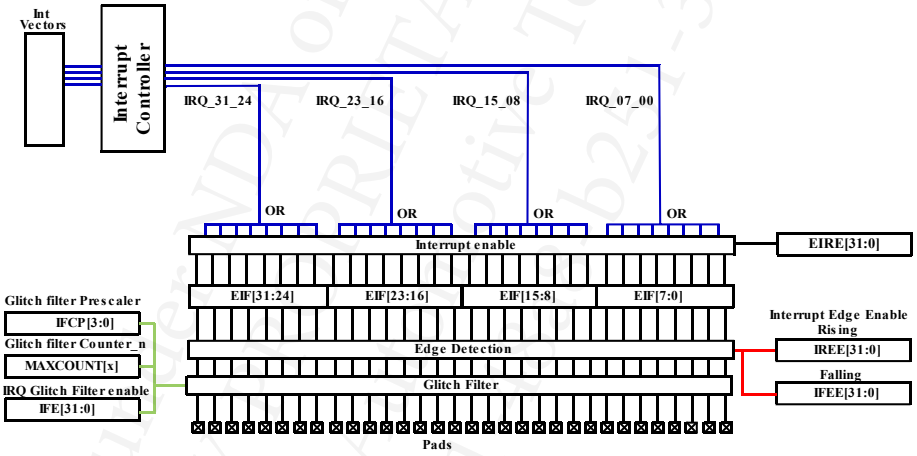


Figure 43. External interrupt pad diagram

16.5.3.3 External interrupt request

The REQ input pins on the chip are sources for interrupt or DMA requests. The chip provides four possible interrupt vectors for the SIUL2. The 32 interrupt request sources map to vectors and channels as follows:

Table 61. Interrupt source mapping to SIUL2 interrupt request output for 32 interrupt sources

Vector/Channel #	Interrupt vector source
0	REQ[07] | REQ[06] | REQ[05] | REQ[04] | REQ[03] | REQ[02] | REQ[01] | REQ[00]
1	REQ[15] | REQ[14] | REQ[13] | REQ[12] | REQ[11] | REQ[10] | REQ[09] | REQ[08]

Table continues on the next page...

Table 61. Interrupt source mapping to SIUL2 interrupt request output for 32 interrupt sources (continued)

Vector/Channel #	Interrupt vector source
2	REQ[23] | REQ[22] | REQ[21] | REQ[20] | REQ[19] | REQ[18] | REQ[17] | REQ[16]
3	REQ[31] | REQ[30] | REQ[29] | REQ[28] | REQ[27] | REQ[26] | REQ[25] | REQ[24]

16.5.3.4 DMA requests

The REQ pins on the chip map to independent DMA request channels in the DMA controller.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 17
Semaphores2 (SEMA42)

17.1 Introduction

SEMA42 is a memory-mapped module that provides robust hardware support needed in multi-core systems for implementing semaphores and provides a simple mechanism to achieve "lock and unlock" operations via a single - write access. The hardware semaphore module provides hardware-enforced gates as well as other useful system functions related to the gating mechanisms.

17.1.1 Multi-core programming: software gates

Multi-processor systems require a function that can be used to safely and easily provide a locking mechanism for system software to control access to shared data structures, shared hardware resources, and so on. The software uses the gating mechanisms to serialize (and synchronize) accesses to shared data and/or resources to prevent race conditions and preserve memory coherency between different processes and domains.

Consider the following description of a typical use-case: Domain X enters a section of code where shared data values are to be updated. The domain must first acquire a semaphore. Think of this as the locking (or closing) of a software gate. After the gate locks, a properly-architected software system does not allow other processes (or domains) to execute the same code segment or modify the shared data structure protected by the gate. In other words, the system locks out other processes/domains. Many software implementations include a spin-wait loop within the lock function until the gate locks. After domain X obtains the lock, domain X continues execution and updates the data values protected by the particular lock. After domain X completes the updates, it unlocks (or opens) the software gate, allowing other processes/domains access to the updated data values.

A correctly-implemented system solution must follow these important rules:

- A gate variable must protect all writes to shared data values or shared hardware resources.
- After a domain locks a gate, the system must block other processes/domains from accessing the shared data or resources. Software conventions enforce this.
- The domain that locks a particular gate is the only domain that can open (unlock) that gate.

Information in the hardware gate identifying the locking domain can be extremely useful for system-level debugging.

17.1.2 Features

SEMA42 implements hardware-enforced semaphores as an IPS-mapped slave peripheral device. The feature set includes:

- Support for 16 hardware-enforced gates in a multi-domain configuration that supports up to 15 domains. In this chapter, dm*X* represents domain *X*.
  - Gates appear as a 16-entry byte-size array with read and write accesses.
    - Domains lock gates by writing "domainID_number+1" to the appropriate gate and must read back the gate value to verify that the lock operation succeeded.
    - After the gate locks, the locking domain unlocks the gate by writing zeroes.
  - Each hardware gate appears as a 16-state, 4-bit state machine.
    - 16-state implementation

      If gate = 0h, then state = unlocked

      if gate = 1h, then state = locked by domain (master) 0

      if gate = 2h, then state = locked by domain (master) 1

      …

      if gate = Fh, then state = locked by domain (master) 14

- SEMA42 uses the logical domain number and the specified data patterns as reference attributes to validate all write operation.
- After a gate locks, the locking domain must unlock the gate by writing zeroes.

— Support for secure reset mechanisms to clear the contents of individual gates, as well as a clear-all capability

- Memory-mapped slave peripheral that offers programming-model accesses

The following figure shows a simplified SEMA42 block diagram. In the diagram, the register blocks named gate0, gate1,…, gate (n-2) and gate (n-1) include the finite state machines implementing the semaphore gates.

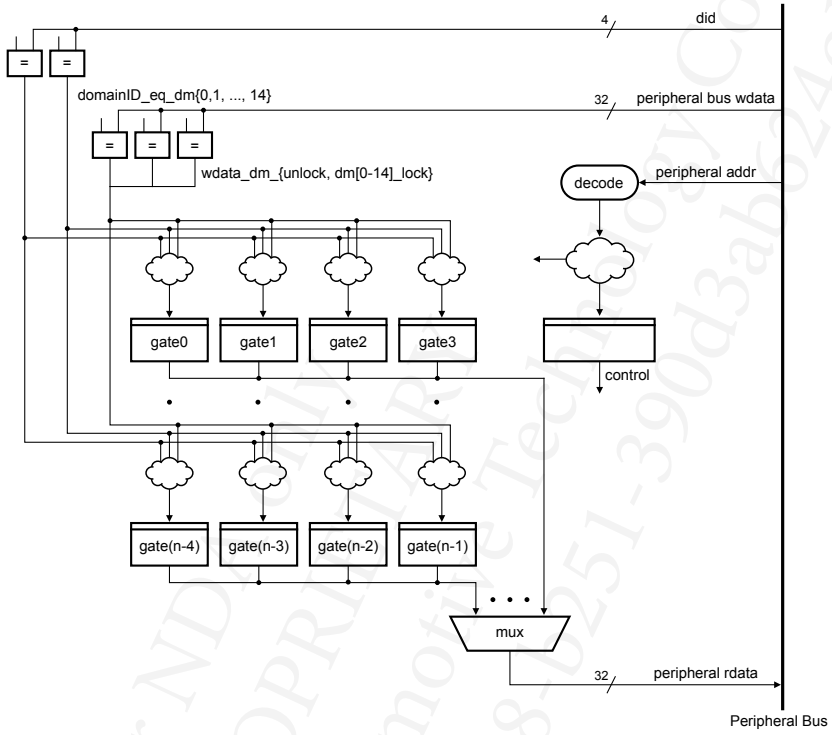


Figure 44. SEMA42 block diagram

17.2 Memory map/register definition

You can access these registers only in Supervisor mode. User accesses terminate with an error.

17.2.1 SEMA42 register descriptions

17.2.1.1 SEMA42 memory map

SEMA42 base address: 4029_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h - Fh	Gate Register (GATE0 - GATE15)	8	RW	00h
42h	Reset Gate Read (RSTGT_R)	16	RO	0000h
42h	Reset Gate Write (RSTGT_W)	16	WO	See description

17.2.1.2 Gate Register (GATE0 - GATE15)

Offset

Register	Offset
GATE3	0h
GATE2	1h
GATE1	2h
GATE0	3h
GATE7	4h
GATE6	5h
GATE5	6h
GATE4	7h
GATE11	8h
GATE10	9h
GATE9	Ah
GATE8	Bh
GATE15	Ch
GATE14	Dh
GATE13	Eh
GATE12	Fh

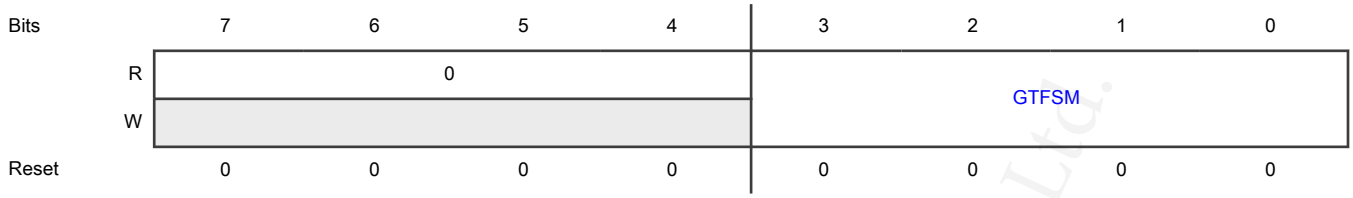
Function

SEMA42 implements each semaphore gate in a 4-bit finite state machine, right-justified in a byte data structure. The hardware uses the logical domain-identifier number in conjunction with the data patterns to validate all attempted write operations. Only domain masters can modify the gate registers. After a gate locks, only the locking domain must open (unlock) the gate.

You can read multiple gate values in a single access. However, you can update only a single gate at a time, via a write operation. If you attempt to write a byte-wide value that is neither the unlock value (00h) nor the appropriate lock value (domainID_number + 1), SEMA42 considers this as "no operation" and does not change any gate state. Attempts to write multiple gates in a single-aligned access with a size larger than 8 bits (byte) generate an error termination and do not allow any gate state changes.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4e1c-4a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
7-4 —	Reserved
3-0 GTFSM	Gate finite state machine The state of the gate reflects the last domain that locked it. This can be useful during system debug. The hardware gate has a 16-state implementation, defined as: 0000b - The gate is unlocked (free). 0001b - Domain 0 locked the gate. 0010b - Domain 1 locked the gate. 0011b - Domain 2 locked the gate. 0100b - Domain 3 locked the gate. 0101b - Domain 4 locked the gate. 0110b - Domain 5 locked the gate. 0111b - Domain 6 locked the gate. 1000b - Domain 7 locked the gate. 1001b - Domain 8 locked the gate. 1010b - Domain 9 locked the gate. 1011b - Domain 10 locked the gate. 1100b - Domain 11 locked the gate. 1101b - Domain 12 locked the gate. 1110b - Domain 13 locked the gate. 1111b - Domain 14 locked the gate.

17.2.1.3 Reset Gate Read (RSTGT_R)

Offset

Register	Offset
RSTGT_R	42h

Provided under NDA to COMPANY PROPRIETARY Beijing CarRobotics Automotive Technology Co., Ltd. c8c462e5-4821-48a8-b251-390d3ab624c1

Function

This section and Reset Gate Write (RSTGT_W) describe the same register. This section describes the general operation of the register and also shows how the register fields appear when you read the register. Reset Gate Write (RSTGT_W) shows how the register fields appear when you write to the register.

The intent of the hardware gate implementation is to specify a protocol where the locking domain must unlock the gate. However, some systems may require a reset function to re-initialize the state of any gate(s) without requiring a system-level reset. To support this special gate reset requirement, SEMA42 implements a secure reset mechanism that allows you to initialize a hardware gate (or all the gates) by following a specific dual-write access pattern. The secure-gate reset:

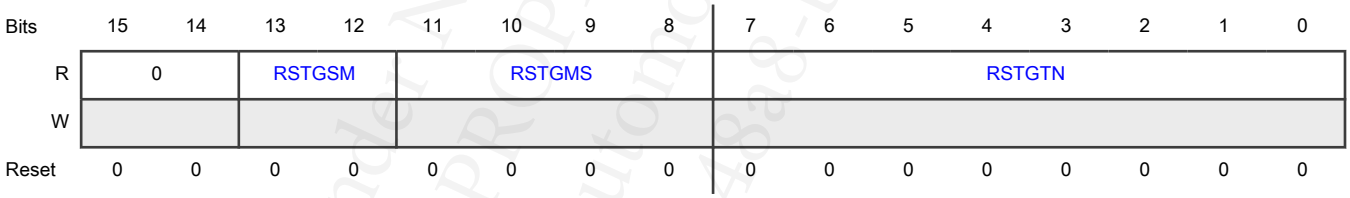
- Uses a technique similar to that required for the servicing of a software watchdog timer
- Requires two consecutive writes with pre-defined data patterns from the same domain

You must do this to force the clearing of the specified gate(s). The required access pattern as follows:

1. A domain performs a 16-bit write to the RSTGT memory location. The most significant byte (RSTGT_W[RSTGDP]) must be E2h. The value of least significant byte is irrelevant for this reference and can be anything.
2. The same domain then performs a second 16-bit write to the RSTGT location. For this write, the upper byte (RSTGT_W[RSTGDP]) is the logical complement of the first data pattern (1Dh) and the lower byte (RSTGT_W[RSTGTN]) specifies the gate(s) to be reset. This gate field can specify a single gate or all gates to be cleared. If the same domain writes incorrect data on the second access or another domain performs the second write access, SEMA42 aborts the special gate reset sequence and does not assert an error signal.
3. Reads of the RSTGT location return information on the 2-bit state machine (RSTGT_R[RSTGSM]) that implements these functions:
   - The domain performing the reset (RSTGT_R[RSTGMS])
   - The last-cleared gate number(s) (RSTGT_R[RSTGTN])

   Reads of the RSTGT register do not affect the secure-reset finite state machine in any manner.

Diagram



Fields

Field	Function
15-14 ROZ	ROZ This field always returns the value 0 when you read it.
13-12 RSTGSM	Reset gate finite state machine Reads of the RSTGT register return the encoded state machine value. RSTGSM = 10b is valid for only a single machine cycle, so a read can never return this value. SEMA42 maintains the reset state machine in a 2-bit, 3-state implementation, defined as follows: 00b - Idle, waiting for the first data pattern write.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01b - Waiting for the second data pattern write 10b - The 2-write sequence has completed. Generate the specified gate reset(s). After the reset is performed, this machine returns to the idle (waiting for first data pattern write) state. 11b - This state encoding is never used and therefore reserved.
11-8 RSTGMS	Reset gate domain This field records the logical number of the domain performing the gate reset function. The logical number is the domain number. This domain number is determined by the XRDC's Master Domain Assignment Controller (XRDC_MDAC). To succeed, this function requires that the same domain initiate the two consecutive writes to this register. SEMA42 updates the field each time a write to this register occurs.
7-0 RSTGTN	Reset gate number This field specifies the specific hardware gate to be reset. The second write updates this field. If RSTGTN < 64, SEMA42 resets the single gate defined by RSTGTN. Otherwise, SEMA42 resets all the gates.

17.2.1.4 Reset Gate Write (RSTGT_W)

Offset

Register	Offset
RSTGT_W	42h

Function

This section describes how the RSTGT fields appear when the register is written. See Reset Gate Read (RSTGT_R) for the description of the register's overall operation and how the register fields appear when you read the register.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	RSTGDP								RSTGTN							
Reset [1]	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

1. Reset value is not applicable for writes.

Fields

Field	Function
15-8 RSTGDP	Reset gate data pattern You access this field with the specified data patterns on the two consecutive writes to enable the gate-reset mechanism. For the first write, RSTGDP must be E2h. For the second write, RSTGDP must be 1Dh.
7-0 RSTGTN	Reset gate number This field specifies the specific hardware gate to be reset. The second write updates this field. If RSTGTN < 64, SEMA42 resets the single gate defined by RSTGTN. Otherwise, SEMA42 resets all the gates.

17.3 Functional description

This section describes more about the SEMA42 functional operation and specific details of the state machines of the GATE*n* registers.

As described previously, each of the GATE*n* registers implements a 4-bit, 16-state machine. The following figure shows a simplified diagram of the state transitions for each gate.

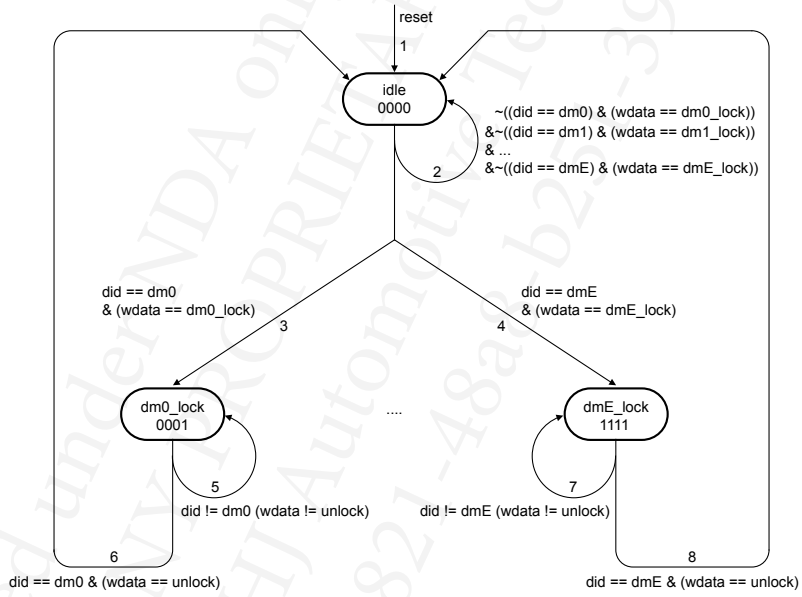


Figure 45. GATE*n* state machine

In the figure above, "did" is the domain identifier and "dm" is the domain number. The domain number identifies core domain *X* (dm*X*). Thus, for example, "dmE" represents domain 14 (E in hexadecimal). The platform passes the domain number to SEMA42.

The following table defines the GATE*n* state transitions.

Table 62. GATE*n* state transitions

Current state	Next state	Transition	Description
–	idle	1	Any reset, whether a system reset or a software-initiated gate reset, unconditionally forces the gate into the idle state.
idle	idle	2	Unless a write of the appropriate lock value from the corresponding domain occurs, the gate remains in the idle state.
idle	dm0_lock	3	When domain 0h initiates a write of the dm0_lock data value, the gate transitions into the dm0_lock state.
idle	dmE_lock	4	When domain Eh initiates a write of the dmE_lock value, the gate transitions into the dmE_lock state.
dm0_lock	dm0_lock	5	When in this state, the gate remains here if any attempted write is not from domain 0h with the unlock data value.
dm0_lock	idle	6	The gate returns to the idle (unlocked) state after a write from domain 0h with the unlock data value occurs.
dmE_lock	dmE_lock	7	When in this state, the gate remains here if any attempted write is not from domain Eh with the unlock data value.
dmE_lock	idle	8	The gate returns to the idle (unlocked) state after a write from domain Eh with the unlock data value occurs.

SEMA42 uses these gate data values:

- The lock data value is (domain number) + 1.
- The unlock data value is 00h.

17.4 Glossary

IPS Protocol used for peripheral accesses to the programming model

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b253-3b0a8a624c

Chapter 18
System Resource Controller (SRC)

18.1 Introduction

SRC contains miscellaneous control and status registers.

18.2 SRC register descriptions

18.2.1 SRC memory map

SRC base address: 4007_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Software-Triggered Faults (SW_NCF)	32	RW	0000_0000h
4h	GMAC Control (GMAC_0_CTRL_STS)	32	RW	0000_0000h
Ch	HSE Cortex-M7 Status (HSE_CM7_STATUS_REG)	32	RO	0000_0000h
28h	CMU Status 1 (CMU_STATUS_REG1)	32	W1C	0000_0000h
2Ch	CMUs Status 2 (CMU_STATUS_REG2)	32	W1C	0000_0000h
30h	FCCU EOUT Override Clear (FCCU_EOUT_OVERRIDE_CLEAR_REG)	32	RW	0000_0000h
54h	GPR21 (GPR21)	32	RW	0000_0000h
CCh	Debug Control (DEBUG_CONTROL)	32	RW	0000_0000h
F0h	Timestamp Control (TIMESTAMP_CONTROL_REGISTER)	32	RW	0000_0000h
F4h	FlexRay OS Tick Input Select (FLEXRAY_OS_TICK_INPUT_SELECT_REG)	32	RW	0000_0000h

18.2.2 Software-Triggered Faults (SW_NCF)

Offset

Register	Offset
SW_NCF	0h

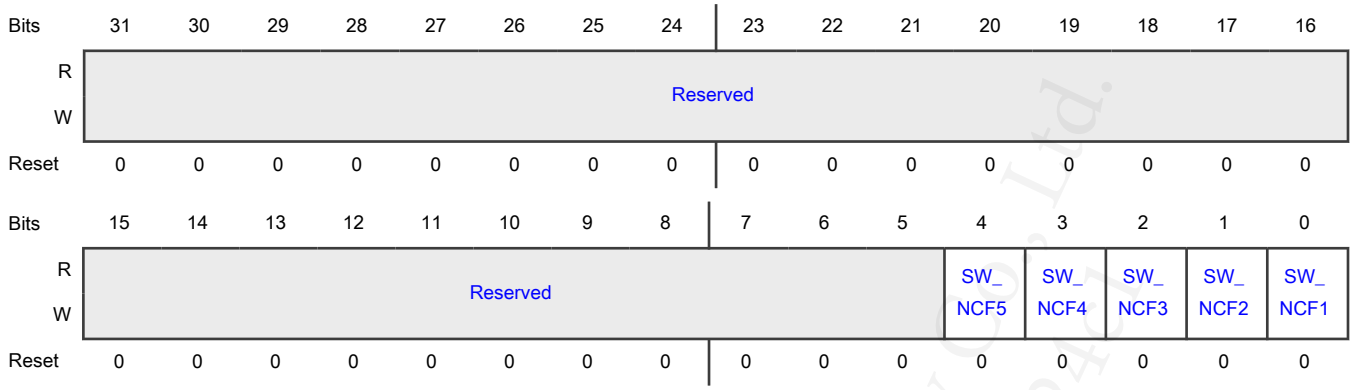
Function

Allow you to trigger faults to FCCU.

Reset phase: Functional

provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462c5-4821-4ae8-b21-3900d3a624c1

Diagram



Fields

Field	Function
31-5 —	Reserved
4 SW_NCF5	Software-Triggered Fault 5 Controls whether FCCU software fault 5 is asserted or not. 0b - Not asserted 1b - Asserted
3 SW_NCF4	Software-Triggered Fault 4 Controls whether FCCU software fault 4 is asserted or not. 0b - Not asserted 1b - Asserted
2 SW_NCF3	Software-Triggered Fault 3 Controls whether FCCU software fault 3 is asserted or not. 0b - Not asserted 1b - Asserted
1 SW_NCF2	Software-Triggered Fault 2 Controls whether FCCU software fault 2 is asserted or not. 0b - Not asserted 1b - Asserted
0 SW_NCF1	Software-Triggered Fault 1 Controls whether FCCU software fault 1 is asserted or not. 0b - Not asserted 1b - Asserted

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c0

18.2.3 GMAC Control (GMAC_0_CTRL_STS)

Offset

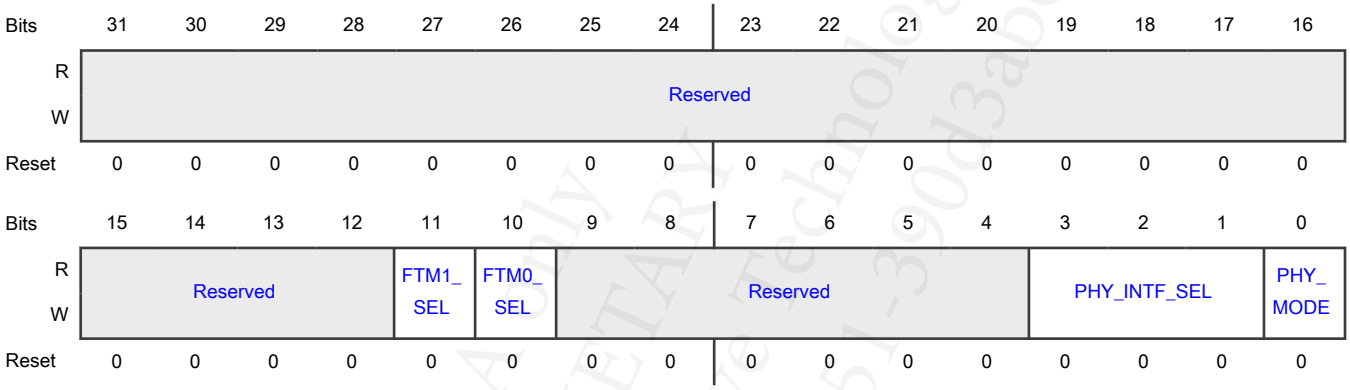
Register	Offset
GMAC_0_CTRL_STS	4h

Function

Controls GMAC configuration options.

Reset phase: Functional

Diagram



Fields

Field	Function
31-12 —	Reserved
11 FTM1_SEL	FTM_1 Select Controls whether the FTM_1 trigger connects to GMAC_0. 0b - GMAC_0 1b - Reserved
10 FTM0_SEL	FTM_0 Select Controls whether the FTM_0 trigger connects to GMAC_0. 0b - GMAC_0 1b - Reserved
9-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-1 PHY_INTF_SEL	PHY Interface Select Selects the PHY interface. 000b - MII 001b - RGMII 100b - RMII All other values are reserved.
0 PHY_MODE	PHY Mode Selects the PHY mode. 0b - Other PHY modes (for example: RGMII, RMII, MII) 1b - SGMII mode

18.2.4 HSE Cortex-M7 Status (HSE_CM7_STATUS_REG)

Offset

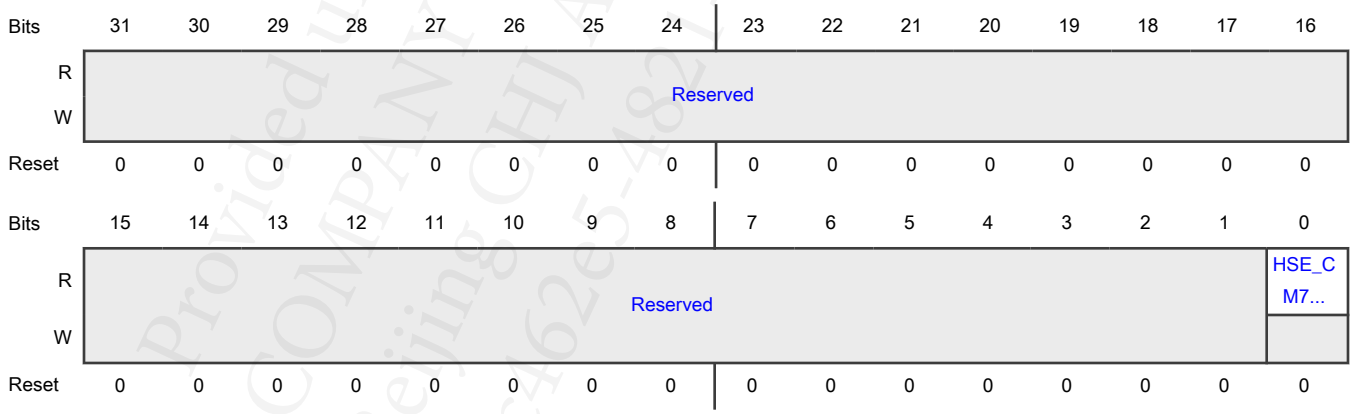
Register	Offset
HSE_CM7_STATUS_REG	Ch

Function

Provides status information of the HSE Cortex-M7 core.

Reset phase: Functional

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482a-48a8-b251-390d3ab624c1

Fields

Field	Function
31-1 —	Reserved
0 HSE_CM7_SLEEP_STAT	HSE Cortex-M7 Core Sleep Status Indicates the sleep state of the HSE Cortex-M7 core. 0b - Not in Sleep mode 1b - In Sleep mode

18.2.5 CMU Status 1 (CMU_STATUS_REG1)

Offset

Register	Offset
CMU_STATUS_REG1	28h

Function

Shows the status of these CMU events:

- If a monitored clock frequency is higher than high frequency reference (FHH)
- If a monitored clock frequency is lower than low frequency reference (FLL)

Reset phase: Destructive

NOTE

Immediately after the functional reset, do not read the content of the CMU_STATUS_REG1 for diagnosing the clock monitoring errors. Before entering the normal operation, you must clear the register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	Reserved		FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C			W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...	FLL_FHH...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing Chipsea Automotive Technology Co., Ltd.
c8e1e65-5f31-488-b251-390d3ab624c1

Fields

Field	Function
31 FLL_FHH_STAT_31	CMU_21 FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
30 FLL_FHH_STAT_30	CMU_21 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
29 FLL_FHH_STAT_29	cm7_cluster_2 CMU FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
28 FLL_FHH_STAT_28	cm7_cluster_2 CMU FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
27 FLL_FHH_STAT_27	cm7_cluster_1 CMU FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
26 FLL_FHH_STAT_26	cm7_cluster_1 CMU FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
25 FLL_FHH_STAT_25	cm7_cluster_0 CMU FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
24 FLL_FHH_STAT_24	cm7_cluster_0 CMU FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
23 FLL_FHH_STAT_23	CMU_20 FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
22 FLL_FHH_STAT_22	CMU_20 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
21-20 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 FLL_FHH_STAT_19	CMU_18 FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
18 FLL_FHH_STAT_18	CMU_18 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
17 FLL_FHH_STAT_17	CMU_17 FHH Interrupt Event Status 0b - Event has not occurred. 1b - Event has occurred.
16 FLL_FHH_STAT_16	CMU_17 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
15 FLL_FHH_STAT_15	CMU_16 FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
14 FLL_FHH_STAT_14	CMU_16 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
13 FLL_FHH_STAT_13	CMU_15 FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
12 FLL_FHH_STAT_12	CMU_15 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
11 FLL_FHH_STAT_11	CMU_14 FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
10 FLL_FHH_STAT_10	CMU_14 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
9	CMU_13 FHH Interrupt Event Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLL_FHH_STAT_9	0b - Event did not occur 1b - Event occurred
8 FLL_FHH_STAT_8	CMU_13 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
7 FLL_FHH_STAT_7	CMU_12 FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
6 FLL_FHH_STAT_6	CMU_12 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
5 FLL_FHH_STAT_5	CMU_11 FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
4 FLL_FHH_STAT_4	CMU_11 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
3 FLL_FHH_STAT_3	CMU_10 FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
2 FLL_FHH_STAT_2	CMU_10 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
1 FLL_FHH_STAT_1	CMU_7 FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
0 FLL_FHH_STAT_0	CMU_7 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESWIN Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

18.2.6 CMUs Status 2 (CMU_STATUS_REG2)

Offset

Register	Offset
CMU_STATUS_REG2	2Ch

Function

Shows the status of these CMU events:

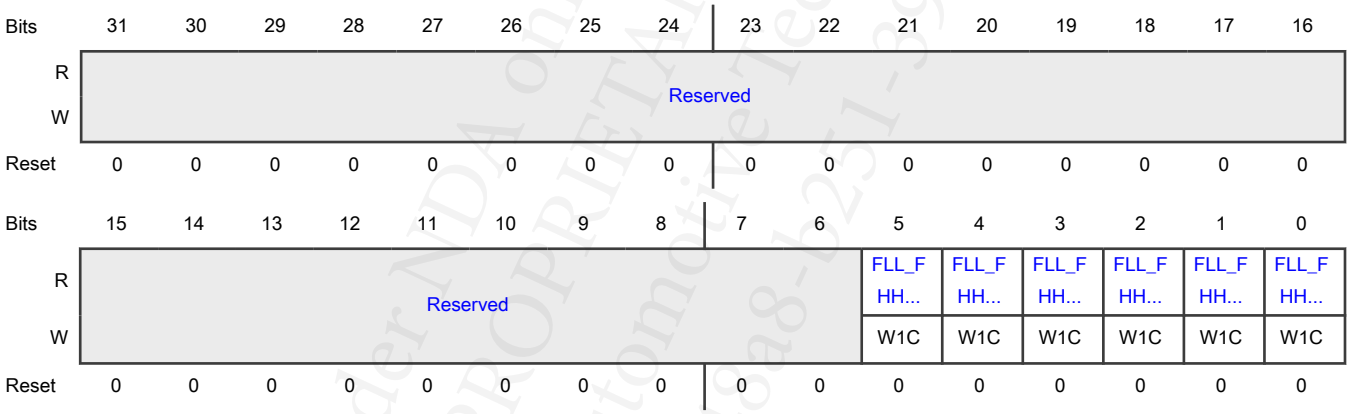
- If a monitored clock frequency is higher than high frequency reference (FHH)
- If a monitored clock frequency is lower than low frequency reference (FLL)

Reset phase: Destructive

NOTE

Immediately after the functional reset, do not read the content of the CMU_STATUS_REG2 for diagnosing the clock monitoring errors. Before entering the normal operation, you must clear the register.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 FLL_FHH_STAT_5	a53_cluster1 FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
4 FLL_FHH_STAT_4	a53_cluster1 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 FLL_FHH_STAT_3	a53_cluster0 FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
2 FLL_FHH_STAT_2	a53_cluster0 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred
1 FLL_FHH_STAT_1	CMU_22 FHH Interrupt Event Status 0b - Event did not occur 1b - Event occurred
0 FLL_FHH_STAT_0	CMU_22 FLL Interrupt Event Status 0b - Event did not occur 1b - Event occurred

18.2.7 FCCU EOUT Override Clear (FCCU_EOUT_OVERRIDE_CLEAR_REG)

Offset

Register	Offset
FCCU_EOUT_OVERRIDE_CLEAR_REG	30h

Function

Clears the override on the FCCU EOUT after a reset occurs due to the completion of the self-test/LBIST on the main reset domain.

A destructive reset resets this register.

Reset phase: Destructive

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved														EOUT_OV...	EOUT_OV...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-2 —	Reserved
1 EOUT_OVERRIDE_DISABLE_DURING_SELFTEST	EOUT Override Disable Controls whether to move FCCU EOUT into Fault state during the main domain self-test. 0b - Move 1b - Do not move
0 EOUT_OVERRIDE_CLEAR	EOUT Override Clear Allow you to clear the EOUT override control. Write 1 to this field after: • The FCCU EOUT configuration completes after the self-test exit reset sequence. • You configure FCCU EOUT. After you write 1 to this field, write 0 to it to enable another clearing operation in the future. 0b - Do not clear 1b - Clear

18.2.8 GPR21 (GPR21)

Offset

Register	Offset
GPR21	54h

Function

Controls parity mode for PCIe.

The individual bits within the PCIE_PARITY_MODE_data field select parity type (odd or even) in a logic that generates an error after comparing:

- Address bits that the PCIe controller generates
- The associated parity bits

See Table 63 for details.

Reset phase: Functional

Table 63. PCIE_PARITY_MODE_data field details

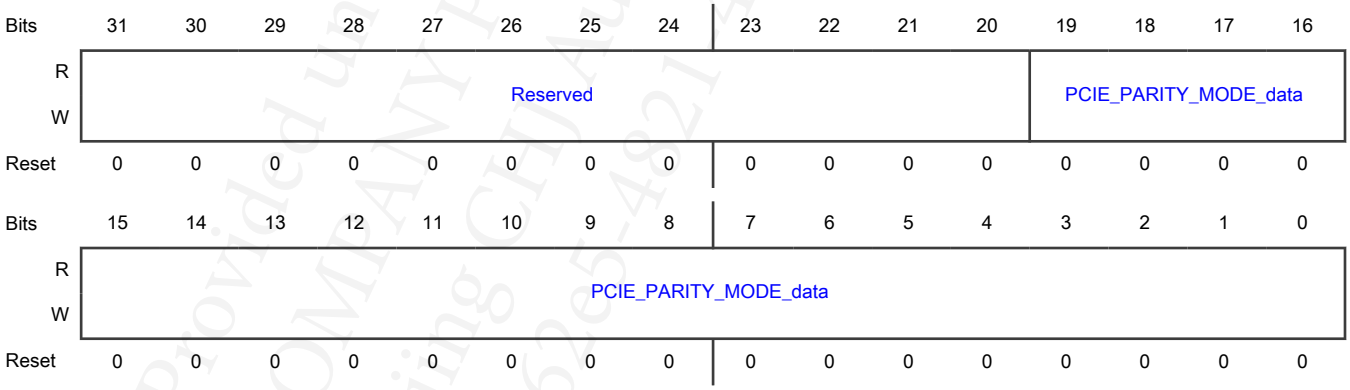
Bit position in PCIE_PARITY_MODE_data	PARITY_MODE bus
0	AXI 0 master read address

Table continues on the next page...

Table 63. PCIE_PARITY_MODE_data field details (continued)

Bit position in PCIE_PARITY_MODE_data	PARITY_MODE bus
1	AXI 0 master write address
2	AXI 0 master write response
3	AXI 0 master read data
4	AXI 0 master write data
5	AXI 1 slave read address
6	AXI 1 slave write address
7	AXI 1 slave write response
8	AXI 1 slave read data
9	AXI 1 slave write data
10	AXI 1 master read address
11	AXI 1 master write address
12	AXI 1 master write response
13	AXI 1 master read data
14	AXI 1 master write data
15	AXI 0 slave read address
16	AXI 0 slave write address
17	AXI 0 slave write response
18	AXI 0 slave read data
19	AXI 0 slave write data

Diagram



Provided under NDA to COMPANY PRIVATE Beijing CHJ Automotive Technology Co., Ltd. c8c4676e5492148a8-b251-390d3ab624c1

Fields

Field	Function
31-20 —	Reserved
19-0 PCIE_PARITY_MODE_data	Parity Mode Data Select parity type. Table 63 shows the bit numbers within this field and the buses associated with that number. For each bit: • A value of 0 means "odd parity." • A value of 1 means "even parity."

18.2.9 Debug Control (DEBUG_CONTROL)

Offset

Register	Offset
DEBUG_CONTROL	CCh

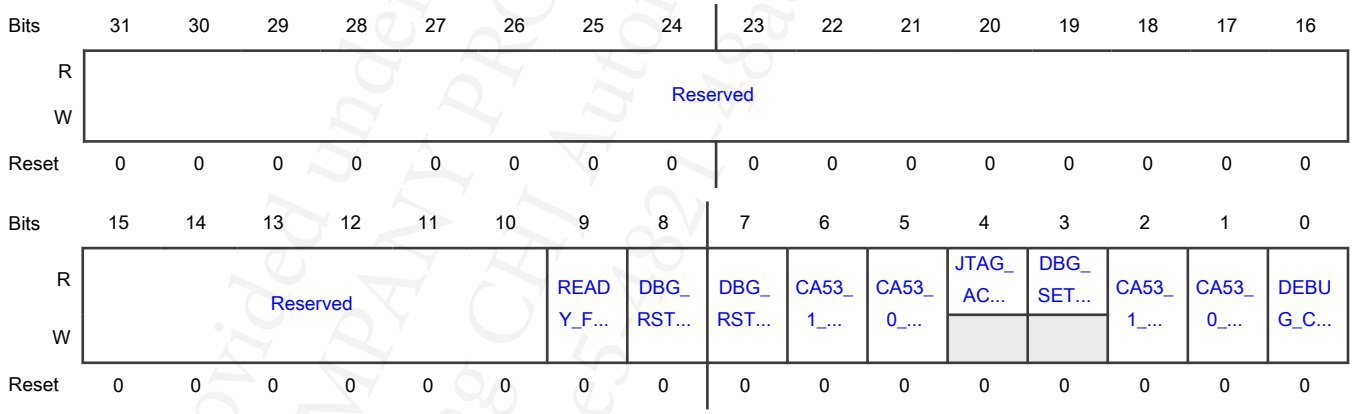
Function

Controls debug features.

A destructive reset resets the non-reserved fields in this register.

Reset phase: Destructive

Diagram



Fields

Field	Function
31-10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
9 READY_FOR_DEBUG	Handshake With Debugger Used to initiate a handshake with the debugger to allow the debugger to start its setup. After the debugger completes its setup, it writes 1 to DBG_SETUP_DONE to allow the boot ROM to execute. 0b - Debug configuration cannot start 1b - Debug configuration can start
8 DBG_RST_MSK_1a	Mask Domain Reset Allows you to mask a debug domain reset from a system functional reset. 0b - Not masked 1b - Masked
7 DBG_RST_MSK_0a	Mask nPRESETDBG Allows you to mask nPRESETDBG of Cortex-A53 cores from a system functional reset. 0b - Not masked 1b - Masked
6 CA53_1_L2RSTDISABLE	L2RSTDISABLE Driver Input Of Cortex-A53 Cluster 1 Enables or disables the automatic invalidation of L2 data cache at reset. 0b - Enable 1b - Disable
5 CA53_0_L2RSTDISABLE	L2RSTDISABLE Driver Input Of Cortex-A53 Cluster 0 Enables or disables the automatic invalidation of L2 data cache at reset. 0b - Enable 1b - Disable
4 JTAG_ACTIVE	JTAG_ACTIVE Status When this field is 1, it indicates that: • JTAG_ACTIVE is high. • The debugger has established a connection to the TAPs. • You can run the AUX TAP state machine. 0b - JTAG_ACTIVE is low 1b - JTAG_ACTIVE is high
3	Status Of Debug Setup Completion By Debugger The debugger writes 1 to this field when the debugger completes debug setup.

Table continues on the next page...

Table continued from the previous page...

Field	Function
DBG_SETUP_DONE	This field is used in the handshake between the debugger and the boot ROM. 0b - Debug setup not complete 1b - Debug setup complete
2 CA53_1_DBGL1RSTDISABLE	DBGL1RSTDISABLE Driver Input Of Cortex-A53 Cluster 1 Enables or disables the automatic invalidation of L1 data cache at reset. 0b - Enable 1b - Disable
1 CA53_0_DBGL1RSTDISABLE	DBGL1RSTDISABLE Driver Input Of Cortex-A53 Cluster 0 Enables or disables the automatic invalidation of L1 data cache at reset. 0b - Enable 1b - Disable
0 DEBUG_CLK_DISABLE	Disable Clocks Of Debug And Trace Components In Functional Mode Writing 1 to this field disables all the clocks related to debug and trace (except clocks related to DAP and JTAGC to save run-time power. This field takes effect only when no debug is authorized (such as external or self-hosted debug). 0b - Enable clock 1b - Disable clock

18.2.10 Timestamp Control (TIMESTAMP_CONTROL_REGISTER)

Offset

Register	Offset
TIMESTAMP_CONTROL_REGISTER	F0h

Function

Control the following operations:

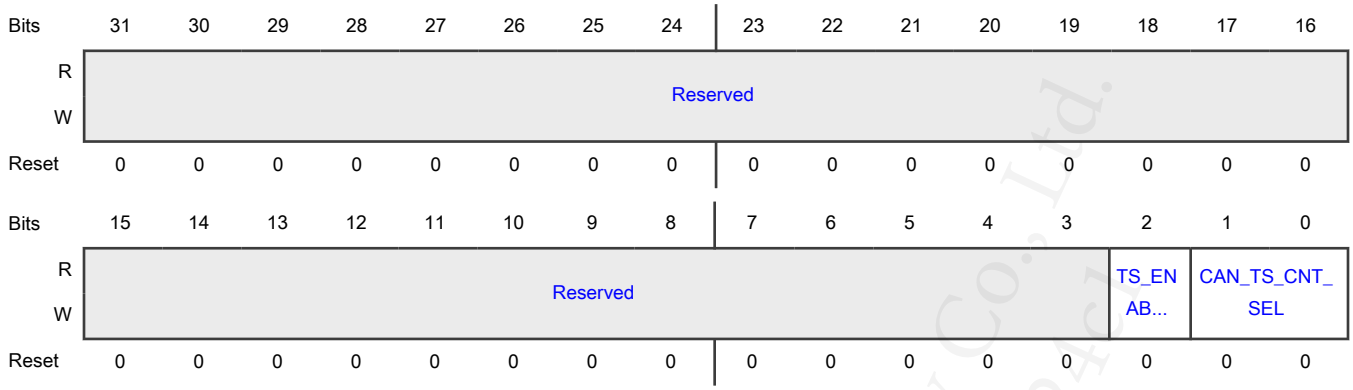
- Select the Timestamp counter source
- Enable Timestamp

This register helps Timestamp to distribute and synchronize up to three counter sources from different clock domains to one common domain, for use by communication peripherals such as FlexCAN and LLCE. See the Timestamp chapter for more information on the sources.

Reset phase: Functional

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c02e5-4821-48a6-b25-590d3ab624c1

Diagram



Fields

Field	Function
31-3 —	Reserved
2 TS_ENABLE	Timestamp Enable Control Enables or disables Timestamp. When this field is 0, the clocks to the FIFOs and flops are gated off to save power. When this field is 1, the asynchronous Timestamp FIFOs read and write during every cycle. 0b - Disable 1b - Enable
1-0 CAN_TS_CNT_SEL	Timestamp Counter Select Selects one of several possible counter sources as the Timestamp output. 00b - Select Timestamp 0 counter output 01b - Select Timestamp 1 counter output 10b - Select Timestamp 2 counter output 11b - Reserved—causes Timestamp output to go to 0

18.2.11 FlexRay OS Tick Input Select (FLEXRAY_OS_TICK_INPUT_SELECT_REG)

Offset

Register	Offset
FLEXRAY_OS_TICK_INPUT_SELECT_REG	F4h

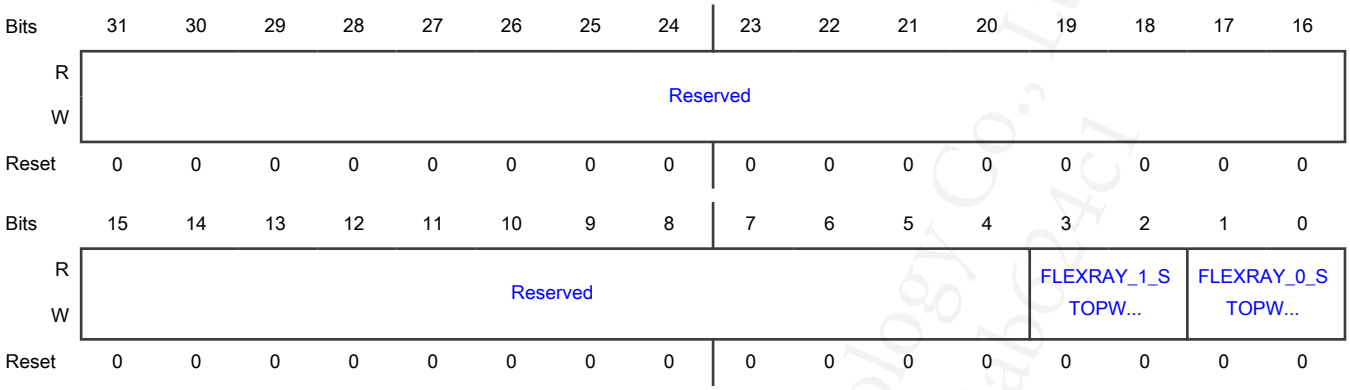
Function

Selects the OS tick source for the FlexRay stopwatch counter.

FlexRay has a 32-bit stopwatch counter that runs on the system clock and increments at every cycle until the counter receives an OS tick event. The fields in this register allow you to select various PIT and STM sources as the OS tick source.

Reset phase: Functional

Diagram



Fields

Field	Function
31-4 —	Reserved
3-2 FLEXRAY_1_STOPWATCH_MUX_SELECT	FlexRay_1 Stopwatch Mux Select 00b - PIT_1 channel number 5 01b - STM_0 int0 10b - STM_1 int0 11b - STM_2 int0
1-0 FLEXRAY_0_STOPWATCH_MUX_SELECT	FlexRay_0 Stopwatch Mux Select 00b - PIT_0 channel number 4 01b - STM_0 int0 10b - STM_1 int0 11b - STM_2 int0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab6e4c1

Chapter 19
Main General Purpose Registers (Main GPRs)

19.1 Introduction

S32G_GPR configures and provides status of various system options.

19.2 Memory map and register definition

19.2.1 S32G_GPR register descriptions

S32G Main GPRs support several miscellaneous functions of the accelerator portion of the chip.

19.2.1.1 S32G_MAIN_GPR memory map

S32G_GPR base address: 4007_CA00h

Offset	Register	Width (In bits)	Access	Reset value
0h	PFE Port Coherency Enable (PFE_COH_EN)	32	RW	0000_0000h
4h	PFE EMAC Interface Mode (PFE_EMACX_INTF_SEL)	32	RW	0000_0000h
20h	PFE EMACX Power Control (PFE_PWR_CTRL)	32	RW	0000_0000h
40h	LLCE Subsystem Status (LLCE_STAT)	32	RO	0000_0000h
44h	LLCE Power Control (LLCE_CTRL)	32	RW	0000_0000h
48h	DDR Urgent Control (DDR_URGENT_CTRL)	32	RW	0000_0000h
4Ch	FlexTimer Global Load Control (FLXTIM_CTRL)	32	WOO	0000_0000h
50h	FlexTimer LDOK Status (FLXTIM_STAT)	32	RO	0000_0000h
54h	Top CMU_FC Status (CMU_STAT)	32	W1C	0000_0000h
58h	Accelerator NoC NoPendingTrans Status (NOC_NOPEND_TRANS)	32	RO	0000_0000h
90h	SerDes RD/WD Toggle Control (PCIE_TOGGLE)	32	RW	0000_0000h
94h	SerDes Toggle Done Status (PCIE_TOGGLEDONE_STAT)	32	RO	0000_0000h
E0h	Generic Control 0 (GENCTRL0)	32	RW	0000_0004h
E4h	Generic Control 1 (GENCTRL1)	32	RW	0000_0000h
F0h	Generic Status 0 (GENSTAT0)	32	RO	0000_0000h

19.2.1.2 PFE Port Coherency Enable (PFE_COH_EN)

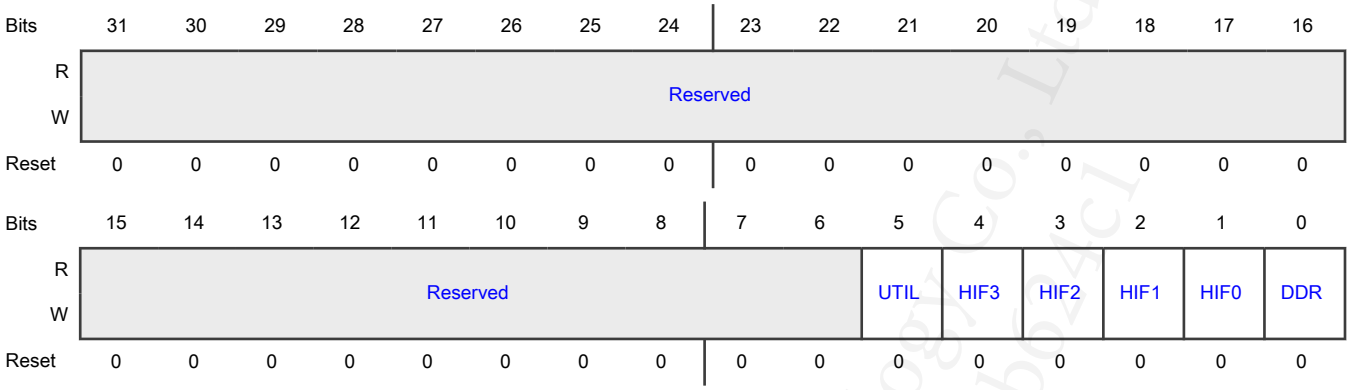
Offset

Register	Offset
PFE_COH_EN	0h

Function

Enables coherency for PFE ports. See "PPFE ports" in the S32GACCELNOCRM document.

Diagram



Fields

Field	Function
31-6 —	Reserved
5 UTIL	PFE UTIL Coherency Enable 0b - PFE UTIL interface not coherent 1b - PFE UTIL interface coherent to A53
4 HIF3	PFE HIF3 Coherency Enable 0b - PFE HIF3 interface not coherent 1b - PFE HIF3 interface coherent to A53
3 HIF2	PFE HIF2 Coherency Enable 0b - PFE HIF2 interface not coherent 1b - PFE HIF2 interface coherent to A53
2 HIF1	PFE HIF1 Coherency Enable 0b - PFE HIF1 interface not coherent 1b - PFE HIF1 interface coherent to A53
1 HIF0	PFE HIF0 Coherency Enable 0b - PFE HIF0 interface not coherent 1b - PFE HIF0 interface coherent to A53
0 DDR	PFE DDR Coherency Enable 0b - PFE DDR interface not coherent 1b - PFE DDR interface coherent to A53

Provided under NDA only
COMPANY CONFIDENTIAL PROPRIETARY
Copyright (C) Autonotive Technology Co., Ltd.
c8462c25-821-48a8-b251-390d3ab624c1

19.2.1.3 PFE EMAC Interface Mode (PFE_EMACX_INTF_SEL)

Offset

Register	Offset
PFE_EMACX_INTF_SEL	4h

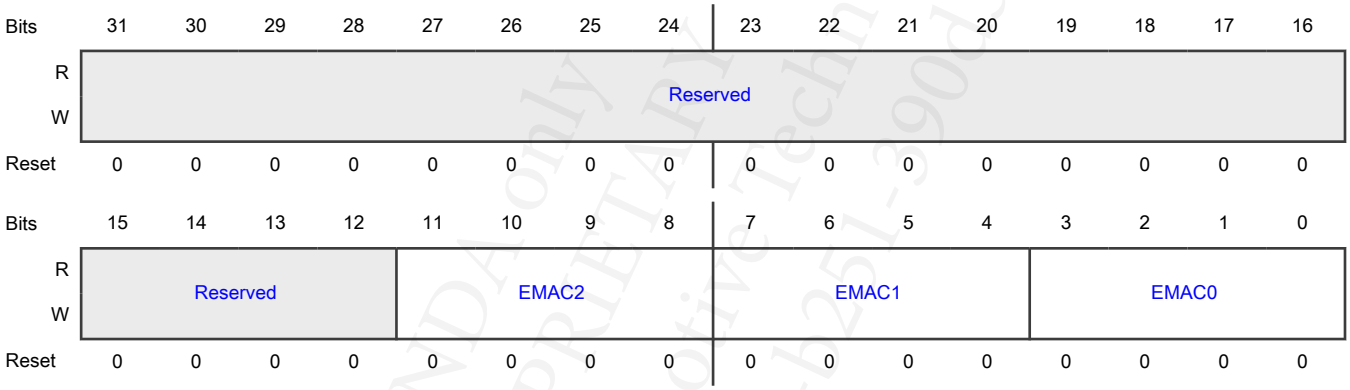
Function

Selects the PFE EMAC interface mode on a per-MAC basis. PFE samples these fields at reset, and therefore must be programmed before bringing PFE out of reset.

The EMAC*n* fields select the Ethernet interface and PHY. 0000b selects the PCIE_0 PHY. The other three valid values route the Ethernet signals to the GPIOs.

Modes not listed are reserved and are not supported.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-8 EMAC2	EMAC2 Interface Select 0000b - Selects GMII and PCIE_0 PHY 0001b - Selects MII and external pads/PHY 0010b - Selects RGMII and external pads/PHY 1001b - Selects RMII and external pads/PHY
7-4 EMAC1	EMAC1 Interface Select 0000b - Selects GMII and PCIE_0 PHY 0001b - Selects MII and external pads/PHY

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0010b - Selects RGMII and external pads/PHY 1001b - Selects RMII and external pads/PHY
3-0 EMAC0	EMAC0 Interface Select 0000b - Selects GMII and PCIE_0 PHY 0001b - Selects MII and external pads/PHY 0010b - Selects RGMII and external pads/PHY 1001b - Selects RMII and external pads/PHY

19.2.1.4 PFE EMACX Power Control (PFE_PWR_CTRL)

Offset

Register	Offset
PFE_PWR_CTRL	20h

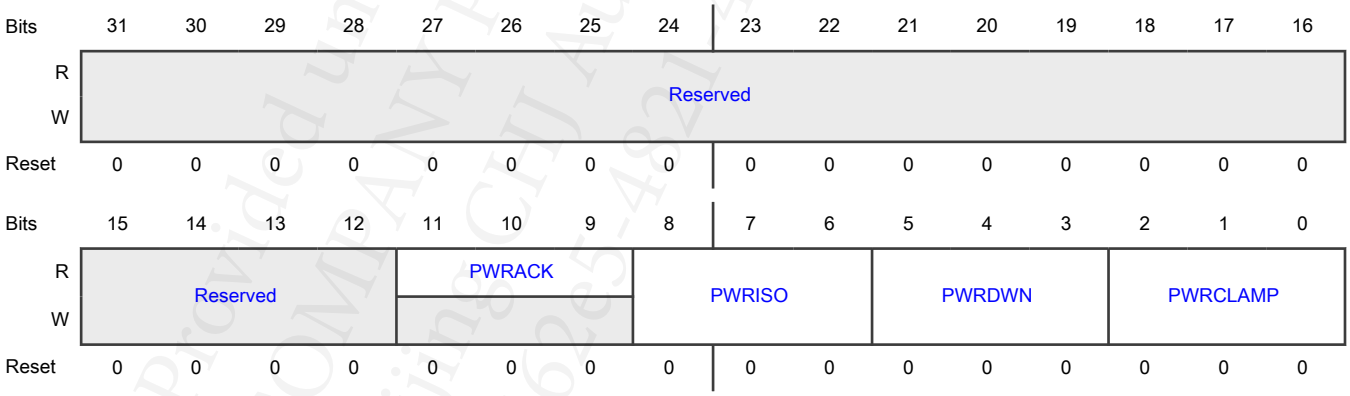
Function

Drives PFE signals:

- emacx_pwr_clamp[2:0]
- emacx_pwr_down_ctrl[2:0]
- emacx_pwr_isolate[2:0]

It also captures the value of the emacx_sbd_pwr_down_ack[2:0] pins.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-4821-48a8-b251-590d3ab624c1

Fields

Field	Function
31-12 —	Reserved
11-9 PWRACK	Power Down Acknoledge Samples outputs emacx_pwr_down_ack[2:0].
8-6 PWRISO	Power Isolate Drives inputs emacx_pwr_isolate[2:0].
5-3 PWRDWN	Power Down Drives inputs emacx_pwr_down_ctrl[2:0].
2-0 PWRCLAMP	PFE Power Clamp Drives inputs emacx_pwr_clamp_ctrl[2:0].

19.2.1.5 LLCE Subsystem Status (LLCE_STAT)

Offset

Register	Offset
LLCE_STAT	40h

Function

Provides status signals for the LLCE subsystem and its M0+ cores.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MISC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LLCE_LSPI_TRIG_FM				LLCE_LSPI_TRIG_RX				SYSRESETREQ				SLEEPING			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e6-4221-48a8-b251-390d3ab624c1

Fields

Field	Function
31-16 MISC	Reserved
15-12 LLCE_LSPI_TRIG_FM	LLCE LSPI Trigger Frame Status of the llce_lspi_trig_frame[3:0] output bus.
11-8 LLCE_LSPI_TRIG_RX	LLCE LSPI Trigger Receive Status of the llce_lspi_trig_receive[3:0] output bus.
7-4 SYSRESETREQ	System Reset Request Status of the LLCE M0+ core's SYS_RESET_REQ status. Each bit in this field represents the status of one of the LLCE cores. The least significant bit corresponds to the c0 SYS_RESET_REQ output signal and the most significant bit corresponds to the c3 SYS_RESET_REQ output signal. The bit value is the corresponding output signal's value.
3-0 SLEEPING	Sleeping SLEEPING status of the LLCE M0+ cores. Each bit in this field represents the status of one of the LLCE cores. The least significant bit corresponds to the c0 SLEEPING output signal and the most significant bit corresponds to the c3 SLEEPING output signal. The bit value is the corresponding output signal's value.

19.2.1.6 LLCE Power Control (LLCE_CTRL)

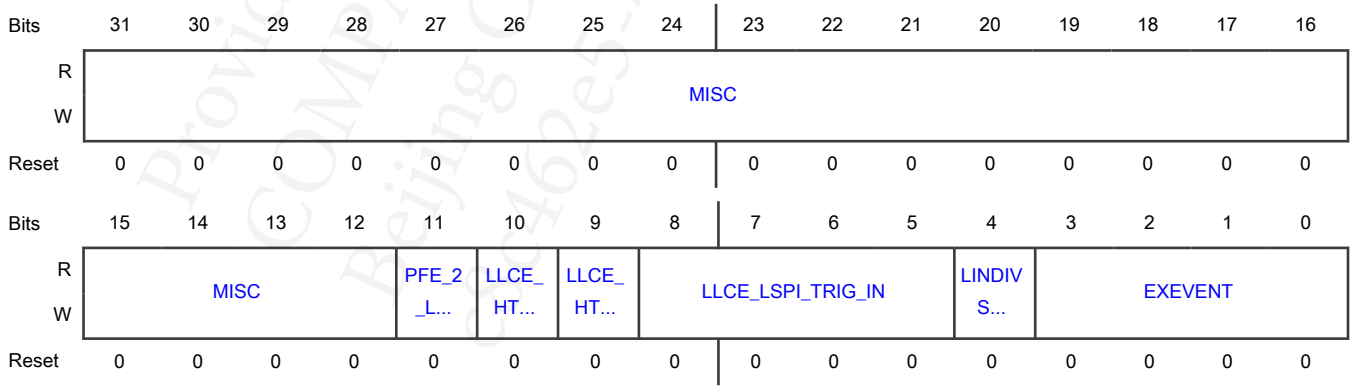
Offset

Register	Offset
LLCE_CTRL	44h

Function

Controls the LLCE subsystem and its M0+ cores.

Diagram



Fields

Field	Function
31-12 MISC	Reserved
11 PFE_2_LLCE_INT_HS_BP	PFE 2 LLCE Interrupt Bypass Controls the PFE-to-LLCE interrupt bypass enable. If this field equals 1, PFE does not clear the interrupt immediately. Instead, it reads the buffer descriptor status (for example, in LLCE RAM) prior to clearing interrupt. Interrupt-stretching logic is bypassed, but synchronization is still required. If this field equals 0, PFE clears its own LLCE interrupt. Logic holds the interrupt long enough for LLCE to latch it in NVIC. 0b - Interrupt bypass disabled 1b - Interrupt bypass enabled
10 LLCE_HTM1_TRC_DIS	LLCE HTM1 Trace Disable Drives the LLCE HTM1 tracedisable input. This functionality can be used by software and not with the debugger attached as the debugger might overwrite this bit.
9 LLCE_HTM0_TRC_DIS	LLCE HTM0 Trace Disable Drives the LLCE HTM0 tracedisable input. This functionality can be used by software and not with the debugger attached as the debugger might overwrite this bit.
8-5 LLCE_LSPI_TRIG_IN	LLCE LSPI Trigger Drives the llce_lspi_trg_input[3:0] bus.
4 LINDIVSEL	LINFlexD Divide Select Selects which clock drives the LLCE ipg_clk_lin_pe_div2 clock. 0b - LIN_CLK (a divided-by-2 version of LIN_BAUD_CLK) drives ipg_clk_lin_pe_div2 1b - LIN_BAUD_CLK drives the ipg_clk_lin_pe_div2xE2 pin
3-0 EXEVENT	External Event Controls the external event inputs of the LLCE M0+ cores. Each bit in this field corresponds to the external event input for a particular core and can be programmed to wake up a core from WFE. The least significant bit corresponds to the c0 EXEV input signal. Bits 1, 2, and 3 correspond to c1, c2, and c3 EXEV input signals, respectively. The value written to this field is written to the input signals.

19.2.1.7 DDR Urgent Control (DDR_URGENT_CTRL)

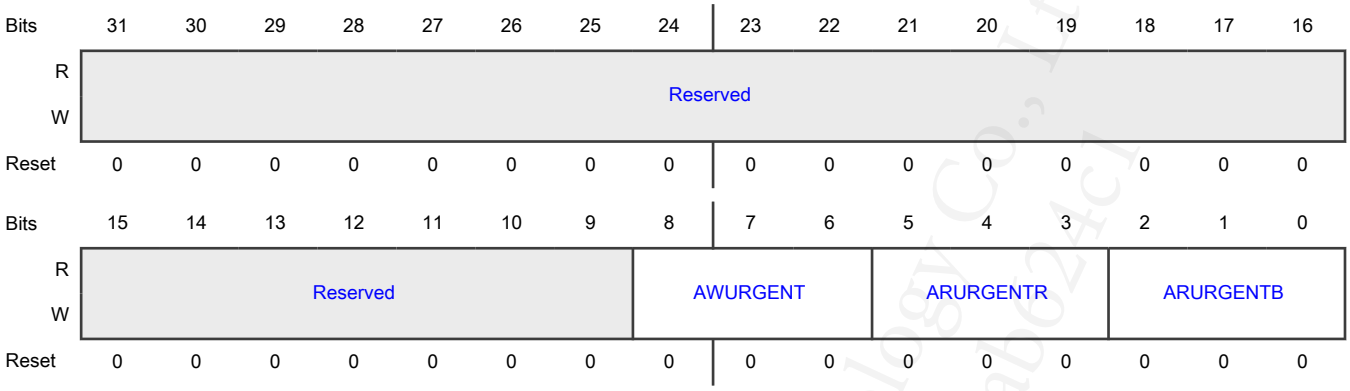
Offset

Register	Offset
DDR_URGENT_CTRL	48h

Function

Controls the values of the arurgent and awurgent inputs of the DDR controller. See the "DDR Subsystem Reference Manual" for more information.

Diagram



Fields

Field	Function
31-9 —	Reserved
8-6 AWURGENT	awurgent Control Drives the following: • Bit 0 - ddr_awurgent_0 • Bit 1 - ddr_awurgent_1 • Bit 2 - ddr_awurgent_2
5-3 ARURGENTR	arurgentr Control Drives the following: • Bit 0 - ddr_arurgentr_0 • Bit 1 - ddr_arurgentr_1 • Bit 2 - ddr_arurgentr_2
2-0 ARURGENTB	arurentb Control Drives the following: • Bit 0 - ddr_arurgentb_0 • Bit 1 - ddr_arurgentb_1 • Bit 2 - ddr_arurgentb_2

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

19.2.1.8 FlexTimer Global Load Control (FLXTIM_CTRL)

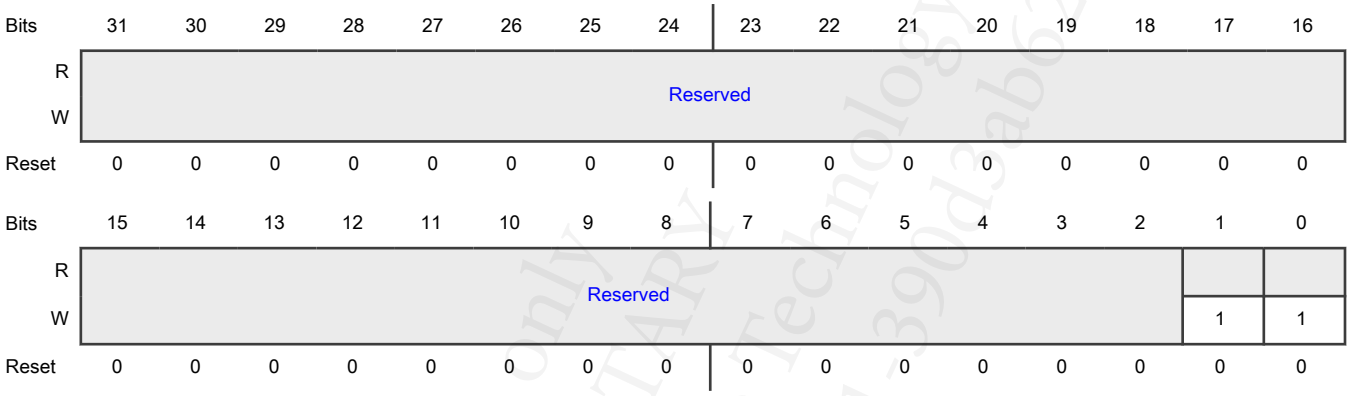
Offset

Register	Offset
FLXTIM_CTRL	4Ch

Function

Controls the FlexTimer global load OK control signals. See the FlexTimer "Global Load" section for details.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 GLOBAL_LDOK_IN1	Global LDOK Input 1 Drives the FTM_1 PWMLOAD[GLDOK] input.
0 GLOBAL_LDOK_IN0	Global LDOK Input 0 Drives the FTM_0 PWMLOAD[GLDOK] input.

19.2.1.9 FlexTimer LDOK Status (FLXTIM_STAT)

Offset

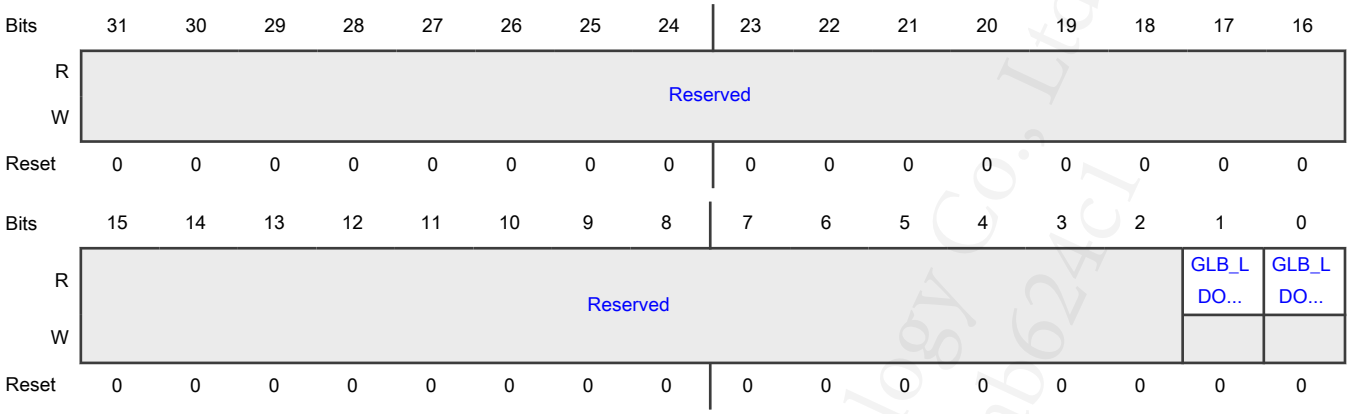
Register	Offset
FLXTIM_STAT	50h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CTJ Automotive Technology Co., Ltd.
c8c46625-4821-48a8-b251-390d3ab624c1

Function

Provides the FlexTimer global LDOK status signals. See the FlexTimer "Global Load" section for details.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 GLB_LDOK_OUT1	Global Load OK Out 1 Value of FTM_1 PWMLOAD[GLDOK].
0 GLB_LDOK_OUT0	Global Load OK Out 0 Value of FTM_0 PWMLOAD[GLDOK].

19.2.1.10 Top CMU_FC Status (CMU_STAT)

Offset

Register	Offset
CMU_STAT	54h

Function

Contains one field, FLL_FHH, that provides FLL and FHH status as described in Table 64.

Table 64. FLL_FHH bits and corresponding CMU_FC events

Bit	Event
0	CMU_39 FLL

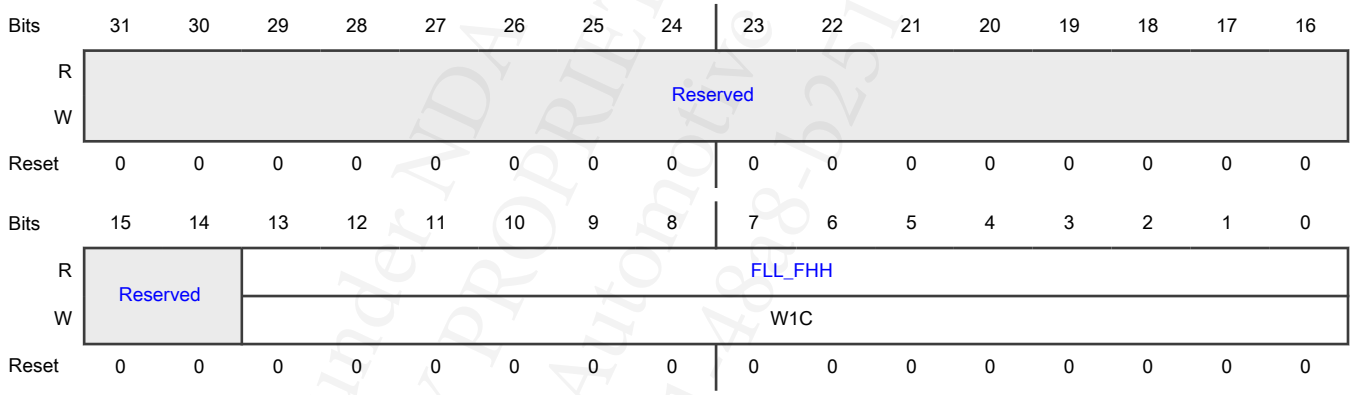
Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c6e5-4821-48a8-b251-390d3ab624c1

Table 64. FLL_FHH bits and corresponding CMU_FC events (continued)

Bit	Event
1	CMU_39 FHH
2	CMU_46 FLL
3	CMU_46 FHH
4	CMU_47 FLL
5	CMU_47 FHH
6	CMU_48 FLL
7	CMU_48 FHH
8	CMU_49 FLL
9	CMU_49 FHH
10	CMU_50 FLL
11	CMU_50 FHH
12	CMU_51 FLL
13	CMU_51 FHH

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0 FLL_FHH	CMU FLL FHH Status CMU FLL and FHH interrupt event status 1. See Table 64 for the encoding of the individual bits in this field. Resettable on destructive reset.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8162e54821f4868-b251-390d3ab624c1

19.2.1.11 Accelerator NoC NoPendingTrans Status (NOC_NOPEND_TRANS)

Offset

Register	Offset
NOC_NOPEND_TRANS	58h

Function

Presents the Accelerator NoC NoPendingTrans status signals.

A 1 indicates there are no pending transactions for that port of the NoC. A 0 indicates there are pending transactions for that port. You must wait until noPendingTrans is 1 when deciding to place a NoC domain in Idle.

Table 65. STAT encoding

Bit	Description
0	m_pcie_1 noPendingTrans status
1	m_ppfe_ddr noPendingTrans status
2	m_ppfe_hif0 noPendingTrans status
3	m_ppfe_hif1 noPendingTrans status
4	m_ppfe_hif2 noPendingTrans status
5	m_ppfe_hif3 noPendingTrans status
6	m_ppfe_util noPendingTrans status
7	m_usb noPendingTrans status
8	m_to_off_chassis noPendingTrans status
9	m_dbg_apb noPendingTrans status
10	service_regs noPendingTrans status
11	s_pcie_1 noPendingTrans status
12	s_ppfe noPendingTrans status
13	s_off_chassis_dram_0 noPendingTrans status
14	s_off_chassis_sram_0 noPendingTrans status
15	s_llce noPendingTrans status
16	s_stdby_sram noPendingTrans status
17	s_concerto noPendingTrans status
18	s_from_off_chassis noPendingTrans status
19	s_ocp2ips noPendingTrans status
20:31	Reserved. Behaves as read-only-one.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a0a-ab62-1c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STAT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STAT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STAT	Port Status Provides the port status as shown in Table 65.

19.2.1.12 SerDes RD/WD Toggle Control (PCIE_TOGGLE)

Offset

Register	Offset
PCIE_TOGGLE	90h

Function

Controls SerDes_0 and SerDes_1 toggle signals. This register is applicable only when DMA is in linked mode and will be ignored when DMA is in non-linked mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PCIE0_RD_TOGG				PCIE0_WD_TOGG				PCIE1_RD_TOGG				PCIE1_WD_TOGG			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c10e54-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-16 —	Reserved
15-12 PCIE0_RD_TOGG	Drives the RD TOGGLE [3:0] inputs of SerDes_0.
11-8 PCIE0_WD_TOGG	Drives the WD TOGGLE [3:0] inputs of SerDes_0.
7-4 PCIE1_RD_TOGG	Drives the RD TOGGLE [3:0] inputs of SerDes_1.
3-0 PCIE1_WD_TOGG	Drives the WD TOGGLE [3:0] inputs of SerDes_1.

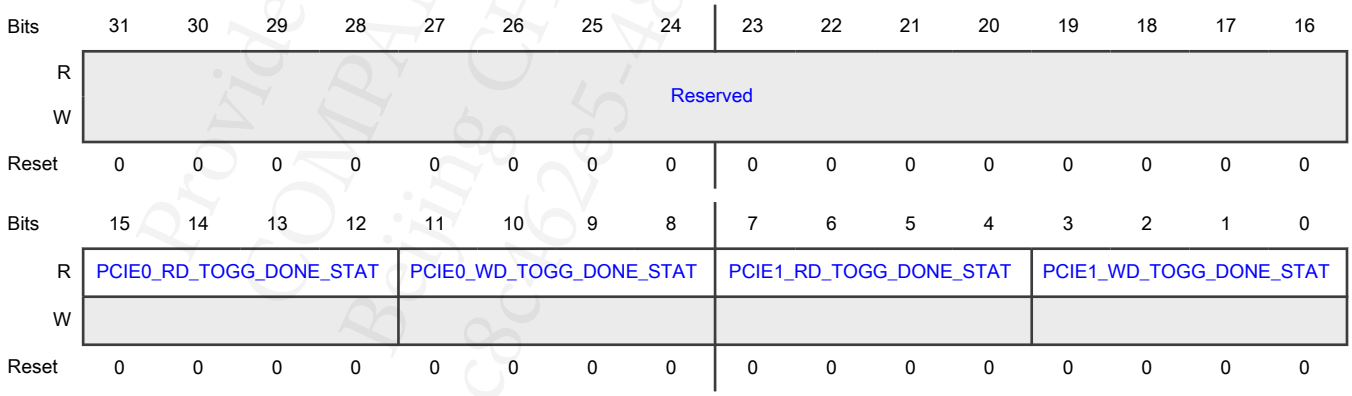
19.2.1.13 SerDes Toggle Done Status (PCIE_TOGGLEDONE_STAT)

Offset

Register	Offset
PCIE_TOGGLEDONE_STAT	94h

Function
Provides status of SerDes_0 and SerDes_1 toggle done signals.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8d6255-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-16 —	Reserved
15-12 PCIE0_RD_TOGG_DONE_STAT	Status of the RD TOGGLE [3:0] outputs of SerDes_0.
11-8 PCIE0_WD_TOGG_DONE_STAT	Status of the WD TOGGLE [3:0] outputs of SerDes_0.
7-4 PCIE1_RD_TOGG_DONE_STAT	Status of the RD TOGGLE [3:0] outputs of SerDes_1.
3-0 PCIE1_WD_TOGG_DONE_STAT	Status of the WD TOGGLE [3:0] outputs of SerDes_1.

19.2.1.14 Generic Control 0 (GENCTRL0)

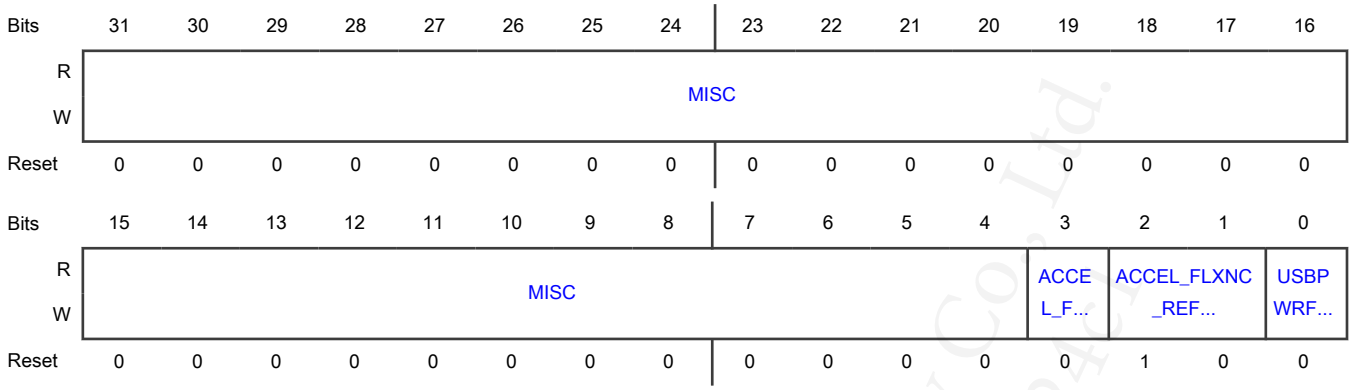
Offset

Register	Offset
GENCTRL0	E0h

Function
Controls certain NoC and USBOTG functions.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-4 MISC	Reserved
3 ACCEL_FLXNC_RD0_IDLEREQ	Accelerator NoC RD0 IdleReq Drives the accelerator NoC's IdleReq signal for the main domain. Writing 1 causes NoC to enter an Idle state. NoC waits for any existing transactions (except for PFE and LLCE portions, which have their own Idle controls in PCM) to complete, then asserts the Idle signal (bit 1 in Generic Status 0 (GENSTAT0)). 0b - Accelerator NoC in normal operation 1b - Request accelerator NoC to go into Idle state
2-1 ACCEL_FLXNC_REF_DIV	Accelerator NoC Reference Divide Controls the rate of the NoC timeout counter reference. The reference input clock to the counter is 100 MHz. NoC slaves time out at 4096 counts and masters at 32768 counts. 00b - Divide by 2 (50 MHz reference) 01b - Divide by 4 (25 MHz reference) 10b - Divide by 8 (12.5 MHz reference) 11b - Divide by 16 (6.25 MHz reference)
0 USBPWRFLT	USB Power Fault Drives the USBOTG otg_vbus_pwr_fault power fault indicator.

19.2.1.15 Generic Control 1 (GENCTRL1)

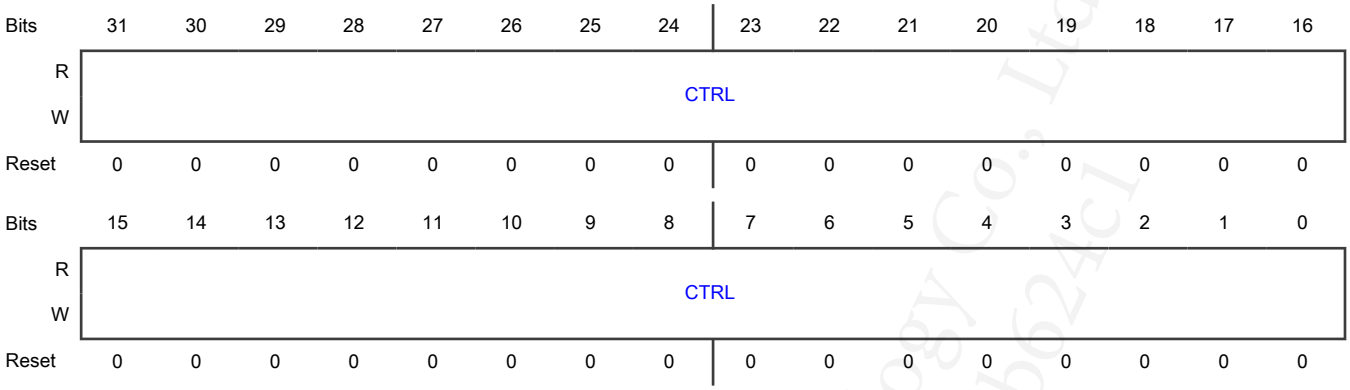
Offset

Register	Offset
GENCTRL1	E4h

Function

Controls SGMII_CSEL.

Diagram



Fields

Field	Function
31-0 CTRL	Bit 0 - SGMII_CSEL • 0b - Not running PFE EMAC0 in SGMII mode • 1b - Running PFE EMAC0 in SGMII mode Bits 32:1 - Reserved

19.2.1.16 Generic Status 0 (GENSTAT0)

Offset

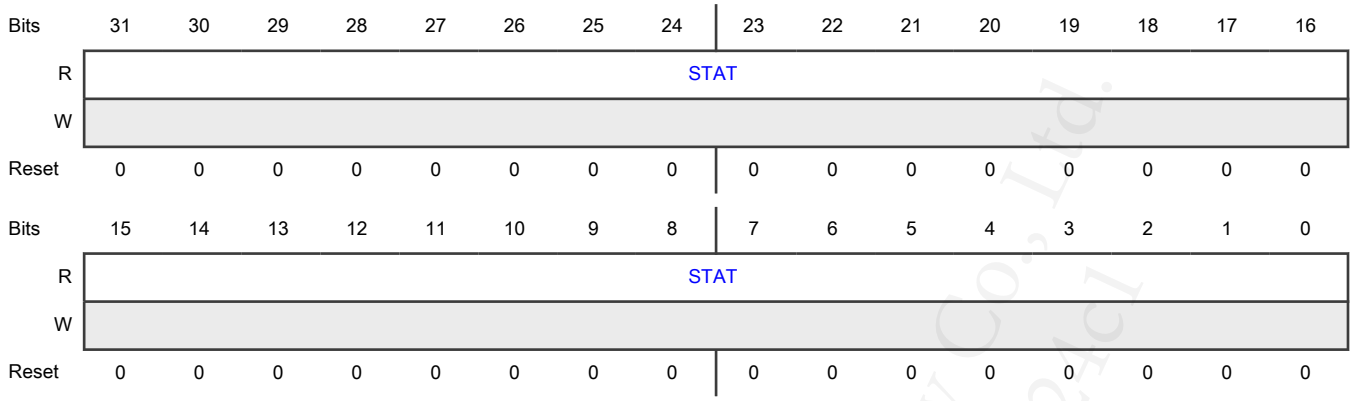
Register	Offset
GENSTAT0	F0h

Function

Provides NoC main domain (RD0) Idle and IdleAck status.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHI Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 STAT	Status Bit 0 - NoC main domain (RD0) IdleAck status Bit 1 - NoC main domain (RD0) Idle status Bits 31-2 - Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8ab624c1

Chapter 20
Standby General Purpose Registers (Stby GPRs)

20.1 Introduction

S32G_STDBY_GPR controls various functions related to Standby mode.

20.2 S32G_STDBY_GPR register descriptions

These registers reside inside the Standby domain. Therefore, the values stored in these registers are not lost when the chip enters Standby mode.

20.2.1 S32G_STDBY_GPR memory map

S32G_STDBY_GPR base address: 4007_CB00h

Offset	Register	Width (In bits)	Access	Reset value
4h	WKUP Pad Pullup/Pulldown Select (WKUP_PUS)	32	RW	0000_0000h

20.2.2 WKUP Pad Pullup/Pulldown Select (WKUP_PUS)

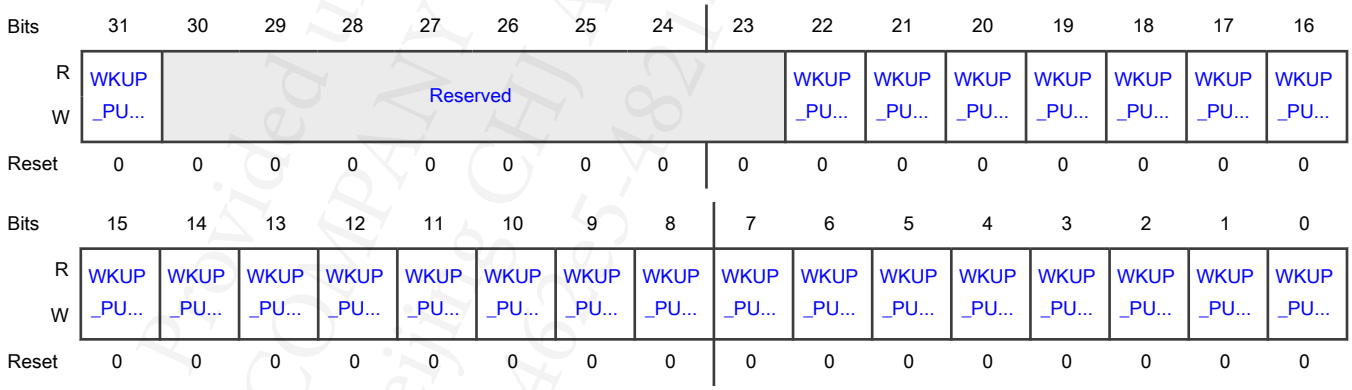
Offset

Register	Offset
WKUP_PUS	4h

Function

Selects pullup or pulldown for each WKUP pad.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e05-4824-48a8-b251-39063ab6240l

Fields

Field	Function
31 WKUP_PU_OVERRIDE	WKUP Pullup Override This field selects one of two methods to control the WKUP pins. If the value is 1, that overrides SIUL2's MSCR[PUE] and MSCR[PUS] fields. **NOTE** It is observed that the wakeup pads configured pulled up or pulled down from the SIUL2 module retains their pull in standby mode also. This occurs even when the WKUP_PUS[WKIP_PU_OVERRIDE] = 0, which disabled the WKPU to drive the pull on the wakeup pads. 0b - Use SIUL2's MSCR[PUE] and MSCR[PUS] fields 1b - Use WKPU and GPR modules
30-23 —	Reserved
22-0 WKUP_PUSn	WKUP Pad n Pullup Select Selects pullup or pulldown on WKUP pad n if SIUL2's MSCR[PUE]=1 on the pad. 0b - Pulldown 1b - Pullup

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 21
Register Protection (REG_PROT)

21.1 Introduction

The REG_PROT module offers a mechanism to protect defined memory-mapped address locations, in a module under protection, from being written. The address locations that can be protected are module specific.

The module is located between the module under protection and the platform peripheral interface. This is shown in the following figure.

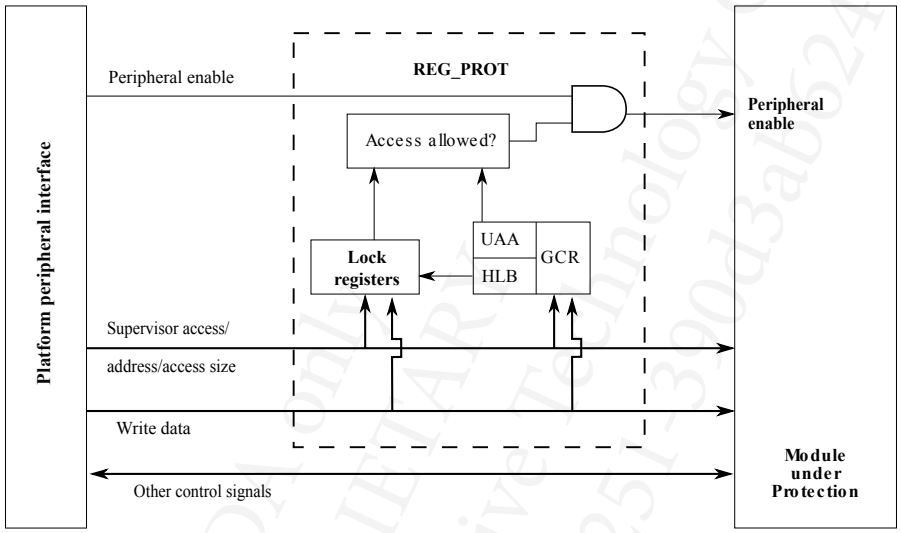


Figure 46. Block diagram

21.2 Features

REG_PROT includes these distinctive features:

- Write accesses for the module under protection are restricted to the supervisor mode only.
- Registers with 1 KB to 32 KB of memory-mapped address space, depending on the PROT_MEM parameter, can be dynamically locked. See Memory map and register definition.
- Write to address mirror automatically sets the corresponding lock bit.
- After the lock bits are configured, they can be protected from changes.

21.3 Modes of operation

The REG_PROT module is operable when the module under protection is operable.

21.4 External signal description

There are no external signals.

21.5 Memory map and register definition

This section provides a detailed description of the memory map of a module with register protection.

Following is the memory map for a module that is protected by the REG_PROT module.

Table 66. Module memory map

Protection size/ PROT_MEM	Area	Area size	Use	Offset
1 kB	Area 1	1 kB	Module Register MR0 — MR1023	0x0000-0x03FF
	Area 2	1 kB	Module Register MR0 — MR1023 Set Soft Lock Bit LMR0 — LMR1023	0x0400-0x07FF
	Area 3	256 B	Soft Lock Bit Register SLBR0 — SLBR255	0x0800-0x08FF
	Area 4	1.75 kB	Global Configuration Register (GCR)	**0x0900-0x0FFC**
2 kB	Area 1	2 kB	Module Register MR0 — MR2047	0x0000-0x07FF
	Area 2	2 kB	Module Register MR0 — MR2047 Set Soft Lock Bit LMR0 — LMR2047	0x0800-0x0FFF
	Area 3	512 B	Soft Lock Bit Register SLBR0 — SLBR511	0x1000-0x11FF
	Area 4	3.5 kB	Global Configuration Register (GCR)	**0x1200-0x1FFC**
4 kB	Area 1	4 kB	Module Register MR0 — MR4095	0x0000-0x0FFF
	Area 2	4 kB	Module Register MR0 — MR4095 Set Soft Lock Bit LMR0 — LMR4095	0x1000-0x1FFF
	Area 3	1 kB	Soft Lock Bit Register SLBR0 — SLBR1023	0x2000-0x23FF
	Area 4	3 kB	Global Configuration Register (GCR)	0x2400-0x27FC
8 kB	Area 1	8 kB	Module Register MR0 — MR8191	0x0000-0x1FFF
	Area 2	8 kB	Module Register MR0 — MR8191 Set Soft Lock Bit LMR0 — LMR8191	0x2000-0x3FFF
	Area 3	2 kB	Soft Lock Bit Register SLBR0 — SLBR2047	0x4000-0x47FF
	Area 4	2 kB	Global Configuration Register (GCR)	0x4800-0x4FFC
16 kB	Area 1	16 kB	Module Register MR0 — MR16383	0x0000-0x3FFF
	Area 2	16kB	Module Register MR0 — MR16383 Set Soft Lock Bit LMR0 — LMR16383	0x4000-0x7FFF
	Area 3	4 kB	Soft Lock Bit Register SLBR0 — SLBR4095	0x8000-0x8FFF
	Area 4	4 kB	Global Configuration Register (GCR)	0x9000-0x9FFC
32 kB	Area 1	32 kB	Module Register MR0 — MR32767	0x0000-0x7FFF
	Area 2	32 kB	Module Register MR0 — MR32767 Set Soft Lock Bit LMR0 — LMR32767	0x8000-0xFFFF
	Area 3	8 kB	Soft Lock Bit Register SLBR0 — SLBR8191	0x10000-0x11FFF
	Area 4	**4 kB**	Global Configuration Register (GCR)	**0x12000-0x12FFC**

NOTE

Reserved registers in area #1 are handled as specified in the protected module's documentation.

NOTE

There is a single register in configuration space that corresponds to Area 4 while the size mentioned in Area size column is to turn the whole area multiple of 4 KB. See the Offset column to know the exact area for GCR registers.

- Area 1 can take the memory space of 1 KB/2 KB/4 KB/8 KB/16 KB/32 KB, which holds the normal functional module registers and is transparent for all read/write operations.
- Area 2 can take the memory space of 1 KB/2 KB/4 KB/8 KB/16 KB/32 KB and is a mirror of Area 1. For an example, if Area 1 takes 8 KB of memory space, then read/write access to an offset 0x2000+X reads/writes the register at offset X. However, a write access to offset 0x2000+X additionally sets the optional soft lock bits for this offset X in the same cycle as the register at offset X is written. This provides for an automatic write and lock operation. Not all registers in Area 1 need to have protection defined by the associated soft lock bits. For unprotected registers at offset Y, accesses to offset 0x2000+Y are identical to accesses at offset Y.
- Area 3 can take the memory space of 256 KB/512 KB/1 KB/2 KB/4 KB/8 KB and hold soft lock bits, one bit per byte in Area 1. The four soft lock bits associated with one module register word are arranged at byte boundaries in the memory map. The Soft Lock Bit registers can be directly written using a bit mask.
- Area 4 can take the memory space of 1.75 KB/3.5 KB/3 KB/2 KB/4 KB/4 KB and holds the configuration bits of the protection mode. There is one configuration hard lock bit per module that prevents all further modifications to the soft lock bits and can only be cleared by a system reset after it is set. If other bits are set, they allow user access to the protected module.

If you access a locked byte with a write transaction, a transfer error is issued to the system and the write transaction is not executed. This is true even if not all accessed bytes are locked.

Accessing unimplemented 32-bit registers in Area 3 results in a transfer error. In Area 4, there will not be any transfer error for the address range mentioned in the table shown below because these address spaces of Area 4 are mapped to the GCR register. Accesses in Area 4, beyond the address range mentioned in the Module memory map, are prohibited and might or might not result into a bus abort.

21.5.1 Memory map

21.5.2 Module registers (MR)

Each functional module has its own unique set of registers. This chapter refers to these registers generically as module registers (MR), which exist in the lower module memory space, and receive protection from the REG_PROT module.

21.5.3 MR and set soft lock bit (LMR)

This is memory area #2 that provides mirrored access to module registers with the side-effect of setting soft lock bits in case of a write access to a register that is defined as protectable by the locking mechanism. Each MR is protectable by one associated bit in SLBRn [SLBm], according to the mapping described in Soft Lock Bit Register (SLBRn).

21.6 REG_PROT register descriptions

This register information documents memory areas #3 and #4. To extrapolate to registers for other PROT_MEM values, see the information in Memory map and register definition.

NOTE

REG_PROT registers do not have their own base address. They occupy a portion of the memory space of each module instance that has REG_PROT protection. As a result, this information presents the REG_PROT "base address" as 0h.

21.6.1 REG_PROT memory map

REG_PROT base address: 0h

Offset	Register	Width (In bits)	Access	Reset value
0h	Soft Lock Bit Register (SLBRn)	8	RW	00h
900h	Global Configuration Register (GCR)	32	RW	0000_0000h

21.6.2 Soft Lock Bit Register (SLBRn)

Offset

Register	Offset
SLBRn	0h

Function

NOTE

This register information represents the case where PROT_MEM is 1.

These registers hold the Soft Lock Bits (SLBs) for the protected registers in memory area #1, which is the normal register address space of the protected module. As many number of Soft Lock Bit Registers (SLBR*n*) can be implemented, depending on the number of protected module register bytes. Each SLBR*n* has four soft lock fields (SLB0-SLB3), each of which controls write access to a byte in memory area #1. Each soft lock field also has a corresponding write enable field in the same register that controls whether the soft lock field can be written. The following table shows the mapping between the soft lock fields to the bytes in memory area #1. See Module memory map table for SLBRn address offsets. For SLBRn address offsets, see the REG_PROT sheet attached to this document.

Table 67. SLBs vs. protected address

SLB	Protected address
SLBR0[SLB0]	MR0
SLBR0[SLB1]	MR1
SLBR0[SLB2]	MR2
SLBR0[SLB3]	MR3
SLBR1[SLB0]	MR4
SLBR1[SLB1]	MR5
SLBR1[SLB2]	MR6
SLBR1[SLB3]	MR7
SLBR2[SLB0]	MR8
...	...

Diagram

Bits	7	6	5	4	3	2	1	0
R	0	0	0	0	SLB0	SLB1	SLB2	SLB3
W	WE0	WE1	WE2	WE3				
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7 WE0	Write enable fields for SLB WE0 enables writing to SLB0. 0b - SLB is not modified. 1b - Value is written to SLB.
6 WE1	Write enable fields for SLB WE1 enables writing to SLB1. 0b - SLB is not modified. 1b - Value is written to SLB.
5 WE2	Write enable fields for SLB WE2 enables writing to SLB2. 0b - SLB is not modified. 1b - Value is written to SLB.
4 WE3	Write enable bits for SLB WE3 enables writing to SLB3. 0b - SLB is not modified. 1b - Value is written to SLB.
3 SLB0	Soft lock fields for one MRn register SLB0 can block accesses to MR[n *4 + 0]. 0b - Associated MRn byte is unprotected and writable. 1b - Associated MRn byte is locked against write accesses.
2 SLB1	Soft lock bits for one MRn register SLB1 can block accesses to MR[n *4 + 1]. 0b - Associated MRn byte is unprotected and writable. 1b - Associated MRn byte is locked against write accesses.
1 SLB2	Soft lock fields for one MRn register SLB2 can block accesses to MR[n *4 + 2].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Associated MRn byte is unprotected and writable. 1b - Associated MRn byte is locked against write accesses.
0 SLB3	Soft lock fields for one MRn register SLB3 can block accesses to MR[n *4 + 3]. 0b - Associated MRn byte is unprotected and writable. 1b - Associated MRn byte is locked against write accesses.

21.6.3 Global Configuration Register (GCR)

Offset

Register	Offset
GCR	900h

Function

See Module memory map table for GCR address offsets.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HLB	0							UAA	0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 HLB	Hard Lock Bit This field cannot be cleared after it is set by the software. It can only be cleared by a system reset. 0b - All SLB fields are accessible and can be modified. 1b - All SLB fields are write protected and cannot be modified.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-24 —	Reserved
23 UAA	User Access Allowed Controls the user and supervisor mode access to registers of the module under protection. 0b - The registers in the module under protection can only be written in the supervisor mode. All the write accesses in non-supervisor mode are not executed and a transfer error is issued. This access restriction is in addition to any access restrictions imposed by the protected IP module. 1b - The registers in the module under protection can be accessed in the mode defined for the module registers without any additional restrictions.
22-0 —	Reserved

21.7 Functional description

The following sections describe the functional characteristics of the REG_PROT module.

21.7.1 General

This module provides a generic register (address) write-protection mechanism. The protection size can be:

- 32 bits (address == multiples of 4)
- 16 bits (address == multiples of 2)
- 8 bits (address == multiples of 1)

The addresses that are protected and the protection size depends on the chip and/or module.

For all addresses that are protected, there are SLBRn[SLBm] bits that specify whether the address is locked. When an address is locked, it can only be read but not written in any mode (supervisor/normal). If an address is unprotected, the corresponding SLBRn[SLBm] bit is always 0b0, no matter what software is writing to.

NOTE

For more information on register protection specification, see the REG_PROT sheet attached to this document.

21.7.2 Change lock settings

To change the setting whether an address is locked or unlocked, the corresponding SLBRn.SLBm bit needs to be changed. This can be done using the following methods:

- Modify the SLBRn[SLBm] directly by writing to area #3
- Set the SLBRn[SLBm] bit(s) by writing to the mirror module space (area #2)

Both methods are explained in the following sections.

21.7.2.1 Change lock settings directly via area #3

In memory area #3, the lock bits are located. They can be modified by writing to them. Each SLBR*n*.SLB*m* bit has a mask bit SLBR*n*.WE*m*, which protects it from being modified. This masking makes clear-modify-write operations unnecessary.

The following figure shows two modification examples for 8-bit wise protected area. In the example on the left, there is a write access to the SLBR*n* register specifying a mask value that allows modification of all SLBR*n*.SLBm bits. The example on the right specifies a mask that only allows modification of the bits SLBR*n*.SLB[1:3].

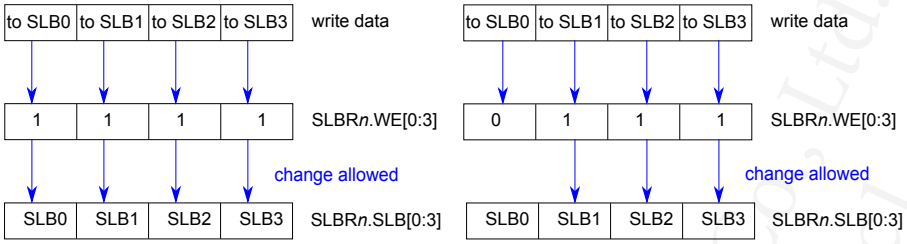


Figure 47. Change lock settings directly via area #3

The following figure shows two modification examples for 16-bit wise protected area. On the right side of the figure, it is shown that the data written to SLBR*n*.SLB[0] is automatically written to SLBR*n*.SLB[1] as well. This is done because the address reflected by SLBR*n*.SLB[0] is protected 16-bit wise. Note that in this case, the write enable SLBR*n*.WE[0] must be set while SLBR*n*.WE[1] does not matter. As the enable bits SLBR*n*.WE[2:3] are cleared, the lock bits SLBR*n*.SLB[2:3] remain unchanged.

In the example on the left side of the figure, the data written to SLBR*n*.SLB[0] is mirrored to SLBR*n*.SLB[1] and the data written to SLBR*n*.SLB[2] is mirrored to SLBR*n*.SLB[3]. This is done because the address reflected by SLBR*n*.SLB[0] and SLBR*n*.SLB[2] are protected 16-bit wise. Note that in this case, the write enable SLBR*n*.WE[0] and SLBR*n*.WE[2] must be set while SLBR*n*.WE[1] and SLBR*n*.WE[3] don't matter.

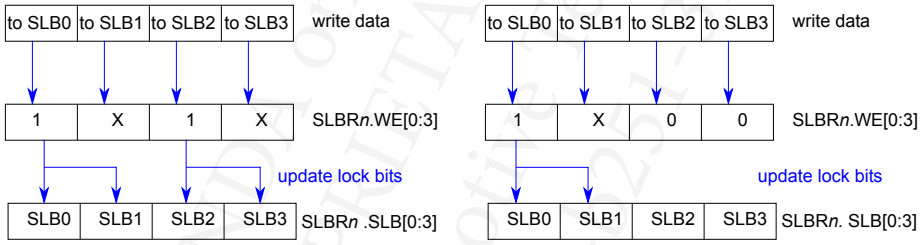


Figure 48. Change lock settings for 16-bit protected addresses

The following figure shows modification example for 32-bit wise protected area. When SLBR*n*.WE[0] is set, the data written to SLBR*n*.SLB[0] is automatically written to SLBR*n*.SLB[1:3] as well. Otherwise, SLBR*n*.SLB[0:3] remains unchanged. Note that in this case, the write enable SLBR*n*.WE[0] must be set while SLBR*n*.WE[1:3] does not matter.

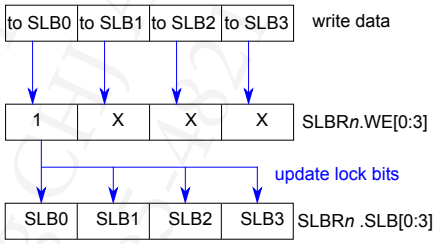


Figure 49. Change lock settings for 32-bit protected addresses

The following figure shows a mixed protection size configuration.

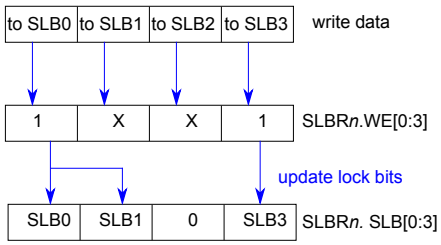


Figure 50. Change lock settings for mixed protection

The data written to SLBR*n*.SLB[0] is mirrored to SLBR*n*.SLB[1] as the corresponding register is 16-bit protected. The data written to SLBR*n*.SLB[2] is blocked because the corresponding register is unprotected. The data written to SLBR*n*.SLB[3] is written to SLBR*n*.SLB[3] as the corresponding register is 8-bit protected.

21.7.2.2 Enable locking via mirror module space (area #2)

It is possible to enable locking for a register after writing to it. See below examples for 8kB protection size (offset of the mirrored registers 0x2000). To do so, the mirrored module address space must be used. The following figure shows an example.

When writing to address 0x0008, the registers MR9 and MR8 in the protected module are updated. The corresponding lock bits remain unchanged (see the left part of the following figure).

When writing to address 0x2008, the registers MR9 and MR8 in the protected module are updated. The corresponding lock bits SLBR2.SLB[0:1] are set while the lock bits SLBR2.SLB[2:3] remain unchanged (see the right part of the following figure).

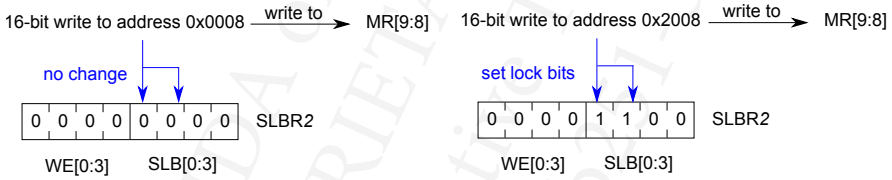


Figure 51. Enable locking via mirror module space (area #2)

The following figure shows an example where some addresses are protected and some are not. Addresses 0x0C and 0x0D are unprotected. Therefore, their corresponding lock bits SLBR3.SLB[0:1] are always 0b0 (shown in bold). When doing a 32-bit write access to address 0x200C, only lock bits SLBR3.SLB[2:3] are set while bits SLBR3.SLB[0:1] stay 0b0.

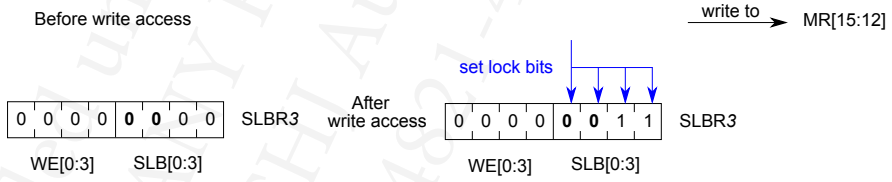


Figure 52. Enable locking for protected and unprotected addresses

NOTE

Lock bits can only be set via writes to the mirror module space. Reads from the mirror module space do not change the lock bits.

21.7.2.3 Write protection for locking bits

Changing the locking bits through any of the procedures mentioned in Change lock settings directly via area #3 and Enable locking via mirror module space (area #2) is only possible as long as the field GCR.HLB is cleared. After you write 1 to this field, the locking bits cannot be modified until there is a system reset.

21.7.3 Access errors

The protection module generates transfer errors under several circumstances. For the area definition, seeMemory map and register definition.

- If accessing area #1 or area #2, the protection module passes on access error from the underlying module under protection.
- If the user mode is not allowed, the user writes to all areas, asserts a transfer error, and the writes get blocked.
- If accessing unimplemented 32-bit registers in area #3 and area #4, a transfer error is asserted.
- If writing to a register in area #1 and area #2 with Soft Lock field set for any of the affected bytes, a transfer error is asserted and the write will get blocked. Also, the complete write operation to non-protected bytes in this word is ignored.
- If writing to a Soft Lock Register in area #3 with the Hard Lock field being set, a transfer error is asserted.
- For any write operation in any access mode to area #2 while Hard Lock field GCR.HLB is set

21.7.4 Sequence

You must configure the REG_PROT first. And then, you need to write 1 to the HLB field in REG_PROT. This ensures that registers under protection are not updated in the user access mode. You need to follow these steps:

1. Enable XRDC protection for the module to enable writes in the supervisor mode only. This step is mandatory, if protected registers of the module are expected to be configured multiple times.
2. Configure REG_PROT to enable write on registers of module under protection.
3. Configure the registers of module under protection.
4. Configure the REG_PROT for the required SLB fields.
5. Set the HLB field in REG_PROT. This step is mandatory, if protected registers of module are expected to be configured only once.

21.8 Initialization/application information

The following sections contain information related to initialization and use of the Register Protection module in applications.

21.8.1 Reset

The reset state of each individual bit is shown within the Register Description section. In summary, after reset, locking for all MRn registers is disabled. You can access the registers only in the Supervisor mode.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a8-b251-390d3a6624c3

Chapter 22
Wakeup Unit (WKPU)

22.1 Chip-specific WKPU information

22.1.1 WKPU configuration

The following table shows the WKPU configuration on this chip.

Table 68. WKUP expose pads

Number of	Value
NMI sources	1
Internal interrupt sources	1
External interrupt sources	23
Glitch filters for external interrupts	23
External interrupt vectors	3

NOTE

Wake-up sources 0 to 22 map to the pads with the corresponding WKPUx signal and I/O supplies are described in the attached IOMUX spreadsheet, IO Signal tab. Wake-up source 31 is mapped to the internal RTC module. For the IO voltage supply range, refer to the Absolute Maximum Ratings section in the chip DataSheet.

22.1.2 Chip-specific NMI Configuration Register (NCR)

The NMI Configuration Register (NCR) does not support the NDSS0 and NWRE0 fields.

Although these fields support read/write access, writing a value other than the reset value is prohibited, and can produce unexpected results.

22.1.3 Interrupts

WKPU allows an external NMI pin to assert non-maskable interrupts in this chip.

- Control the NMI destination interrupt by configuring the NMI Destination Source Select fields of the NMI Configuration register (NCR[NDSSn]).
- Control enabling of rising edge, falling edge, or either edge reaction to the NMI pin for each core independently with the NMI Falling-edge Events Enable (NCR[NFEEn]) and NMI Rising-edge Events Enable (NCR[NREEn]) fields.
- WKPU supports the capturing of a second event per NMI input before the interrupt is cleared, thus reducing the chance of losing an NMI event.
- The NMI supports a status flag and an overrun flag with the NMI Status Flag and NMI Overrun Status Flag fields of the NMI Status Flag register (NSR[NOVFn] and NSR[NOVFn]).

22.1.4 NMI on Standby exit

Entering Standby mode always resets the NMI Configuration register (NCR). Therefore, exiting Standby mode does not generate NMI from WKPU. To generate NMI, software must first initialize the NCR.

NOTE

Only NCR[NLOCK0] is reset by STANDBY mode. The NMI from WKPU will not be generated on STANDBY as software need to disable NMI before standby entry, and NMI pads will be powered off during standby.

22.1.5 Asynchronous wake-up sources for various modes

In this chip, the following modules can provide wake-up in Standby mode:

- WKPU
- RTC

NOTE

This chip only supports the rising-edge detection of RTC wake-up.

22.1.6 Pull control of wake-up pads

This table shows how the PULL value of WKPU pads are controlled by SIUL2 and WKPU configuration.

Device Mode	SIUL2_MSCRn[PUE]	SIUL2_MSCRn[PUS]	WKUP_PUS[WKUP_PU_OVERRIDE]	WKPU_WIPER[IPUE]	WKPU_PUS[WKUP_PUS]	Affect on WKPU PAD
All except STANDBY modes	0	x	0	x	x	Highz
	x	x	1	0	x	Highz
	x	x	1	1	0	Pull down
	x	x	1	1	1	Pull up
	1	0	0	x	x	Pull down
	1	1	0	x	x	Pull up

NOTE

Wake-up pads pull controls must be configured via the WKPU and not by the SIUL during STANDBY. The proper procedure for managing the wake-up pads pulls is described in the "Wake-up pin pullup/pulldown control during Run and Standby modes" section of the Power chapter.

22.1.7 Enabling wakeup pads

Before enabling a wakeup using the Wakeup Request Enable Register (WRER) register, enable the input buffer (IBE) for the corresponding wakeup pin using SIUL*.MSCR[IBE] control field.

22.1.8 Wakeup/Interrupt Pull Enable Register (WIPUER_WIPDER)

This register is used to enable a Pull on the corresponding interrupt pads to pull an unconnected wakeup/interrupt input to a specified voltage level. The register is described in the WKPU register descriptions in this chapter.

NOTE

Enabling pull-ups on the wake-up pins during the Standby mode will increase the overall standby power consumption.

NOTE

If a wake-up source is not being used, then the software should ensure that the corresponding wake-up interrupt is disabled.

22.2 Introduction

The WKPU supports 32 external sources that can generate interrupts or wakeup events and 1 external source(s) that can cause non-maskable interrupt request(s) or wakeup event(s). In addition, it combines its wakeup events with those generated from other wakeup sources to supply a single wakeup to the system. The block diagram of WKPU and its interfaces to other system components is shown below.

NOTE

The signal widths in the following diagram might not depict this device's particular configuration. See chip-specific WKPU for more information.

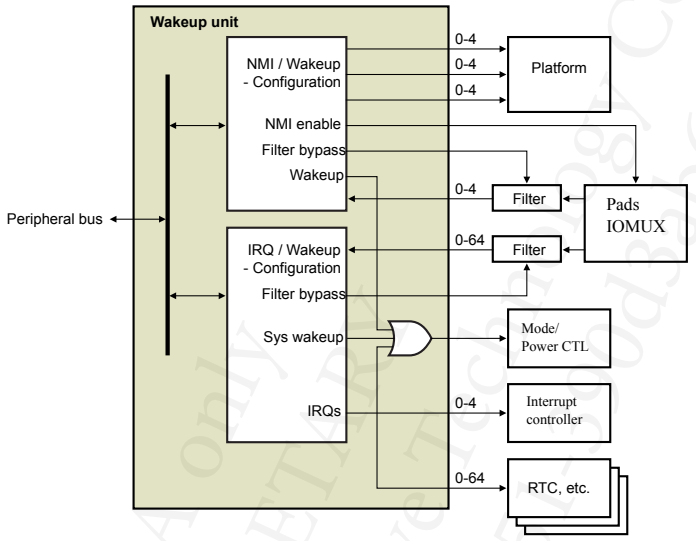


Figure 53. WKPU and connected system components

22.3 Features

The WKPU supports these features:

- Non-maskable interrupt support with:
  - 1 NMI source
  - 1 analog glitch filter
  - Active (rise/fall) edge selection control for events
  - Configurable system wakeup triggering from NMI source
- External wakeup/interrupt support with:
  - System interrupt vector for interrupt sources
  - 31 analog glitch filters
  - Independent interrupt mask
  - Edge detection
  - Configurable system wakeup triggering from all interrupt sources
  - Configurable pullup
- On-chip wakeup support:
  - wakeup sources

Preliminary NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c62e5-4821-48a8-b25...d3624c1

— Wakeup status mapped to same register as external wakeup/interrupt status

22.4 WKPU memory map and registers

22.4.1 WKPU register descriptions

This section provides a detailed description of all the registers accessible in the WKPU module.

NOTE

Reserved registers read as zero and writes have no effect. A transfer error is generated when trying to access a completely reserved register space. The field length of external pad control registers depends on the number of WKPU channels implemented in a chip.

22.4.1.1 WKPU memory map

WKPU base address: 4009_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	NMI Status Flag Register (NSR)	32	W1C	0000_0000h
8h	NMI Configuration Register (NCR)	32	RW	0000_0000h
10h	Wakeup Boot Mode Select Register (WBMSR)	32	RW	0000_0000h
14h	Wakeup/Interrupt Status Flag Register (WISR)	32	W1C	0000_0000h
18h	Interrupt Request Enable Register (IRER)	32	RW	0000_0000h
1Ch	Wakeup Request Enable Register (WRER)	32	RW	0000_0000h
28h	Wakeup/Interrupt Rising-Edge Event Enable Register (WIREER)	32	RW	0000_0000h
2Ch	Wakeup/Interrupt Falling-Edge Event Enable Register (WIFEER)	32	RW	0000_0000h
30h	Wakeup/Interrupt Filter Enable Register (WIFER)	32	RW	0000_0000h
34h	Wakeup/Interrupt Pull Enable Register (WIPUER_WIPDER)	32	RW	0000_0000h

22.4.1.2 NMI Status Flag Register (NSR)

Offset

Register	Offset
NSR	0h

Function

This register holds the non-maskable interrupt status flags.

NOTE

This register is accessible by 8-, 16-, and 32-bit read/write operations.

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NIF0	NOVF0	Reserved						Reserved		Reserved					
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		Reserved						Reserved		Reserved					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 NIF0	NMI Status Flag 0 This flag can be cleared only by writing 1. Writing 0 has no effect. If enabled (NREE0 or NFEE0 set), NIF0 causes an interrupt request. 0b - No event has occurred on the pad 1b - An event as defined by NREE0 and NFEE0 has occurred
30 NOVF0	NMI Overrun Status Flag 0 This flag can be cleared only by writing 1. Writing 0 has no effect. It will be a copy of the current NIF0 value whenever an NMI event occurs, thereby indicating to the software that the NMI occurred when the last one was not yet serviced. If enabled (NREE0 or NFEE0 set), NOVF0 causes an interrupt request. 0b - No overrun has occurred on NMI input 0 1b - An overrun has occurred on NMI input 0
29-24 —	Reserved
23-22 —	Reserved
21-16 —	Reserved
15-14 —	Reserved
13-8 —	Reserved
7-6	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5-0 —	Reserved

22.4.1.3 NMI Configuration Register (NCR)

Offset

Register	Offset
NCR	8h

Function

This register holds the configuration bits for the non-maskable interrupt settings.

NOTE

- This register is accessible by 8-, 16-, and 32-bit read/write operations.
- Writing 0 to both NREE[*n*] and NFEE[*n*] disables the NMI functionality completely (that is, no non-maskable interrupt is generated on any pad activity).

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NLOCK0	NDSS0		NWRE0	0	NREE0	NFEE0	NFE0	Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 NLOCK0	NMI Configuration Lock Register 0 Writing 1 to this bit locks the configuration for the NMI until it is unlocked by a system reset or STANDBY0 mode exit. Writing 0 has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30-29 NDSS0	NMI Destination Source Select 0 **NOTE** As wakeup does not support another interrupt than NMI, the destination source select signal bits are reserved and always retain their reset value. This means no other request other than NMI can be generated. 00b - Non-maskable interrupt 01b - Reserved 10b - Reserved 11b - Reserved
28 NWRE0	NMI Wakeup Request Enable 0 0b - System wakeup requests from the corresponding NIF0 field are disabled 1b - Causes a system wakeup request when NIF0 = 1 or NOVF0 = 1
27 —	Reserved
26 NREE0	NMI Rising-Edge Events Enable 0 0b - Rising-edge event is disabled 1b - Rising-edge event is enabled
25 NFEE0	NMI Falling-edge Events Enable 0 0b - Falling-edge event is disabled 1b - Falling-edge event is enabled
24 NFE0	NMI Filter Enable 0 Enable analog glitch filter on the NMI pad input. 0b - Filter is disabled 1b - Filter is enabled
23-16 —	Reserved
15-8 —	Reserved
7-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
8c4625-4821-48a8-b251-390fab624c1

22.4.1.4 Wakeup Boot Mode Select Register (WBMSR)

Offset

Register	Offset
WBMSR	10h

Function

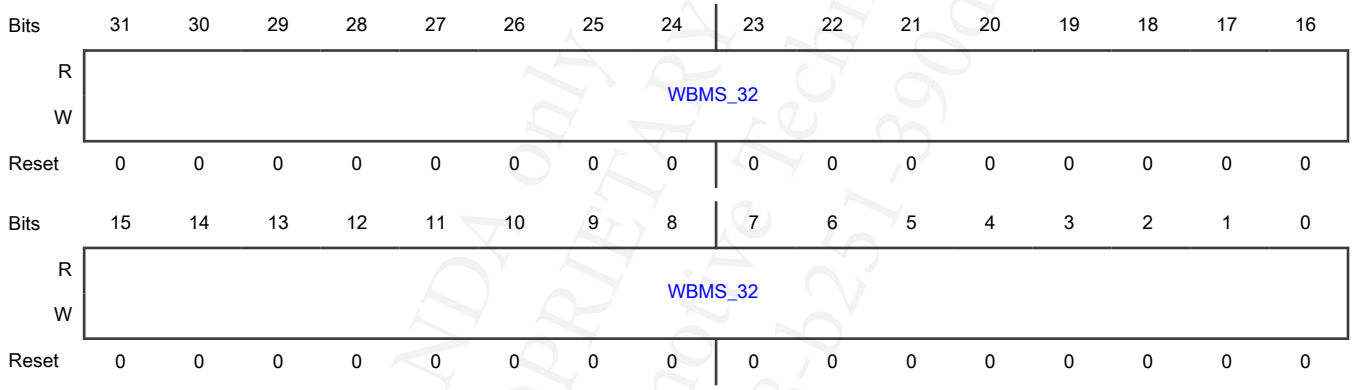
This register is used by the boot software to select between a short boot or a full boot. Application software should write '0' or '1' for each wakeup event in this register. You must ensure that the selected Boot mode matches the corresponding wakeup source.

NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Diagram



Fields

Field	Function
31-0 WBMS_32	Wakeup Short or Long Boot Select 0b - Short boot execution by boot software after wakeup from STANDBY mode 1b - Long boot execution by boot software after wakeup from STANDBY mode

22.4.1.5 Wakeup/Interrupt Status Flag Register (WISR)

Offset

Register	Offset
WISR	14h

Function

This register holds the wakeup/interrupt flags.

NOTE

- This register is accessible only by 32-bit read/write operations.
- Status bits associated with on-chip wakeup sources are located at the left of the external wakeup/interrupt status bits and are read-only. The wakeup for these sources must be configured and cleared at the on-chip wakeup source.

Access: User read/write

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EIF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EIF															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EIF	External Wakeup/Interrupt Status Flag x This flag can be cleared only by writing 1. Writing 0 has no effect. If enabled (IRER[x]), EIF[x] causes an interrupt request. 0b - No event has occurred on the pad 1b - An event as defined by WIREER and WIFEER has occurred

22.4.1.6 Interrupt Request Enable Register (IRER)

Offset

Register	Offset
IRER	18h

Function

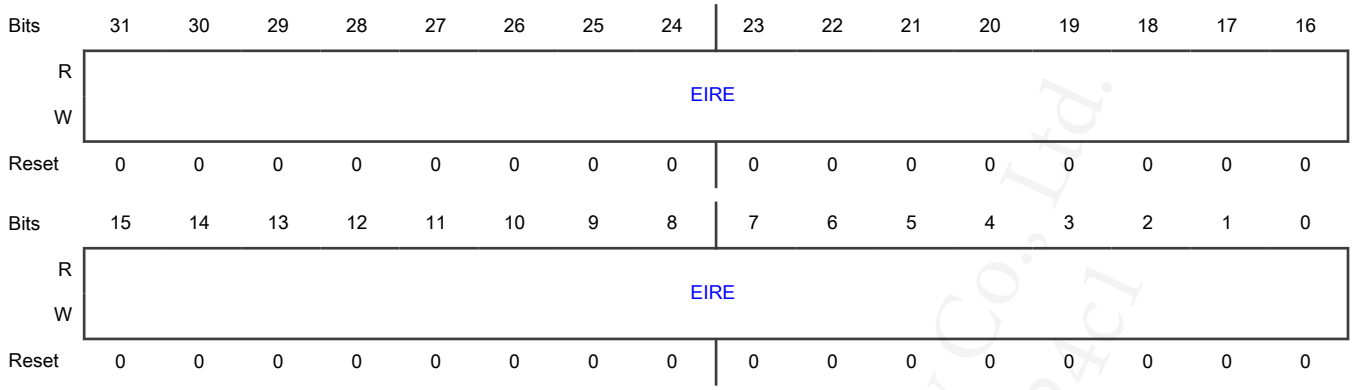
This register is used to enable the interrupt messaging from the wakeup/interrupt pads to the interrupt controller.

NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Diagram



Fields

Field	Function
31-0 EIRE	External Interrupt Request Enable x 0b - Interrupt requests from the corresponding EIF[x] bit are disabled 1b - A set EIF[x] bit causes an interrupt request

22.4.1.7 Wakeup Request Enable Register (WRER)

Offset

Register	Offset
WRER	1Ch

Function

This register is used to enable the system wakeup messaging from the wakeup/interrupt pads to the mode entry and power control modules.

NOTE

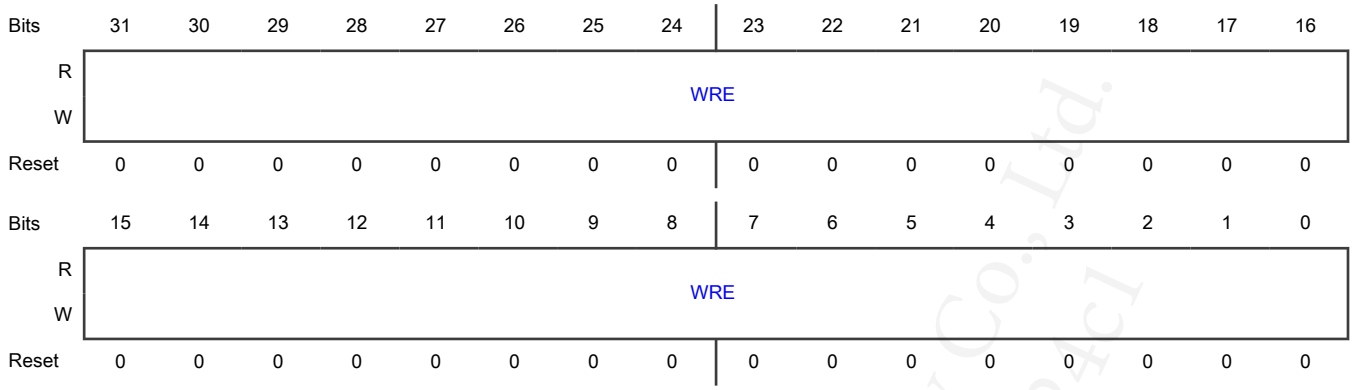
This register is accessible only by 32-bit read/write operations.

If a pin is disabled through this register, the corresponding bits in the WIFEER and WIREER registers must be written to 0 to ensure that the pin does not respond to any change.

Access: User read/write

Provided under NDA only
COMPANY PROPRIETARY
Beijing CTA Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 WRE	External Wakeup Request Enable x 0b - System wakeup requests from corresponding EIF[x] bit are disabled 1b - A set EIF[x] bit causes a system wakeup request

22.4.1.8 Wakeup/Interrupt Rising-Edge Event Enable Register (WIREER)

Offset

Register	Offset
WIREER	28h

Function

This register is used to enable rising-edge triggered events on the corresponding wakeup/interrupt pads.

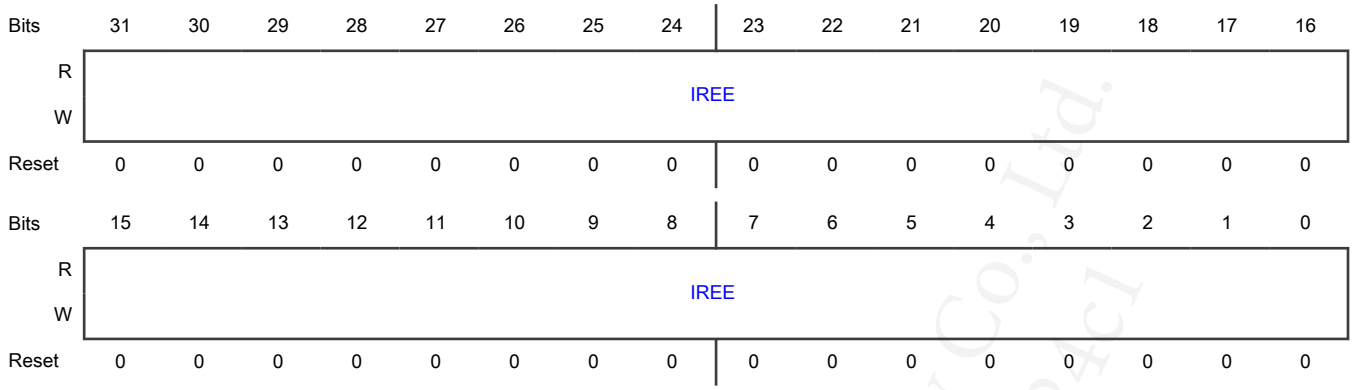
NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-482a-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 IREE	External Interrupt Rising-edge Events Enable x 0b - Rising-edge event is disabled 1b - Rising-edge event is enabled

22.4.1.9 Wakeup/Interrupt Falling-Edge Event Enable Register (WIFEER)

Offset

Register	Offset
WIFEER	2Ch

Function

This register is used to enable falling-edge triggered events on the corresponding wakeup/interrupt pads.

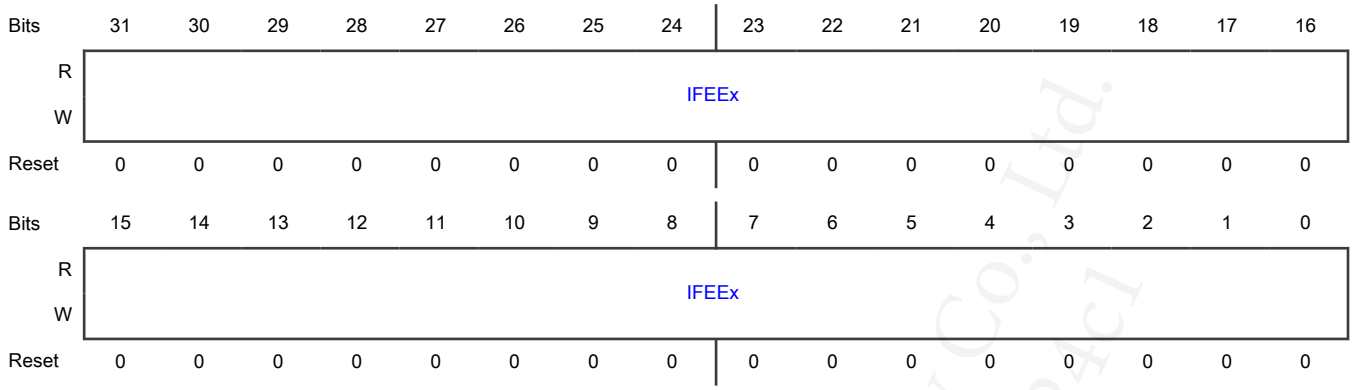
NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482a-8a8-b251390d3ab624c1

Diagram



Fields

Field	Function
31-0 IFEEx	External Interrupt Falling-edge Events Enable x 0b - Falling-edge event is disabled 1b - Falling-edge event is enabled

22.4.1.10 Wakeup/Interrupt Filter Enable Register (WIFER)

Offset

Register	Offset
WIFER	30h

Function

This register is used to enable an analog filter on the corresponding interrupt pads to filter out glitches on the inputs.

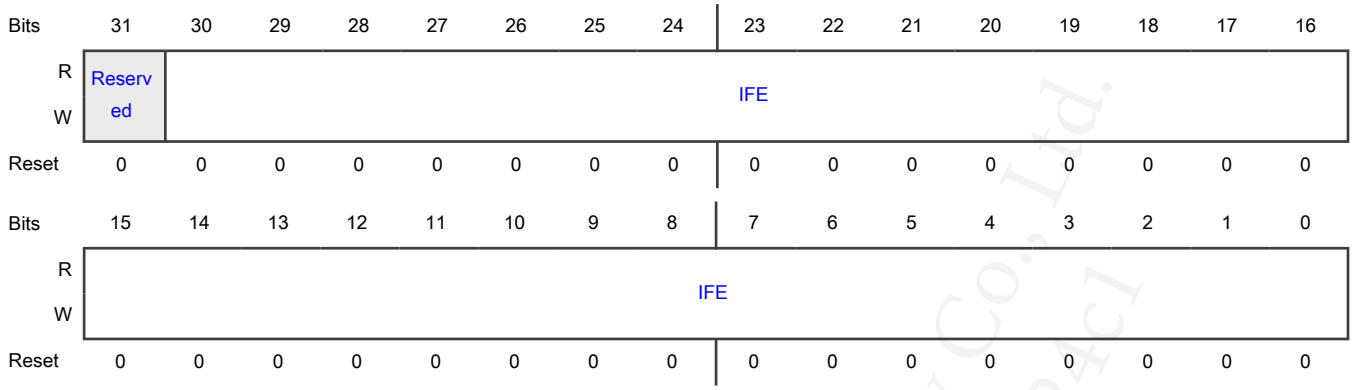
NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482...8a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 —	Reserved
30-0 IFE	External Interrupt Filter Enable x Enable analog glitch filter on the external interrupt pad input. 0b - Filter is disabled 1b - Filter is enabled

22.4.1.11 Wakeup/Interrupt Pull Enable Register (WIPUER_WIPDER)

Offset

Register	Offset
WIPUER_WIPDER	34h

Function

This register is used to enable a Pull on the corresponding interrupt pads to pull an unconnected wakeup/interrupt input to a specified voltage level (See the chip-specific WKPU information for the exact level of voltage). When enabling a PULL, you must ensure that the wakeup or interrupt is disabled to prevent any unintended events.

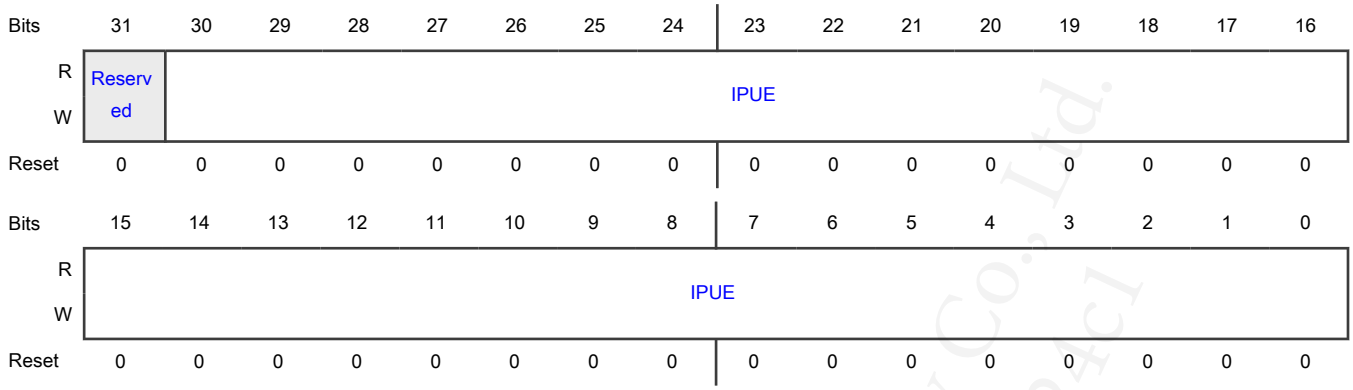
NOTE

This register is accessible only by 32-bit read/write operations.

Access: User read/write

Provided under NDA only
COMPANY PROPRIETARY
Beijing CUU Automotive Technology Co.,Ltd.
c8c46225-8321-48a8-9251-390d3ab624c1

Diagram



Fields

Field	Function
31 —	Reserved
30-0 IPUE	External Interrupt Pull Enable x 00000000000000000000000000000000b - Pull is disabled. 00000000000000000000000000000001b - Pull is enabled.

22.5 Functional description

This section provides functional description of WKPU.

22.5.1 Non-maskable interrupts

The WKPU supports the capturing of a second event per NMI input before the interrupt is cleared, thus reducing the chance of losing an NMI event, although it creates an overrun condition.

Each NMI passes through a bypassable analog glitch filter.

NOTE

Glitch filter control and pad configuration should be performed when NMI is disabled, to avoid erroneous triggering by glitches caused by the configuration process itself.

NOTE

The figure below represents a generic configuration and might not represent this particular device's configuration. See the chip-specific information for details on this chip's WKPU.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHA Automotive Technology Co., Ltd.
c8c462e3-4824-4888-b251-390d3ab624c1

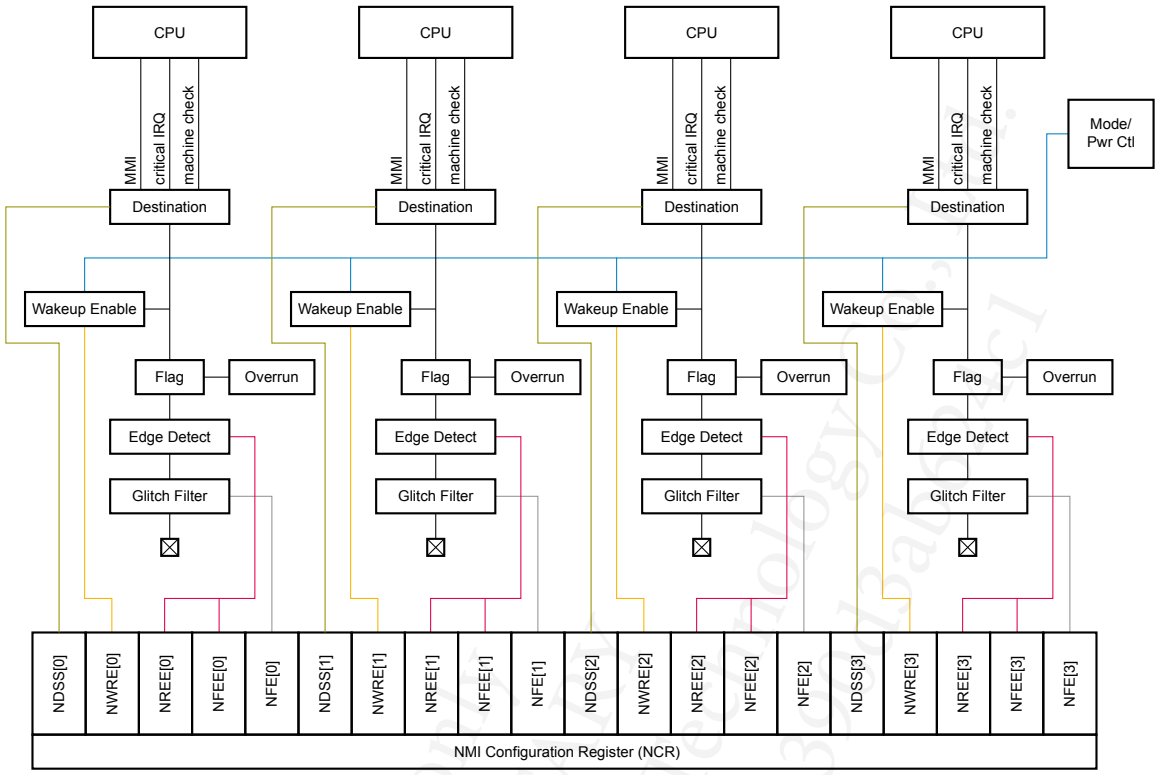


Figure 54. NMI pad diagram

22.5.1.1 NMI management

Each NMI can be enabled or disabled independently. This can be performed using the single NCR register laid out to contain all configuration bits for a given NMI in a single byte (see NMI Configuration Register (NCR)). A pad defined as an NMI can be configured by the user to recognize interrupts with an active rising edge, an active falling edge, or both edges being active. A setting of having both edge events disabled results in no interrupt being detected and should not be configured.

The active NMI edge is controlled by the user through the configuration of the NCR[NREEn] and NCR[NFEEn] bits.

NOTE

After reset, NREE and NFEE are set to '0'; therefore, the NMI functionality is disabled after reset and must be enabled explicitly by software.

After a pad's NMI functionality is enabled, the pad cannot be reconfigured to override or disable the NMI. See the chip-specific WKPU information for details of NMI implementation.

The NMI destination interrupt is controlled by the user through the configuration of the NCR[NDSSn] bits. See NMI Configuration Register (NCR) for details.

Each NMI supports a status flag and an overrun flag, which are located in the NSR register (see NMI Status Flag Register (NSR)). This register is a clear-by-write-1 register type, preventing inadvertent overwriting of other flags in the same register. The status flag is set whenever an NMI event is detected. The overrun flag is set whenever an NMI event is detected and the status flag is set (that is, status flag has not been cleared).

NOTE

The overrun flag is cleared by writing a '1' to the appropriate overrun bit in the NSR register. If the status bit is cleared and the overrun bit is still set, the pending interrupt is not cleared.

During an NMI ISR, on wakeup of the chip from an NMI, any writes to the ECC-protected memory must have the correct ECC.

22.6 External wakeups/interrupts

The WKPU supports 1 interrupt vector to the interrupt controller of the chip. Each interrupt vector can support up to the number of external interrupt sources from the device pads, with the total across all vectors being equal to the number of external interrupt sources. Each external interrupt source is assigned to exactly one interrupt vector. The interrupt vector assignment is sequential so that one interrupt vector is for external interrupt sources 0 through N-1, the next is for N through N+M-1, and so forth.

See the following figure for an overview of the external interrupt implementation, showing an example of four interrupt vectors with eight external interrupt sources each.

NOTE

The figure below shows a generic representation of WKPU. See the chip-specific WKPU information for details on how this device's configuration might differ from this figure.

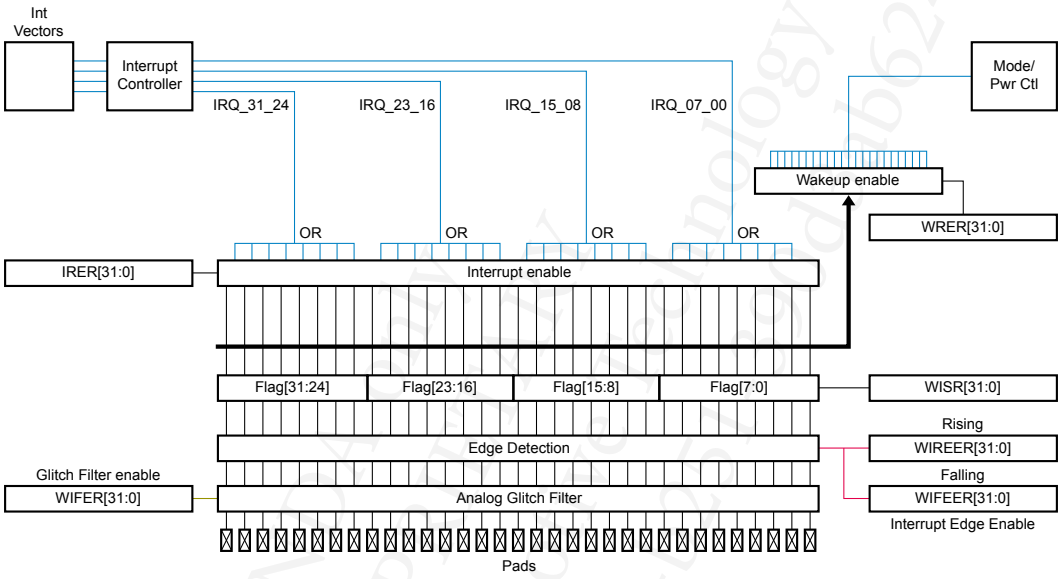


Figure 55. External interrupt pad diagram

All of the external interrupt pads within a single group have equal priority. It is the responsibility of the user software to search through the group of sources in the most appropriate way for their application.

The priority of the vectors used by the external interrupt pads is set based on the platform and the interrupt controller's priority levels, but the allocation of pads to each group of interrupts can be independently configured by the chip.

External interrupt lines have a digital glitch filter applied to them.

22.6.1 External interrupt management

Each interrupt can be enabled or disabled independently. This can be performed using a single rolled-up register (Interrupt Request Enable Register (IRER)). A pad defined as an external interrupt can be configured by the user to recognize interrupts with an active rising edge, an active falling edge, or both edges being active.

NOTE

Writing a '0' to both IREE[x] and IFEE[x] disables the external interrupt functionality for that pad completely (that is, no system wakeup or interrupt is generated on any activity of that pad).

The active IRQ edge is controlled by users through the configuration of the registers WIREER and WIFEER.

Each external interrupt supports an individual flag, which is held in the flag register, WISR. This register is a clear-by-write-1 register type, preventing inadvertent overwriting of other flags in the same register.

22.7 On-chip wakeups

The WKPU supports on-chip wakeup sources. WKPU and on-chip wakeup sources interrupt simultaneously if wakeup from on-chip wakeup sources and WKPU (that is, Wakeup Request Enable Register (WRER)) is configured in Stop and Low-Power modes. Alternatively, in Standby mode a single wakeup is generated to the system through WKPU.

22.8 Initialization information

This section discusses initialization for the following features:

- Glitch filter and pad configuration
- Non-maskable interrupts

22.8.1 Glitch filter and pad configuration

Glitch filter control and pad configuration should be performed when the NMI is disabled. This is to avoid erroneous triggering by glitches caused by the configuration process.

When enabling the glitch filter, do not enable the rising-/falling-edge events bits, i.e., the NCR[NREE], NCR[NFEE], NCR[RREE], and NCR[RFEE] bits in the same register write.

22.8.2 Non-maskable interrupts

If the IBE of NMI is controlled by the NFEE or NREE fields of NMI Configuration Register (NCR) when an NMI interrupt pin is first enabled, it is possible to get a false interrupt flag. You must follow the following sequence to enable the NMI interrupt on a rising edge event on the NMI pin:

1. Write 1 to NCR[NFEE0]
2. Write 1 to NCR[NREE0]
3. Write 0 to NCR[NFEE0]

If the IBE of NMI is tied off to 1, no false interrupt is expected.

22.9 Glossary

NMI	Non-maskable interrupts

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-350d3a6024c1

Chapter 23
Clocking

23.1 Introduction

This chapter describes the architecture of the system-level clocks and includes the following:

- System clock specifics
- Clock sources
- Clock architecture
- Clock control registers
- Clock monitoring
- Clock gating
- Module clocking

The chapter covers information related to clocks generated on the chip. Protocol clocks provided by external chips are covered in the respective chapters where they are used.

23.1.1 Features

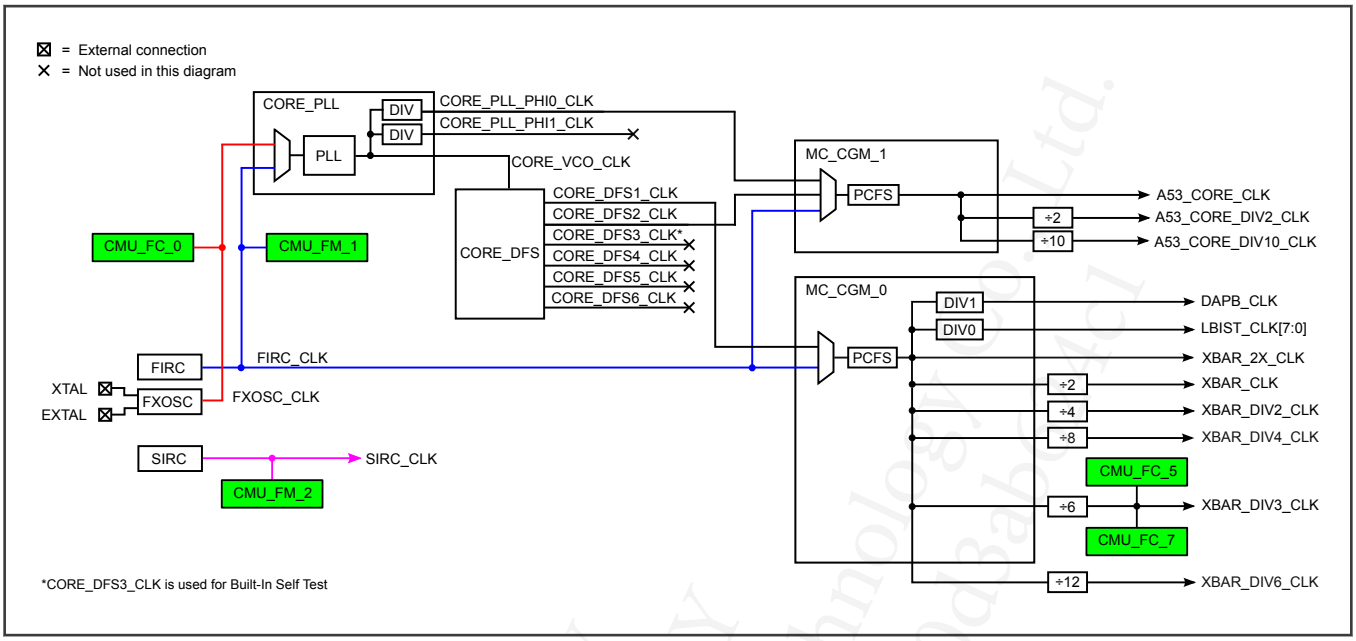
The chip-clocking architecture supports the following features:

- Multiple clock sources supported for clock generation:
  - Fast Internal RC Oscillator (FIRC)
  - Slow Internal RC Oscillator (SIRC)
  - Fast External Crystal Oscillator (FXOSC)
  - Phase-Locked Loops (PLLs)
  - Digital Frequency Synthesizer (DFS) modules
- Frequency Modulated (FM) clock to reduce electromagnetic emissions
- Precise clocks for timers and communication functions
- Glitchless clock switching of Clock Generation Module (MC_CGM) clock selectors
- Progressive Clock Frequency Switching (PCFS) for system clocks driving cores
- Clock monitor units to check the integrity of the clocks
- Module clock gating using the MC_ME Partition *n* Process Configuration registers

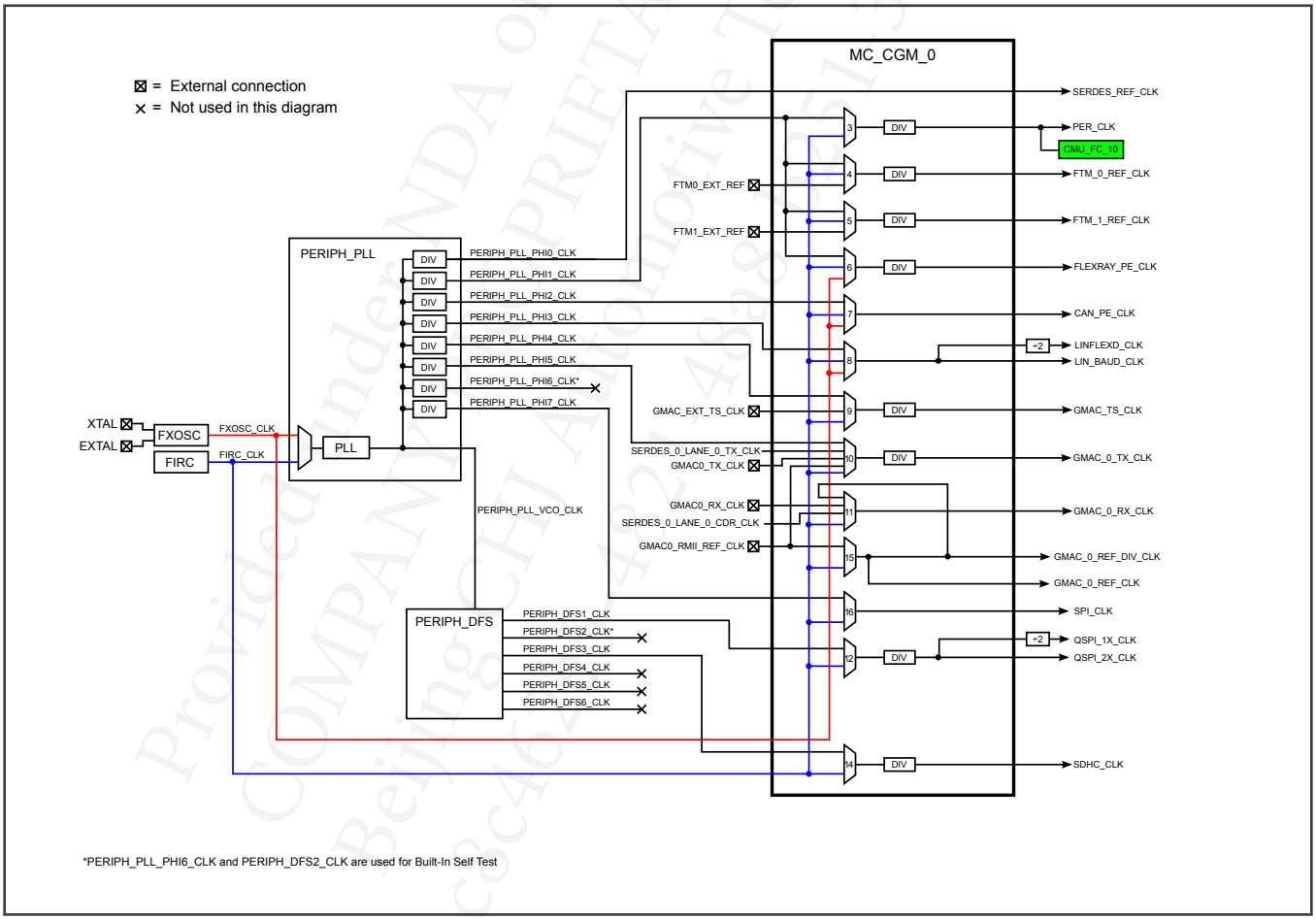
23.1.2 Overview

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4a21-48a8-b251-390d3ab624c1

23.1.2.1 Core-related clock overview

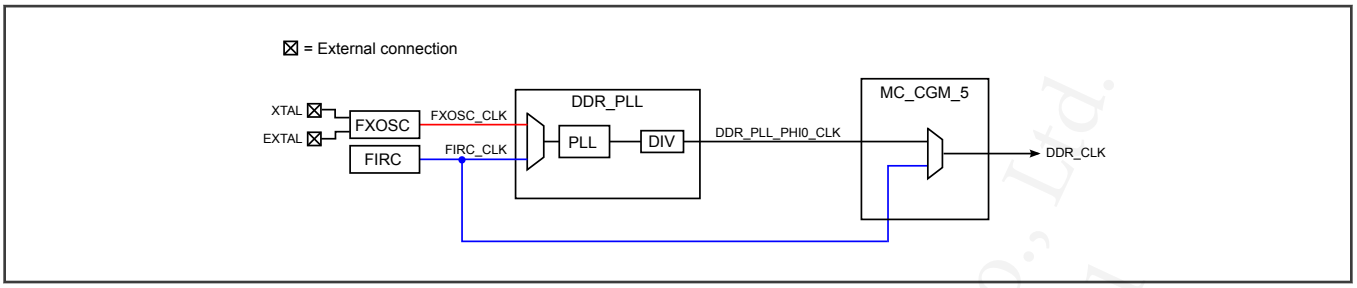


23.1.2.2 Peripheral-related clock overview

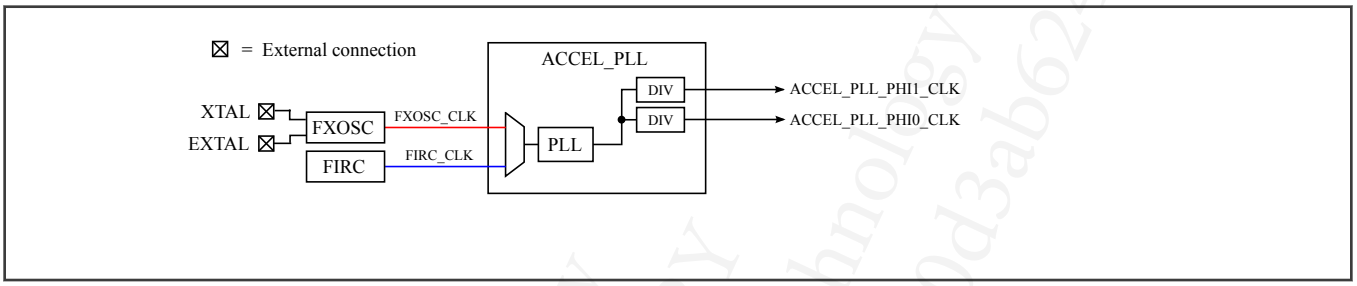


Provided under NDA only
COMPANY PRIFTARY
Beijing CHI Automotive Technology Co.,Ltd.
CCA6243 Sunday 2021/8/29 15:39:01

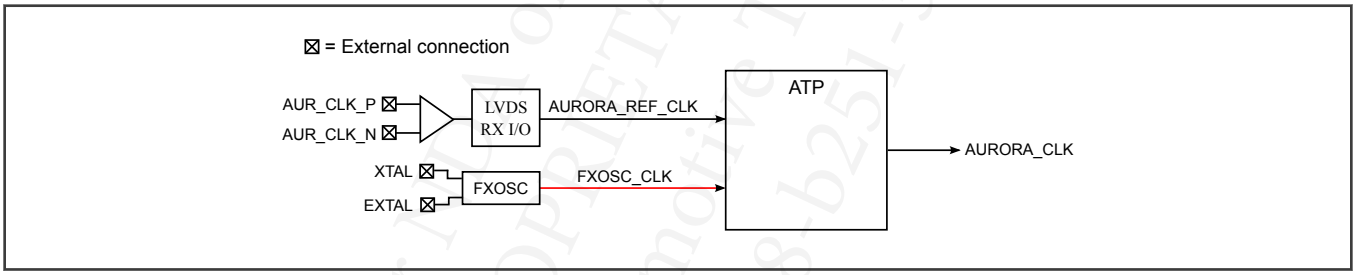
23.1.2.3 DDR PLL clock overview



23.1.2.4 Accelerator PLL clock overview



23.1.2.5 Aurora clocking overview

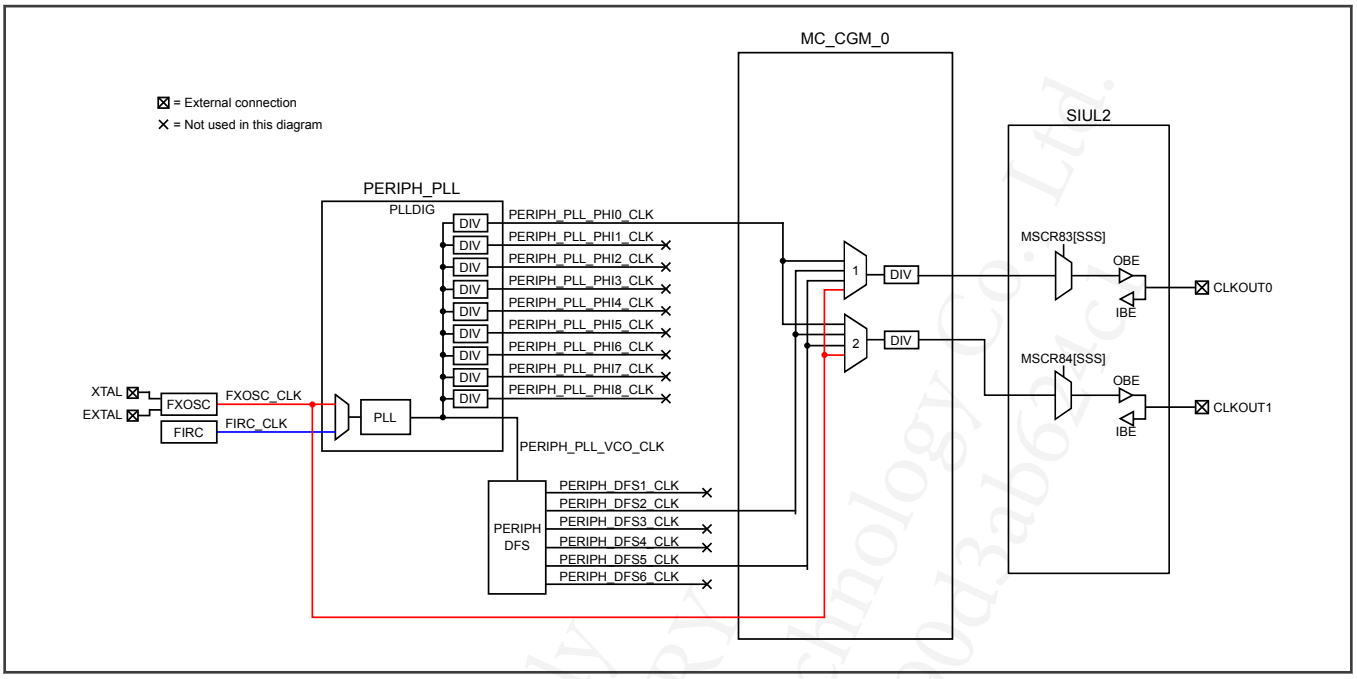


NOTE

See section "Aurora PLL" in chapter "Aurora Trace Port (ATP) Subsystem" for clocking connection details on the ATP block and AURORA_PLL.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a8-b251-390d3ab624c1

23.1.2.6 CLKOUT*n* overview



IMPORTANT

CLKOUT*n* output should be disabled prior to disabling the CLKOUT*n* source.

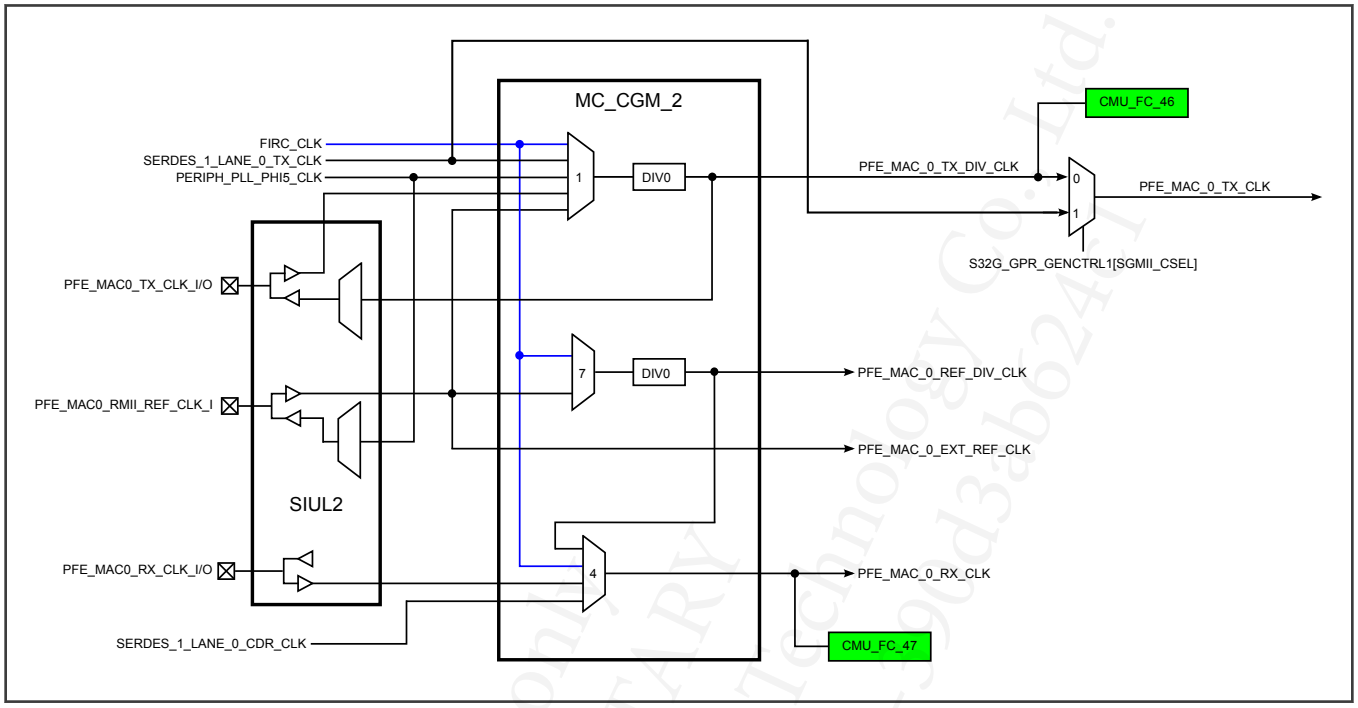
23.1.2.6.1 CLKOUT*n* clock configuration

Destination clock	Source clock	I/O port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
CLKOUT0	See row "MC_CGM_0" and "Clock mux 1 selector" in Application clock source options	PF_03	83	1	0	001b	—	—
CLKOUT1	See row "Clock mux 2 selector" in Application clock source options	PF_04	84	1	0	001b	—	—

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482f-48a8-b251-390d3ab6f4c

23.1.2.7 Other clocks

23.1.2.7.1 PFE_MAC_0 clocking overview

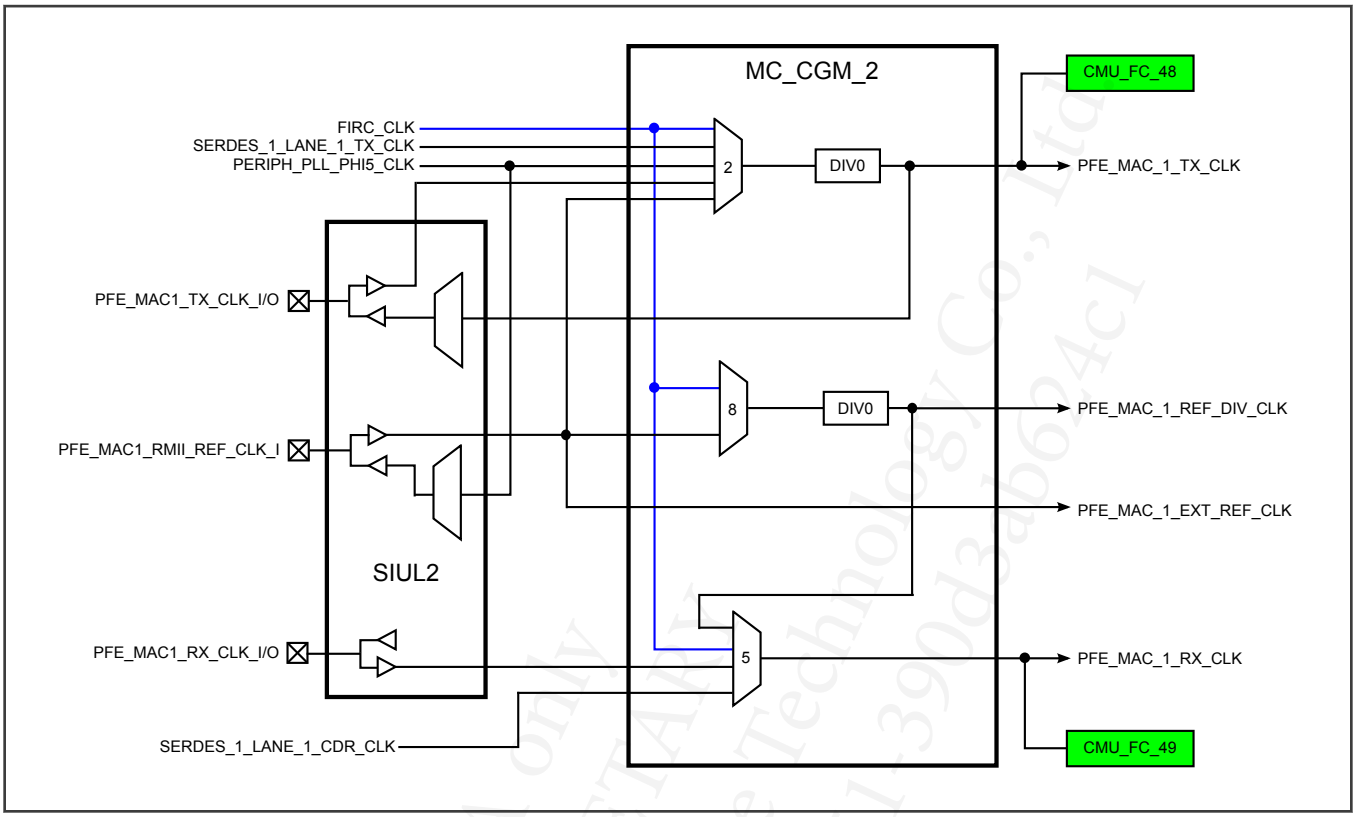


NOTE

Before switching clock source for PFE_MAC_0_TX_CLK, both mux clock inputs must be disabled (for example, PFE_MAC_0_TX_DIV_CLK and SERDES_1_LANE_0_TX_CLK).

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-590d3ab624f1

23.1.2.7.2 PFE_MAC_1 clocking overview



23.1.2.7.3 PFE_MAC_2 clocking overview

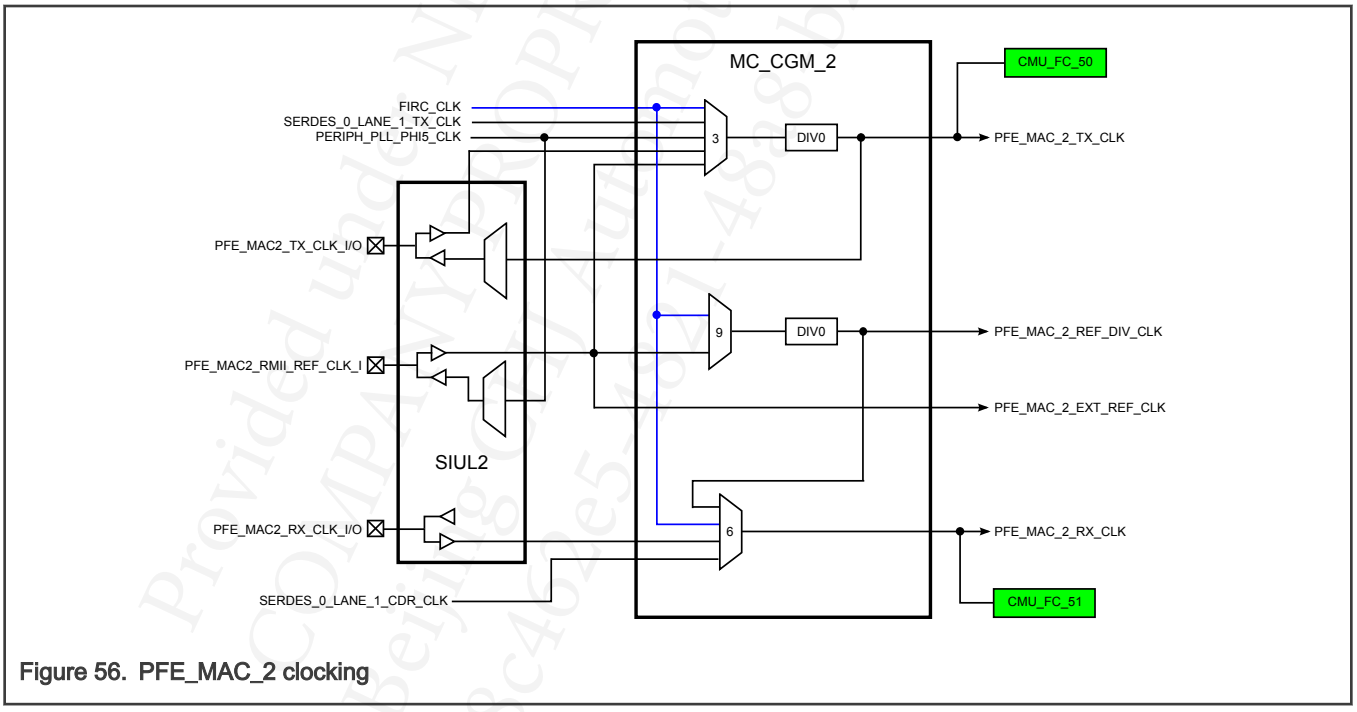
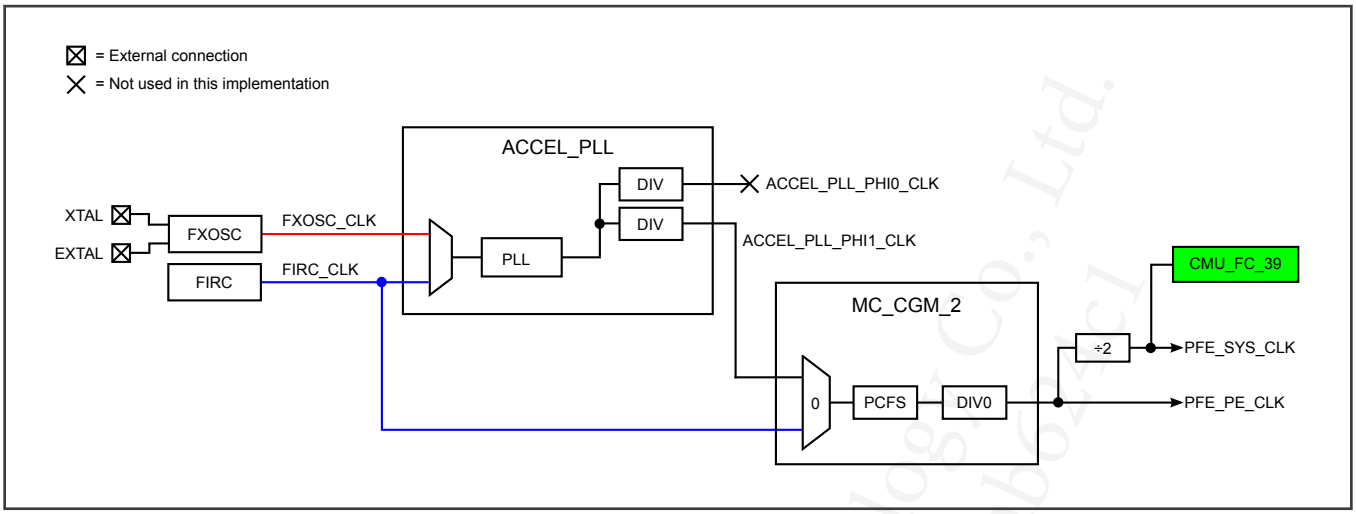


Figure 56. PFE_MAC_2 clocking

Provided under NDA only
COMPANY PROPRIETARY
Beijing Autmotive Technology Co., Ltd
c8c46e5...

23.1.2.7.4 PFE clocking



23.2 Clock sources

23.2.1 Introduction

The internal clocks are generated from one of these input clock sources:

- FIRC — 48 MHz (mainly used as a backup clock) (see note)
- SIRC — 32 kHz (see note)
- FXOSC — 20–40 MHz (see note)
- PLLs:
  - CORE_PLL (FM)
  - PERIPH_PLL (Non-FM)
  - DDR_PLL (FM)
  - ACCEL_PLL (FM)
  - AURORA_PLL (Non-FM)
- DFS blocks:
  - CORE_PLL DFS
  - PERIPH_PLL DFS

NOTE

Nominal frequencies are shown in the list above. Detailed clock operating parameters can be found in this chip's data sheet.

The PLL outputs (direct output or DFS outputs), FXOSC, SIRC, and FIRC can be selected to drive system peripherals, cores, and infrastructure depending on MC_CGM configuration.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46254-4821-48a8-b251-390d3ab62cc1

23.2.2 Input clock pins

Description	Pin
FXOSC pins	XTAL
	EXTAL
FTM_0 external reference clock	FTM0_EXT_REF
FTM_1 external reference clock	FTM1_EXT_REF
SPI_*n* clock	DSPI*n*_SCK_I
Ethernet timestamp clock	GMAC_TS_CLK_I
Ethernet TX clock	GMAC0_TX_CLK_I
Ethernet RX clock	GMAC0_RX_CLK_I
Ethernet RMII REF clock	GMAC0_RMII_REF_CLK_I
Aurora reference clock (differential)	AUR_CLK_P, AUR_CLK_N
JTAG clock	TCK
RTC external clock	RTC_EXT_CLK
PFE_MAC_0 Ethernet TX clock	PFE_MAC0_TX_CLK_I
PFE_MAC_0 Ethernet RX clock	PFE_MAC0_RX_CLK_I
PFE_MAC_0 Ethernet RMII REF clock	PFE_MAC0_RMII_REF_CLK_I
PFE_MAC_1 Ethernet TX clock	PFE_MAC1_TX_CLK_I
PFE_MAC_1 Ethernet RX clock	PFE_MAC1_RX_CLK_I
PFE_MAC_1 Ethernet RMII REF clock	PFE_MAC1_RMII_REF_CLK_I
PFE_MAC_2 Ethernet TX clock	PFE_MAC2_TX_CLK_I
PFE_MAC_2 Ethernet RX clock	PFE_MAC2_RX_CLK_I
PFE_MAC_2 Ethernet RMII REF clock	PFE_MAC2_RMII_REF_CLK_I

23.2.3 Output clock pins

Description	Pin
Output clock	CLKOUT0
Output clock	CLKOUT1

23.2.4 FIRC

This chip has a 48 MHz FIRC (output FIRC_CLK) with the following features:

- Always enabled
- PLL clock source when FXOSC is not available
- Clock source for the following modules:
  - Power Management Controller (PMC)

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-2

- Reset Generation Module (MC_RGM)
- Fault Collection and Control Unit (FCCU)
- FCCU Output Supervision Unit (FOSU)
- Software Watchdog Timer (SWT)
- System Integration Unit Lite2 (SIUL2) (input filter)

- Default system clock after reset
- Serves as the system clock source in case loss of lock or loss of clock detection
- Turned off while entering the Standby mode

23.2.5 SIRC

This chip has a 32 kHz SIRC (output SIRC_CLK) with the following features:

- Always enabled
- Used as a clock source for the following modules:
  - Real-time clock (RTC)
  - POR_WDOG

23.2.6 FXOSC

This chip has a 20—40 MHz FXOSC (output FXOSC_CLK) with the following features:

- Reference for the chip PLLs
- Reference clock source for the following:
  - FlexCAN
  - FlexRay
- Available to drive the CLKOUT pin
- Allows a crystal or external clock source reference clock for the chip
- Uses crystals with a fundamental frequency range of 20–40 MHz
- Supports the following input clock modes:
  - Crystal
  - Single-ended bypass
  - Differential bypass

23.2.6.1 FXOSC register interface

Please see the "FXOSC" chapter for details.

23.2.7 PLL

This chip contains the following PLLs:

- CORE_PLL - Programmable FM PLL (SSCG) used for the following:
  - Cortex-A53 clusters
  - Cortex-M7 clusters
  - HSE_H
  - High-speed chip interconnect

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

- PERIPH_PLL - For peripherals including the following:
  - GMAC
  - FlexCAN
  - FlexRay (non-FM)
  - ...
- ACCEL_PLL - Programmable accelerators supporting FM (SSCG)
- DDR_PLL - For DRAM interface (FM)
- AURORA_PLL - For Aurora interface (non-FM)

These PLLs have the following features:

- One or all of the following modes of operation:
  - Normal PLL mode with crystal reference
  - Normal PLL mode with external reference
  - Normal PLL mode with internal RC oscillator (FIRC) input (used during boot)
- Lock status indication from the monitoring circuitry
- Option to turn on/off frequency modulation (applies to CORE_PLL, DDR_PLL, ACCEL_PLL)
- Fractionally divided outputs using Digital Frequency Synthesizer (DFS) for additional clock domains

23.2.7.1 PLL input clocks

Clock sources for the PLLs can be either the 20-40 MHz FXOSC or 48 MHz FIRC. During boot, FIRC_CLK is used as the default PLL reference clock. After boot, the PLL reference must be changed to FXOSC_CLK.

23.2.7.2 PLL output clocks

PLLs have VCO and multiple PHI*n* output clocks. PLL output clocks can be configured to be divided for use by modules (see Overview for details).

23.2.7.3 PLL register interface

See chapter "PLL digital interface (PLLDIG)" for details.

23.2.7.4 DFS

The chip includes two DFS modules:

- CORE_PLL DFS
- PERIPH_PLL DFS

DFS uses the VCO output clock of the PLLs as its source clock.

DFS modules provide the following features:

- Multiple-phase divider outputs
- Phase divider outputs that are controlled independently
- Phase dividers that have independent resets
- CORE_PLL DFS that supports six outputs
- PERIPH_PLL DFS that supports six outputs

The DFS clocks are mapped to the DFS ports as follows:

Table 69. DFS clock and port mapping

DFS instance	Clock	Port
CORE_DFS	CORE_DFS1_CLK	CORE_DFS_DVPORT0
	CORE_DFS2_CLK	CORE_DFS_DVPORT1
	CORE_DFS3_CLK	CORE_DFS_DVPORT2
	CORE_DFS4_CLK	CORE_DFS_DVPORT3
	CORE_DFS5_CLK	CORE_DFS_DVPORT4
	CORE_DFS6_CLK	CORE_DFS_DVPORT5
PERIPH_DFS	PERIPH_DFS1_CLK	PERIPH_DFS_DVPORT0
	PERIPH_DFS2_CLK	PERIPH_DFS_DVPORT1
	PERIPH_DFS3_CLK	PERIPH_DFS_DVPORT2
	PERIPH_DFS4_CLK	PERIPH_DFS_DVPORT3
	PERIPH_DFS5_CLK	PERIPH_DFS_DVPORT4
	PERIPH_DFS6_CLK	PERIPH_DFS_DVPORT5

23.3 Clock generation

23.3.1 Clock source mapping on MC_CGM clock selectors

Clock selector index[1]	Clock source
0	FIRC_CLK
1	SIRC_CLK
2	FXOSC_CLK
4	CORE_PLL_PHI0_CLK
5	CORE_PLL_PHI1_CLK
12	CORE_DFS1_CLK
13	CORE_DFS2_CLK
14	CORE_DFS3_CLK
15	CORE_DFS4_CLK
16	CORE_DFS5_CLK
17	CORE_DFS6_CLK
18	PERIPH_PLL_PHI0_CLK
19	PERIPH_PLL_PHI1_CLK
20	PERIPH_PLL_PHI2_CLK
21	PERIPH_PLL_PHI3_CLK

Table continues on the next page...

Table continued from the previous page...

Clock selector index[1]	Clock source
22	PERIPH_PLL_PHI4_CLK
23	PERIPH_PLL_PHI5_CLK
24	PERIPH_PLL_PHI6_CLK
25	PERIPH_PLL_PHI7_CLK
26	PERIPH_DFS1_CLK
27	PERIPH_DFS2_CLK
28	PERIPH_DFS3_CLK
29	PERIPH_DFS4_CLK
30	PERIPH_DFS5_CLK
31	PERIPH_DFS6_CLK
32	ACCEL_PLL_PHI0_CLK
33	ACCEL_PLL_PHI1_CLK
34	FTM_0_EXT_REF_CLK
35	FTM_1_EXT_REF_CLK
36	DDR_PLL PHI0
37	GMAC_0_EXT_TX_CLK
38	GMAC_0_EXT_RX_CLK
39	GMAC_0_EXT_REF_CLK
40	SERDES_0_LANE_0_TX_CLK
41	SERDES_0_LANE_0_CDR_CLK
44	GMAC_EXT_TS_CLK
45	GMAC_0_REF_DIV_CLK
46	SERDES_0_LANE_1_TX_CLK (MC_CGM_2 only)
47	SERDES_0_LANE_1_CDR_CLK (MC_CGM_2 only)
48	PFE_MAC_0_EXT_TX_CLK
49	PFE_MAC_0_EXT_RX_CLK
50	PFE_MAC_0_EXT_REF_CLK
51	PFE_MAC_1_EXT_TX_CLK
52	PFE_MAC_1_EXT_RX_CLK
53	PFE_MAC_1_EXT_REF_CLK
54	PFE_MAC_2_EXT_TX_CLK
55	PFE_MAC_2_EXT_RX_CLK

Table continues on the next page...

Table continued from the previous page...

Clock selector index[1]	Clock source
56	PFE_MAC_2_EXT_REF_CLK
57	SERDES_1_LANE_0_TX_CLK
58	SERDES_1_LANE_0_CDR_CLK
59	PFE_MAC_0_REF_DIV_CLK
60	PFE_MAC_1_REF_DIV_CLK
61	PFE_MAC_2_REF_DIV_CLK
62	SERDES_1_LANE_1_TX_CLK
63	SERDES_1_LANE_1_CDR_CLK

1. All clock selector indexes not shown are reserved.

23.3.2 MC_CGM_*n* registers

The tables in the following sections show relationships between MC_CGM_*n* registers and output clocks.

23.3.2.1 MC_CGM_0 registers relationship to output clocks

Clock mux	Description	Register	Selector output	Divider output
MC_CGM_0				
Clock mux 0	Select Control Register	MC_CGM_MUX_0_CSC	XBAR_2X_CLK	–
	Select Status Register	MC_CGM_MUX_0_CSS		
	Divider 0 Control Register	MC_CGM_MUX_0_DC_0		LBIST_CLK
	Divider 1 Control Register	MC_CGM_MUX_0_DC_1		DAPB_CLK[1]
Clock mux 1	Select Control Register	MC_CGM_MUX_1_CSC	–	–
	Select Status Register	MC_CGM_MUX_1_CSS		
	Divider 0 Control Register	MC_CGM_MUX_1_DC_0		CLKOUT0
Clock mux 2	Select Control Register	MC_CGM_MUX_2_CSC	–	–
	Select Status Register	MC_CGM_MUX_2_CSS		
	Divider 0 Control Register	MC_CGM_MUX_2_DC_0		CLKOUT1

Table continues on the next page...

Table continued from the previous page...

Clock mux	Description	Register	Selector output	Divider output
MC_CGM_0				
Clock mux 3	Select Control Register	MC_CGM_MUX_3_CSC	–	–
	Select Status Register	MC_CGM_MUX_3_CSS		
	Divider 0 Control Register	MC_CGM_MUX_3_DC_0		PER_CLK
Clock mux 4	Select Control Register	MC_CGM_MUX_4_CSC	–	–
	Select Status Register	MC_CGM_MUX_4_CSS		
	Divider 0 Control Register	MC_CGM_MUX_4_DC_0		FTM_0_REF_CLK
Clock mux 5	Select Control Register	MC_CGM_MUX_5_CSC	–	–
	Select Status Register	MC_CGM_MUX_5_CSS		
	Divider 0 Control Register	MC_CGM_MUX_5_DC_0		FTM_1_REF_CLK
Clock mux 6	Select Control Register	MC_CGM_MUX_6_CSC	–	–
	Select Status Register	MC_CGM_MUX_6_CSS		
	Divider 0 Control Register	MC_CGM_MUX_6_DC_0		FLEXRAY_PE_CLK
Clock mux 7	Select Control Register	MC_CGM_MUX_7_CSC	CAN_PE_CLK	–
	Select Status Register	MC_CGM_MUX_7_CSS		
Clock mux 8	Select Control Register	MC_CGM_MUX_8_CSC	LIN_BAUD_CLK	–
	Select Status Register	MC_CGM_MUX_8_CSS		
Clock mux 9	Select Control Register	MC_CGM_MUX_9_CSC	–	–
	Select Status Register	MC_CGM_MUX_9_CSS		
	Divider 0 Control Register	MC_CGM_MUX_9_DC_0		GMAC_TS_CLK
Clock mux 10	Select Control Register	MC_CGM_MUX_10_CSC	–	–
	Select Status Register	MC_CGM_MUX_10_CSS		
	Divider 0 Control Register	MC_CGM_MUX_10_DC_0		GMAC_0_TX_CLK

Table continues on the next page...

Table continued from the previous page...

Clock mux	Description	Register	Selector output	Divider output
MC_CGM_0				
Clock mux 11	Select Control Register	MC_CGM_MUX_11_CSC	GMAC_0_RX_CLK	–
	Select Status Register	MC_CGM_MUX_11_CSS		
Clock mux 12	Select Control Register	MC_CGM_MUX_12_CSC	–	–
	Select Status Register	MC_CGM_MUX_12_CSS		
	Divider 0 Control Register	MC_CGM_MUX_12_DC_0		QSPI_2X_CLK
Clock mux 14	Select Control Register	MC_CGM_MUX_14_CSC	SDHC_CLK	–
	Select Status Register	MC_CGM_MUX_14_CSS		
	Divider 0 Control Register	MC_CGM_MUX_14_DC_0		–
Clock mux 15	Select Control Register	MC_CGM_MUX_15_CSC	GMAC_0_REF_CLK	–
	Select Status Register	MC_CGM_MUX_15_CSS		
	Divider 0 Control Register	MC_CGM_MUX_15_DC_0		GMAC_0_REF_DIV_CLK
Clock mux 16	Select Control Register	MC_CGM_MUX_16_CSC	SPI_CLK	–
	Select Status Register	MC_CGM_MUX_16_CSS		

1. Do not modify MC_CGM_MUX_0_DC_1.

23.3.2.2 MC_CGM_1 register relationship to output clocks

Clock mux	Description	Register	Selector output	Divider output
MC_CGM_1				
Clock mux 0	Select Control Register	MC_CGM_MUX_0_CSC	A53_CORE_CLK	–
	Select Status Register	MC_CGM_MUX_0_CSS		

23.3.2.3 MC_CGM_2 registers relationship to output clocks

Clock mux	Description	Register	Selector output	Divider output
MC_CGM_2				
Clock mux 0	Select Control Register	MC_CGM_MUX_0_CSC	–	–
	Select Status Register	MC_CGM_MUX_0_CSS		
	Divider 0 Control Register	MC_CGM_MUX_ 0_DC_0		PFE_PE_CLK
Clock mux 1	Select Control Register	MC_CGM_MUX_1_CSC	–	–
	Select Status Register	MC_CGM_MUX_1_CSS		
	Divider 0 Control Register	MC_CGM_MUX_1_DC_0		PFE_MAC_0_TX_DIV_CLK[1]
Clock mux 2	Select Control Register	MC_CGM_MUX_2_CSC	–	–
	Select Status Register	MC_CGM_MUX_2_CSS		
	Divider 0 Control Register	MC_CGM_MUX_2_DC_0		PFE_MAC_1_TX_CLK
Clock mux 3	Select Control Register	MC_CGM_MUX_3_CSC	–	–
	Select Status Register	MC_CGM_MUX_3_CSS		
	Divider 0 Control Register	MC_CGM_MUX_3_DC_0		PFE_MAC_2_TX_CLK
Clock mux 4	Select Control Register	MC_CGM_MUX_4_CSC	PEF_MAC_0_RX_CLK	–
	Select Status Register	MC_CGM_MUX_4_CSS		
Clock mux 5	Select Control Register	MC_CGM_MUX_5_CSC	PEF_MAC_1_RX_CLK	–
	Select Status Register	MC_CGM_MUX_5_CSS		
Clock mux 6	Select Control Register	MC_CGM_MUX_6_CSC	PEF_MAC_2_RX_CLK	–
	Select Status Register	MC_CGM_MUX_6_CSS		
Clock mux 7	Select Control Register	MC_CGM_MUX_7_CSC	–	–

Table continues on the next page...

Table continued from the previous page...

Clock mux	Description	Register	Selector output	Divider output
MC_CGM_2				
	Select Status Register	MC_CGM_MUX_7_CSS		
	Divider 0 Control Register	MC_CGM_MUX_7_DC_0		PFE_MAC_0_REF_DIV_CLK
Clock mux 8	Select Control Register	MC_CGM_MUX_8_CSC	–	–
	Select Status Register	MC_CGM_MUX_8_CSS		
	Divider 0 Control Register	MC_CGM_MUX_8_DC_0		PFE_MAC_1_REF_DIV_CLK
Clock mux 9	Select Control Register	MC_CGM_MUX_9_CSC	–	–
	Select Status Register	MC_CGM_MUX_9_CSS		
	Divider 0 Control Register	MC_CGM_MUX_9_DC_0		PFE_MAC_2_REF_DIV_CLK

1. See PFE_MAC_0 clocking overview for details.

23.3.2.4 MC_CGM_5 registers relationship to output clocks

Clock mux	Description	Register	Selector output	Divider output
MC_CGM_5				
Clock mux 0	Select Control Register	MC_CGM_MUX_0_CSC	DDR_CLK	–
	Select Status Register	MC_CGM_MUX_0_CSS		

23.3.3 Application clock source options

MC_CGM instance	Clock selector[1]	Target application mode clock selection options
MC_CGM_0	Clock mux 0 selector	CORE_DFS1_CLK
MC_CGM_0	Clock mux 1 selector	FXOSC_CLK PERIPH_DFS5_CLK PERIPH_DFS2_CLK PERIPH_PLL_PHI0_CLK

Table continues on the next page...

Table continued from the previous page...

MC_CGM instance	Clock selector[1]	Target application mode clock selection options
MC_CGM_0	Clock mux 2 selector	FXOSC_CLK PERIPH_DFS5_CLK PERIPH_DFS2_CLK PERIPH_PLL_PHI0_CLK
MC_CGM_0	Clock mux 3 selector	PERIPH_PLL_PHI1_CLK
MC_CGM_0	Clock mux 4 selector	PERIPH_PLL_PHI1_CLK FTM0_EXT_REF_CLK
MC_CGM_0	Clock mux 5 selector	PERIPH_PLL_PHI1_CLK FTM1_EXT_REF_CLK
MC_CGM_0	Clock mux 6 selector	PERIPH_PLL_PHI1_CLK FXOSC_CLK
MC_CGM_0	Clock mux 7 selector	PERIPH_PLL_PHI2_CLK FXOSC_CLK
MC_CGM_0	Clock mux 8 selector	PERIPH_PLL_PHI3_CLK FXOSC_CLK
MC_CGM_0	Clock mux 9 selector	PERIPH_PLL_PHI4_CLK GMAC_EXT_TS_CLK
MC_CGM_0	Clock mux 10 selector	PERIPH_PLL_PHI5_CLK SERDES_0_LANE_0_TX_CLK GMAC_0_EXT_TX_CLK GMAC_0_EXT_REF_CLK
MC_CGM_0	Clock mux 11 selector	SERDES_0_LANE_0_CDR_CLK GMAC_0_EXT_RX_CLK GMAC_0_REF_DIV_CLK
MC_CGM_0	Clock mux 12 selector	PERIPH_DFS1_CLK
MC_CGM_0	Clock mux 14 selector	PERIPH_DFS3_CLK
MC_CGM_0	Clock mux 15 selector	GMAC_0_EXT_REF_CLK
MC_CGM_0	Clock mux 16 selector	PERIPH_PLL_PHI7_CLK
MC_CGM_1	Clock mux 0 selector	CORE_PLL_PHI0_CLK CORE_DFS2_CLK
MC_CGM_2	Clock mux 0 Selector	ACCEL_PLL_PHI1_CLK

Table continues on the next page...

Table continued from the previous page...

MC_CGM instance	Clock selector[1]	Target application mode clock selection options
MC_CGM_2	Clock mux 1 Selector	SERDES_1_LANE_0_TX_CLK PERIPH_PLL_PHI5_CLK PFE_MAC_0_EXT_TX_CLK PFE_MAC_0_EXT_REF_CLK
MC_CGM_2	Clock mux 2 Selector	SERDES_1_LANE_1_TX_CLK PERIPH_PLL_PHI5_CLK PFE_MAC_1_EXT_TX_CLK PFE_MAC_1_EXT_REF_CLK
MC_CGM_2	Clock mux 3 Selector	SERDES_0_LANE_1_TX_CLK PERIPH_PLL_PHI5_CLK PFE_MAC_2_EXT_TX_CLK PFE_MAC_2_EXT_REF_CLK
MC_CGM_2	Clock mux 4 Selector	SERDES_1_LANE_0_CDR_CLK PFE_MAC_0_EXT_RX_CLK PFE_MAC_0_REF_DIV_CLK
MC_CGM_2	Clock mux 5 Selector	SERDES_1_LANE_1_CDR_CLK PFE_MAC_1_EXT_RX_CLK PFE_MAC_1_REF_DIV_CLK
MC_CGM_2	Clock mux 6 Selector	SERDES_0_LANE_1_CDR_CLK PFE_MAC_2_EXT_RX_CLK PFE_MAC_2_REF_DIV_CLK
MC_CGM_2	Clock mux 7 Selector	PFE_MAC_0_EXT_REF_CLK
MC_CGM_2	Clock mux 8 Selector	PFE_MAC_1_EXT_REF_CLK
MC_CGM_2	Clock mux 9 Selector	PFE_MAC_2_EXT_REF_CLK
MC_CGM_5	Clock mux 0 Selector	DDR_PLL_PHI0_CLK

1. The default clock selected for all clock mux selectors is FIRC_CLK (out of reset).

NOTE

Current and target clocks must be operational before switching source clocks.

23.3.4 System clock frequency limitations

The maximum operating frequency of system level clocks is shown in the product data sheet. The limits shown in the data sheet must not be exceeded.

23.3.5 Default clock configuration

FIRC_CLK is the default clock for the entire system at power-up. All system clock dividers are set to their default or reset values.

23.3.6 Standby clock configuration

SIRC_CLK is the only clock available during the Standby mode. FIRC_CLK is the clock used after Standby mode exit.

23.4 Software resettable domain

Before deassertion of software resettable domain 1, you must switch the input source of MC_CGM_1 mux 0 (A53_CORE_CLK output) to FIRC_CLK. After switching the clock, deassert the reset of software domain 1.

Before deassertion of software resettable domain 2, you must switch the input source of MC_CGM_2 mux [0:9] to FIRC_CLK. After switching the clocks, deassert the reset of software domain 2.

Before deassertion of software resettable domain 3, you must switch the input source of the following to FIRC_CLK:

- MC_CGM_0 mux_3 (PER_CLK)
- MC_CGM_0 mux_6 (FLEXRAY_PE_CLK)
- MC_CGM_0 mux_7 (CAN_PE_CLK)
- MC_CGM_0 mux_8 (LINFLEXD_CLK, LIN_BAUD_CLK)

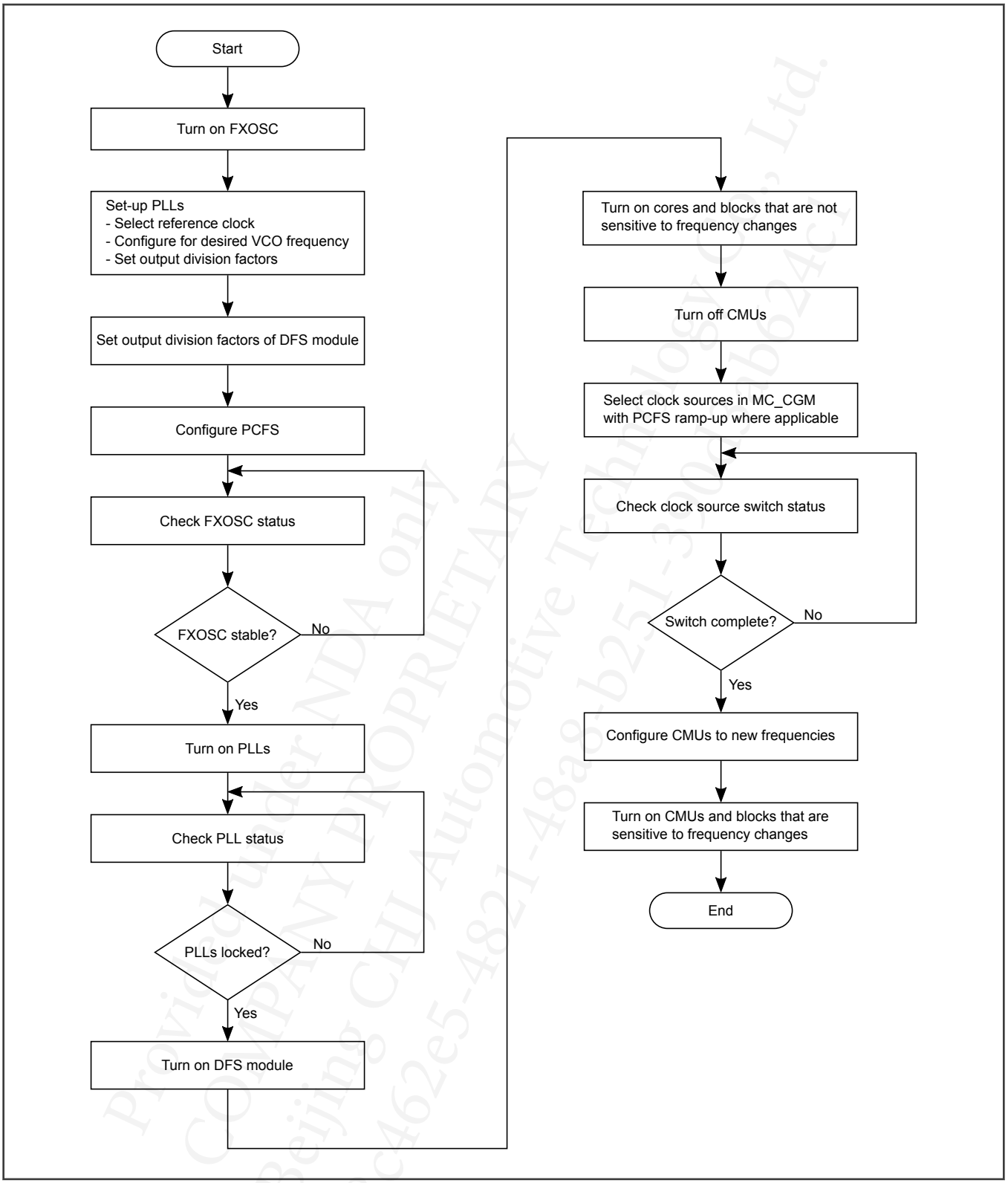
After switching the clocks, deassert the reset of software domain 3.

23.5 Clock on/off procedures

The sections below describe the correct procedures for turning clock signals on and off.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab6240d

23.5.1 Clock-on



NOTE

See PLL turn on sequence for details on correct PLL turn on sequencing.

23.5.1.1 PLL SSCG Setup

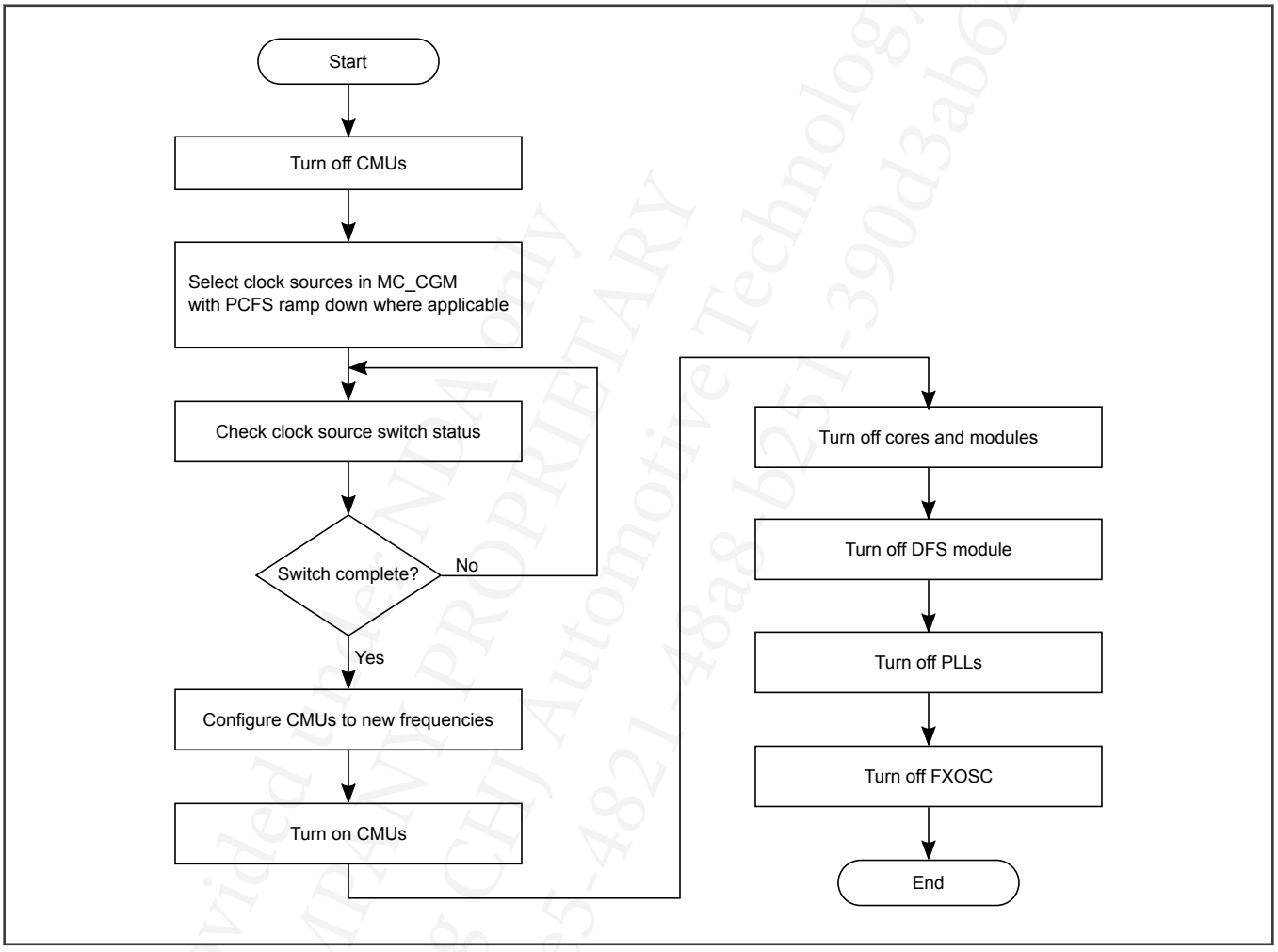
Some PLL instances support SSCG via frequency modulation (FM) to help reduce EMI. See PLL for a list of which PLLs support this feature.

When enabling SSCG for such a PLL, use the center-spread modulation type. To do this, configure the PLL's PLLFM register fields as follows:

- SSCGBYP = 0b
- SPREADCTL = 0b

In this case, refer to the maximum frequencies in the data sheet labeled "with center-spread SSCG enabled" when configuring the PLL. In all other cases, refer to those labeled "without center-spread SSCG enabled".

23.5.2 Clock-off



23.5.3 PLL turn on sequence

Perform the following PLL turn on sequence for correct chip operation:

1. Turn on the AURORA_PLL.
2. Check the AURORA_PLL lock status.
3. Repeat the previous step if the AURORA_PLL is not locked. Else, go to the next step.
4. Turn on the CORE_PLL.

5. Check the CORE_PLL lock status.
6. Repeat the previous step if the CORE_PLL is not locked. Else, go to the next step.
7. Turn on the PERIPH_PLL.
8. Check the PERIPH_PLL lock status.
9. Repeat the previous step if the PERIPH_PLL is not locked. Else, go to the next step.
10. Turn on the DDR_PLL.
11. Check the DDR_PLL lock status.
12. Repeat the previous set if the DDR_PLL is not locked. Else, go to the next step.
13. Turn on the ACCEL_PLL.
14. Check the ACCEL_PLL lock status.
15. Repeat the previous step if the ACCEL_PLL is not locked. Else, the PLL turn on sequencing is complete.

23.5.4 Turning on the CLKOUT*n* signals

To avoid glitches on the CLKOUT*n* signals when turning on the output clocks, you must observe the following:

1. Select the CLKOUT*n* signal functionality by configuring the SIUL2 (for example, SIUL2_MSCR*n*).
2. Configure and turn on the desired output clock source.
3. Select the desired output clock source in the MC_CGM_0 (as shown in Application clock source options).
4. Configure and enable the corresponding output clock divider in the MC_CGM_0 (for example, writing 1 to MC_CGM_0_MUX_1_DC[DE]).

23.5.5 Turning off the CLKOUT*n* signals

To avoid glitches on the CLKOUT*n* signals when turning off the output clocks, you must observe the following:

1. Disable the corresponding output clock divider in the MC_CGM_0 (for example, writing 0 to MC_CGM_0_MUX_1_DC[DE]).
2. Optionally turn off the previously selected output clock source.
3. Optionally reconfigure the CLKOUT*n* pin via the SIUL2 for an alternate functionality.

IMPORTANT

If there is a reset event, a glitch may occur on the CLKOUT*n* signals. In this case, RESET_B asserts indicating to any peripheral monitoring RESET_B that CLKOUT*n* is no longer valid.

23.6 Clock gating

See the partition mapping tables in section "MC_ME partition mapping" of the "Mode Entry Module (MC_ME)" chapter for module clock gating possibilities.

NOTE

For module behavior when the module clock is gated, see the Software resettable domains section in the Reset overview chapter.

23.7 Module clocking

Beijing CHJ Automotive Technology Co., Ltd. COMPANY PROPRIETARY Provided under NDA only c8c46265482e4a8b9251-3900ab624c1

23.7.1 Safety modules

Figure 57 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 70 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK, uses the same source for REG_INTF_CLK as used by the MODULE_CLK.

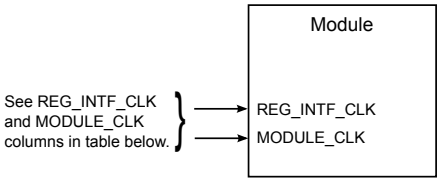


Figure 57. Safety module

Table 70. Safety module clocking

Module	MODULE_CLK	REG_INTF_CLK
CRC	XBAR_DIV3_CLK	
EIM_0	A53_CORE_DIV10_CLK	
EIM_[1:3]	XBAR_DIV6_CLK	
EIM	XBAR_DIV6_CLK	
ERM	See ERM	
FCCU	See FCCU	
RCCU	See Redundancy control and checker unit (RCCU)	
MTR	XBAR_DIV3_CLK	
SBSW	See SBSW	
STCU2	See STCU2	

23.7.1.1 ERM

The following figure shows the ERM clocking configuration.

MEM_CLK connects to the memory clock of the connected module's memories (see the chapter "Error Reporting Module (ERM)" for memory clock details).

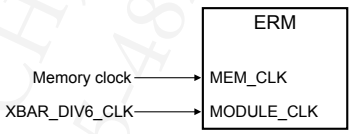
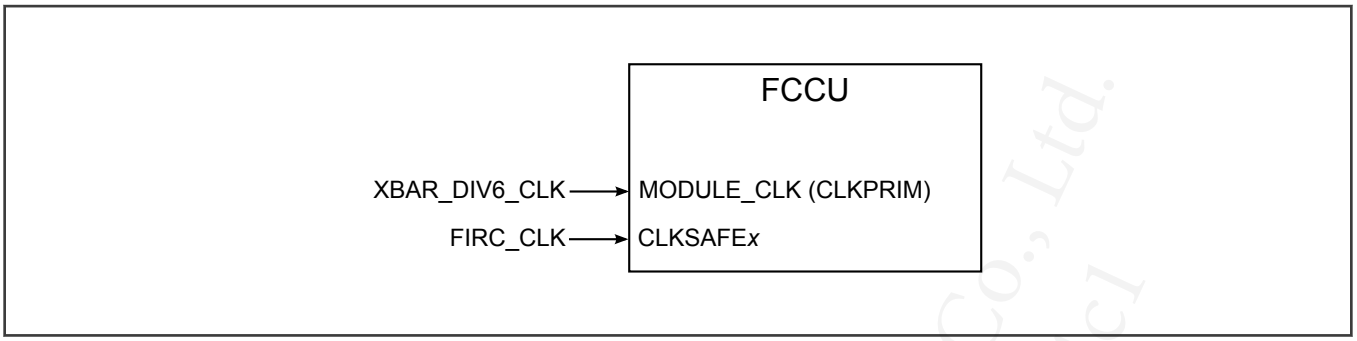


Figure 58. ERM clocking

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e34821-48a8-b0e5-ab624c1

23.7.1.2 FCCU



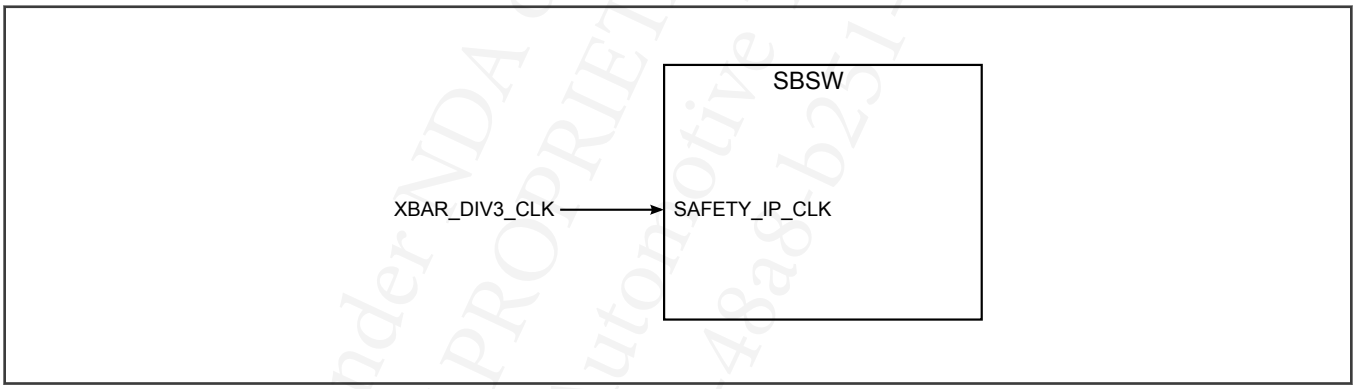
23.7.1.3 Redundancy control and checker unit (RCCU)

In this chip, the RCCU performs lockstep checking for the following:

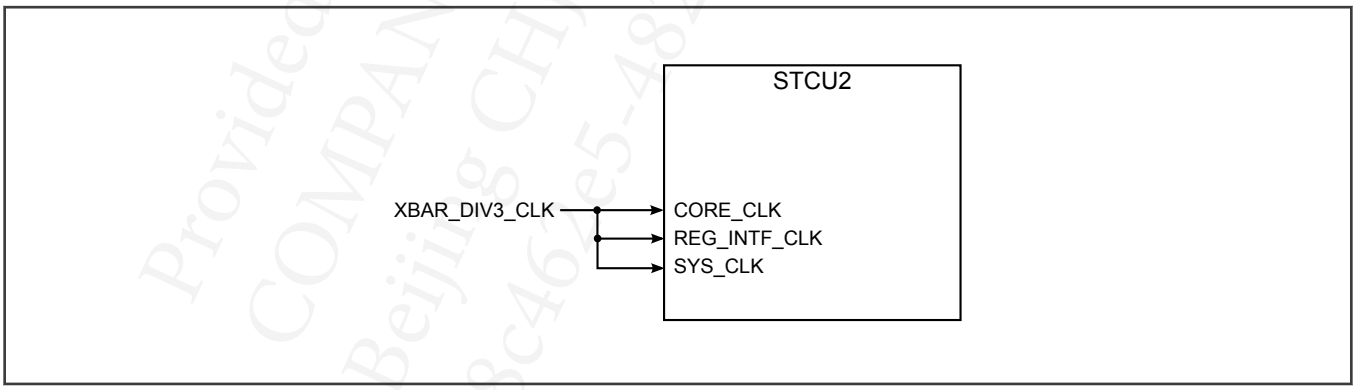
- Cortex-M7
- Cortex-A53
- Enhanced Direct Memory Access (eDMA3)

The RCCU uses the same clock sources as the modules for which it performs lockstep checking (see the section named after each listed module).

23.7.1.4 SBSW



23.7.1.5 STCU2



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46c6e5-4821-48a8-b251-390d3ab624c1

23.7.2 Communication modules

Figure 59 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 71 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK, uses the same source for REG_INTF_CLK as used by the MODULE_CLK.

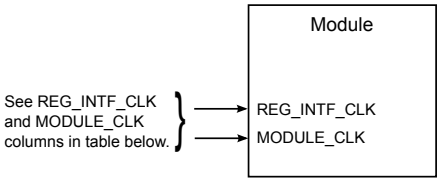


Figure 59. Communication module clocks

Table 71. Communication module clocking

Module	MODULE_CLK	REG_INTF_CLK
SerDes	See SerDes	
Ethernet router	See Ethernet router (PFE) clocking	
Ethernet (GMAC_0)	See Ethernet (GMAC_0)	
SPI	See Serial peripheral interface (SPI) clocking	
FlexCAN	See FlexCAN clocking	
FlexRay	See FlexRay communication controller clocking	
I^2C	See I2C clocking	
LINFlexD	See LINFlexD clocking	
USB	See Universal serial bus (USB) clocking	

23.7.2.1 SerDes

The following figure and table show the SerDes_*n* clocking configuration.

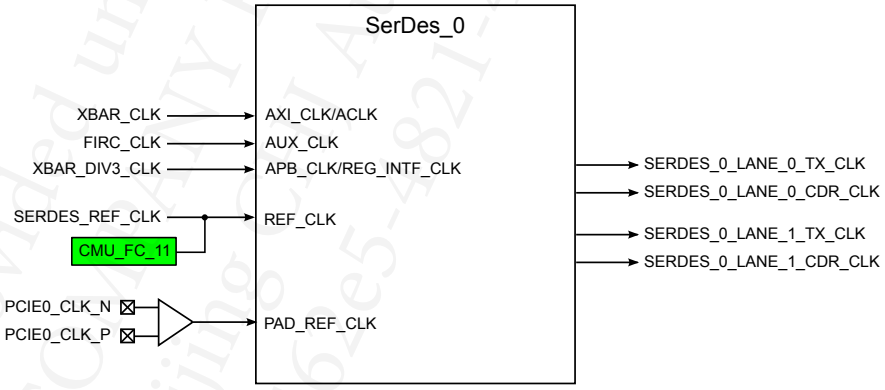


Figure 60. SerDes_0 clocking

Provided under NDA
COMPANY PROPRIETARY
Beijing Chipsea Automotive Technology Co., Ltd.
c8c462e5-4821-4a8b-9525-590d3ab624c1

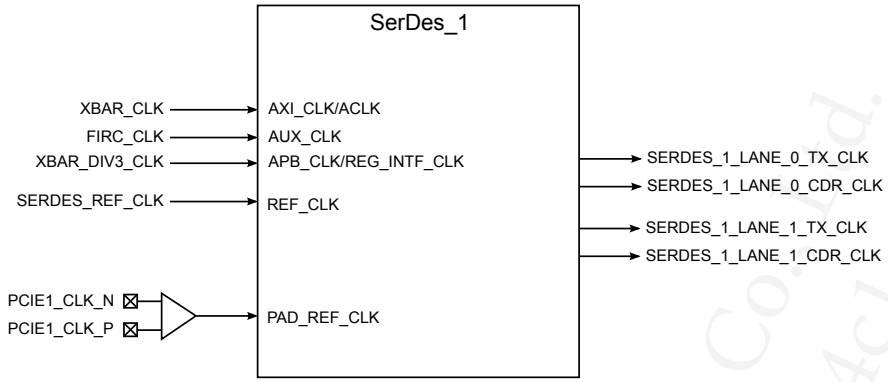


Figure 61. SerDes_1 clocking

NOTE

If using SerDes_*n* output clocks for Ethernet operation, you must configure one of the following inputs:

- PCIE*n*_CLK_N and PCIE*n*_CLK_P
- SERDES_REF_CLK

Table 72. Permissible clock sources for PCIe modes

PCIe mode	Permissible clock source
Gen1	• External clock • Internal PLL
Gen2	
Gen3	Use PCIE*n*_CLK_N and PCIE*n*_CLK_P source only (due to tighter jitter requirements)

NOTE

When you configure SerDes_*n* to use internal clocking, ground PCIE*n*_CLK_N and PCIE*n*_CLK_P to avoid noise on the SerDes_*n* clock inputs.

23.7.2.2 Ethernet (GMAC_0)

Figure 62 shows the overall GMAC_0 Ethernet clocking.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-9151-90d3a1b624c1

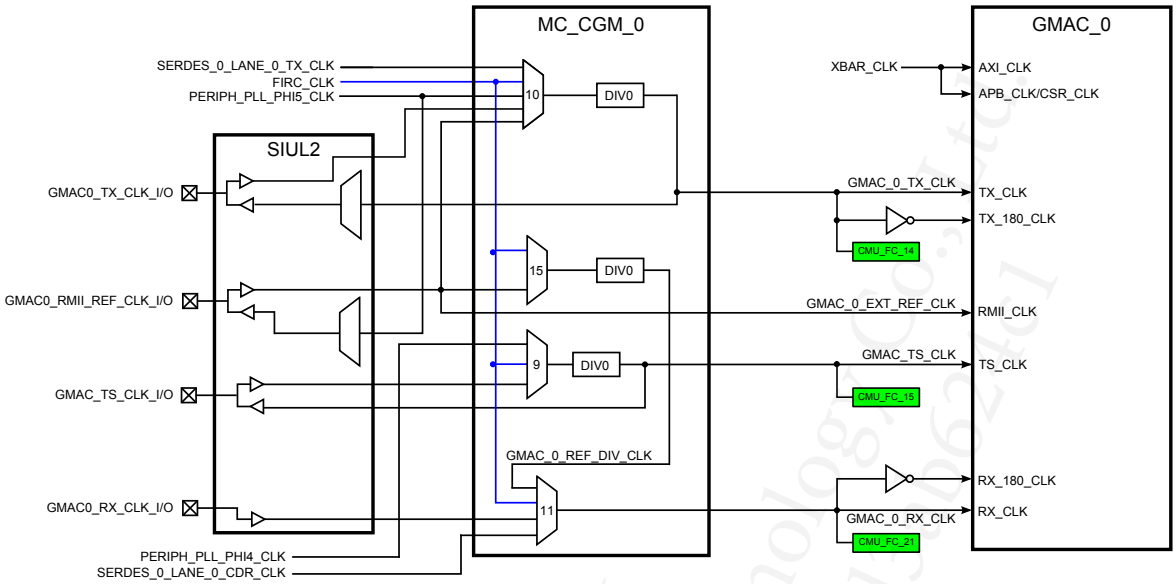


Figure 62. GMAC_0 clocking

NOTE

You should not use FIRC_CLK as the source for any GMAC_0 clock signal.

23.7.2.2.1 Ethernet (GMAC_0) RGMII configuration

Figure 63 shows the GMAC_0 RGMII Ethernet clocking, and Table 73 shows the SIUL2 clock signal configuration for RGMII.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-9251-390d3b62f0c1

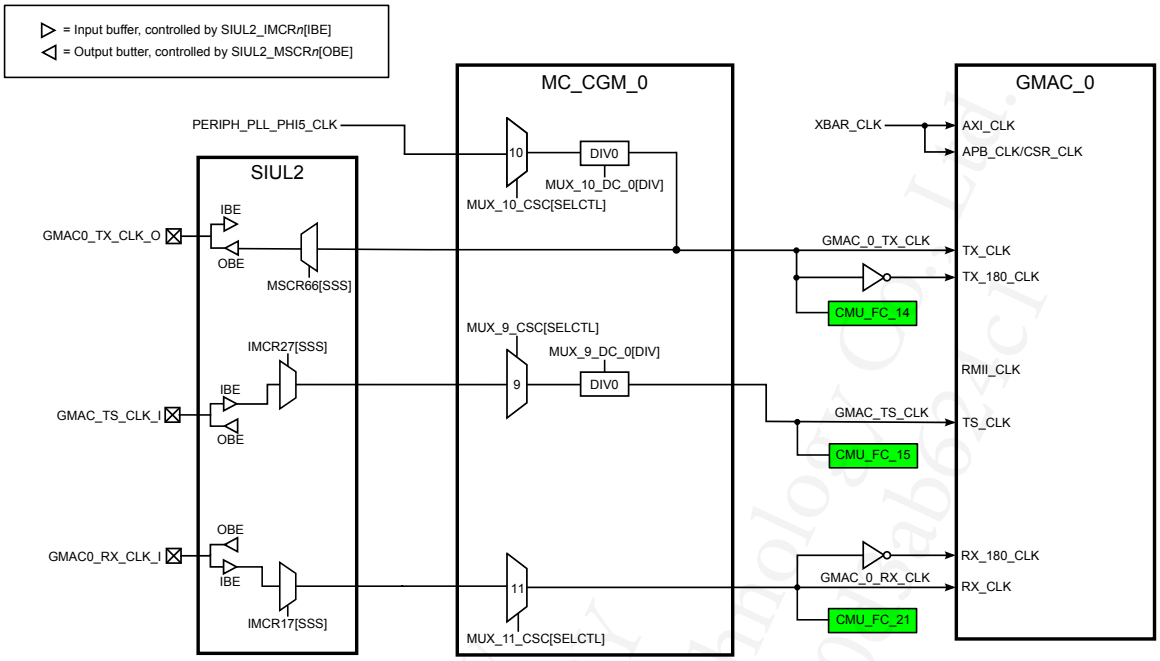


Figure 63. GMAC_0 RGMII clocking

Table 73. GMAC_0 RGMII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	GMAC0_TS_CLK_I	PF_01	81	0	1	X	27	010b
GMAC0_TX_CLK_O	GMAC_0_TX_CLK	PE_02	66	1	0	001b	26	X
GMAC_0_TX_CLK	PERIPH_PLL_PHI5_CLK	—						
GMAC_0_RX_CLK	GMAC0_RX_CLK_I	PE_08	72	0	1	X	17	010b

23.7.2.2.2 Ethernet (GMAC_0) RMII configuration

Figure 64 shows the GMAC_0 RMII Ethernet clocking using and external reference, and Table 74 shows the SIUL2 clock signal configuration for RMII.

Provided under NDA only
COMPANY PROPRIETARY
Beijing C-Life Automotive Technology Co., Ltd.
c8c462e5482a4ab6-251-390

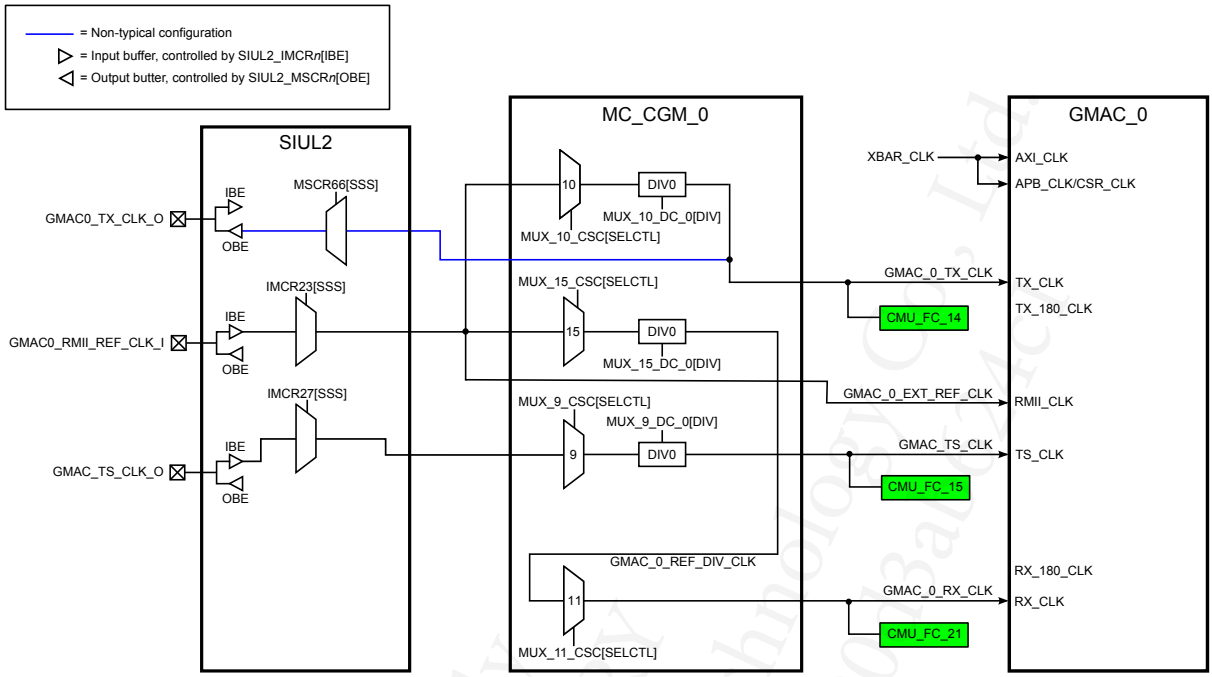


Figure 64. GMAC_0 RMII clocking

Table 74. GMAC_0 RMII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	GMAC0_TS_CLK_I	PF_01	81	0	1	X	27	010b
GMAC_0_EXT_REF_CLK	GMAC0_RMII_REF_CLK_I	PE_07	71	0	1	X	23	010b
GMAC_0_RX_CLK								
GMAC_0_TX_CLK								
GMAC0_TX_CLK_O	GMAC_0_TX_CLK	PE_02	66	1	0	001b	26	X

23.7.2.2.3 Ethernet (GMAC_0) MII configuration

Figure 65 shows the GMAC_0 MII Ethernet clocking, and Table 75 shows the SIUL2 clock signal configuration for MII.

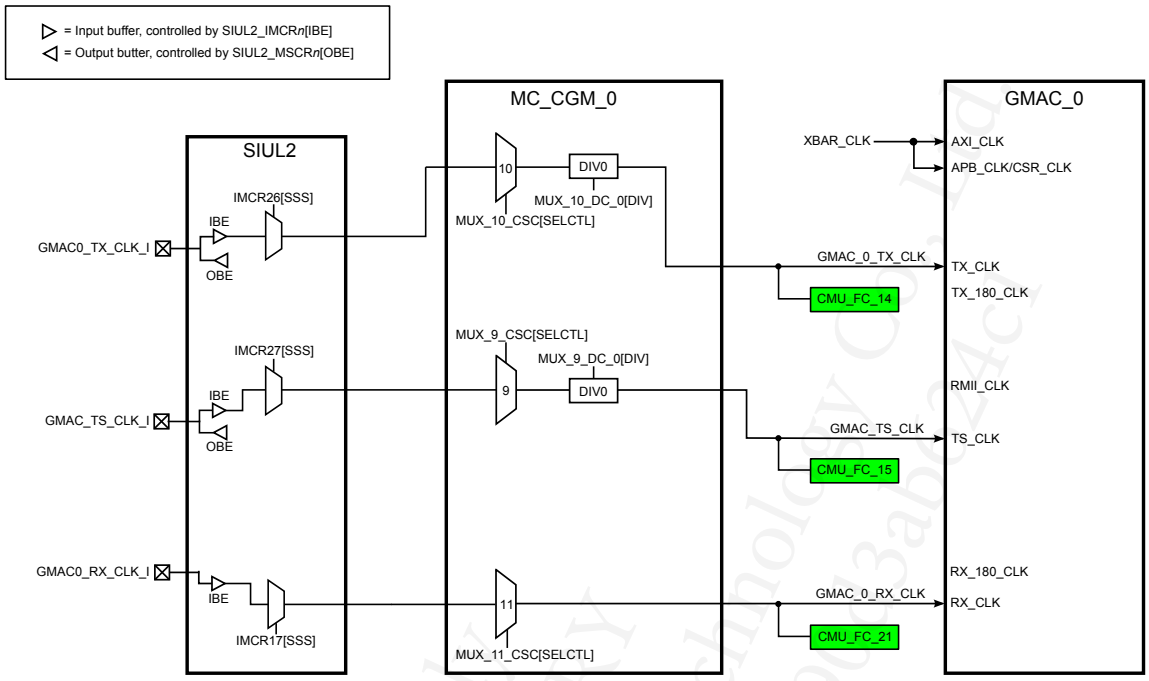


Figure 65. GMAC_0 MII clocking

Table 75. GMAC_0 MII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	GMAC0_TS_CLK_I	PF_01	81	0	1	X	27	010b
GMAC0_TX_CLK_I	GMAC_0_TX_CLK	PE_02	66	0	1	X	26	010b
GMAC_0_RX_CLK	GMAC0_RX_CLK_I	PE_08	72	0	1	X	17	010b

23.7.2.2.4 Ethernet (GMAC_0) SGMII configuration

Figure 66 shows the GMAC_0 SGMII Ethernet clocking, and Table 76 shows the SIUL2 clock signal configuration for SGMII.

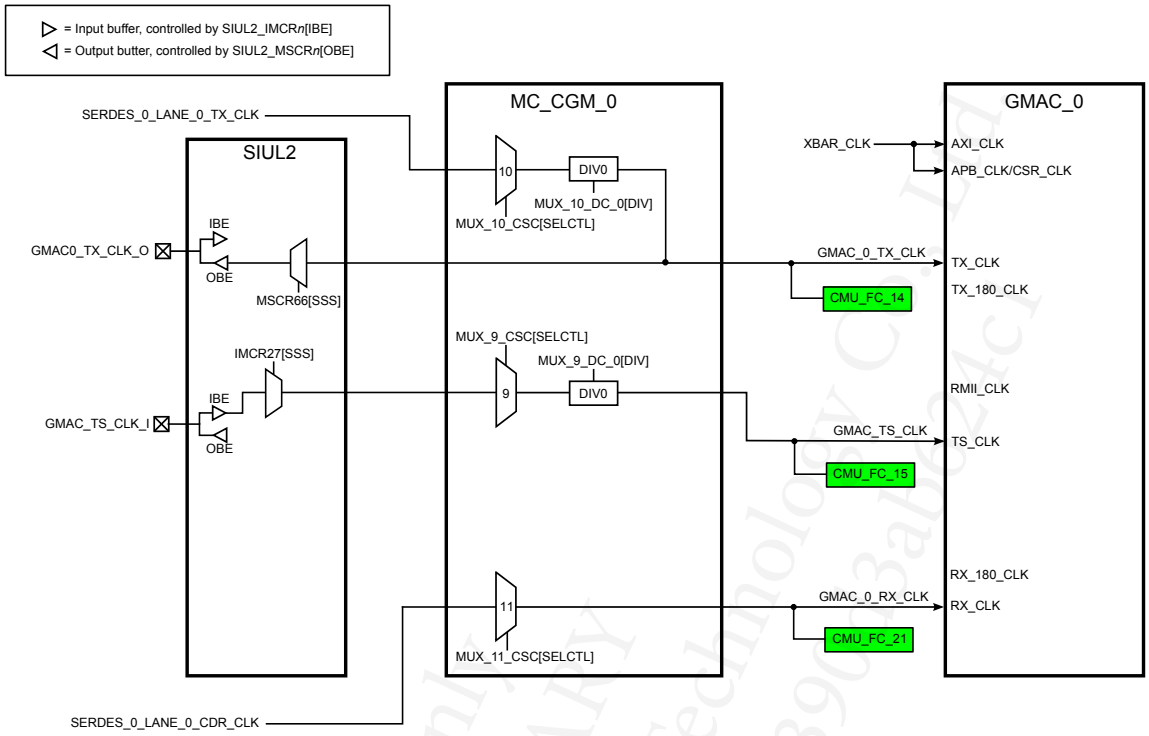


Figure 66. GMAC_0 SGMII clocking

Table 76. GMAC_0 SGMII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	GMAC0_TS_CLK_I	PF_01	81	0	1	X	27	010b
GMAC_0_TX_CLK	SERDES_0_LANE_0_TX_CLK	—						
GMAC0_TX_CLK_O	GMAC_0_TX_CLK	PE_02	66	1	0	001b	26	X
GMAC_0_RX_CLK	SERDES_0_LANE_0_CDR_CLK	—						

23.7.2.3 Ethernet router (PFE) clocking

Figure 67 shows the PFE_MAC_*n* Ethernet router clocking overview (see Ethernet (GMAC_0) for details on GMAC_TS_CLK generation).

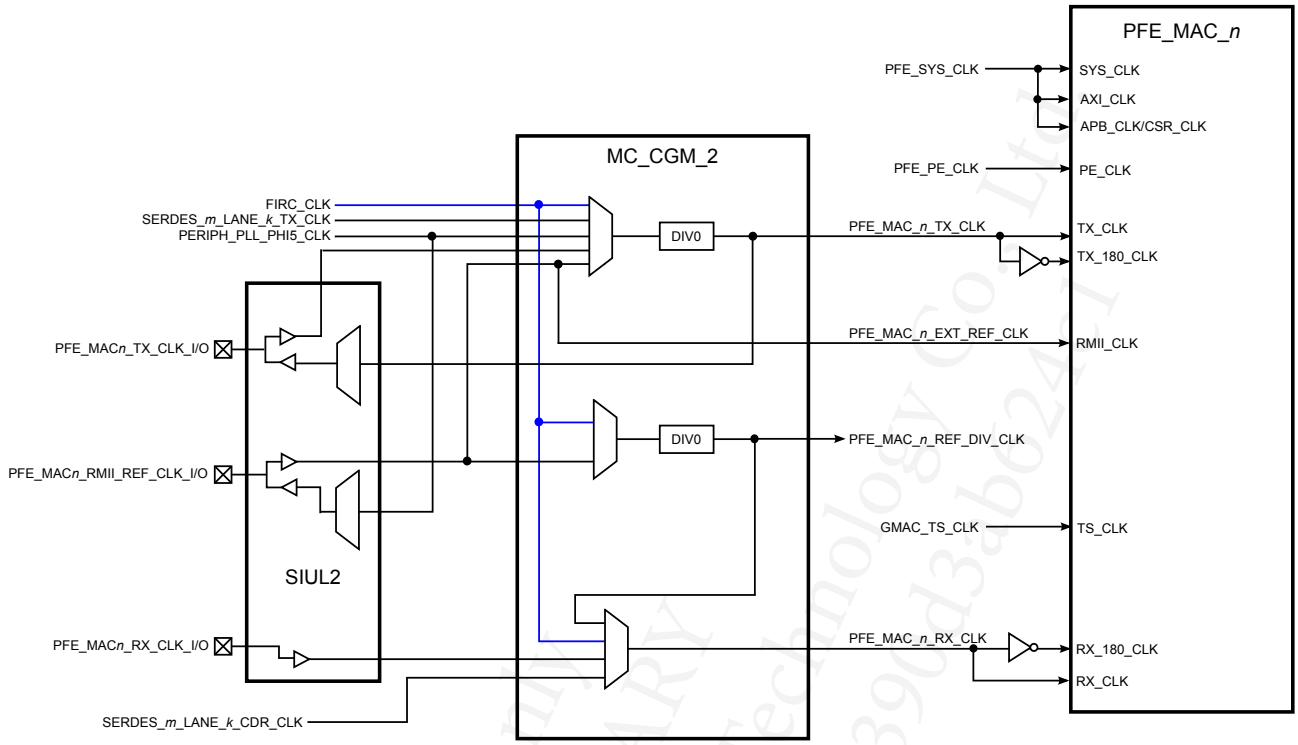


Figure 67. Ethernet router clocking overview

NOTE

You should not use FIRC_CLK as the source for any PFE_MAC_*n* clock signal.

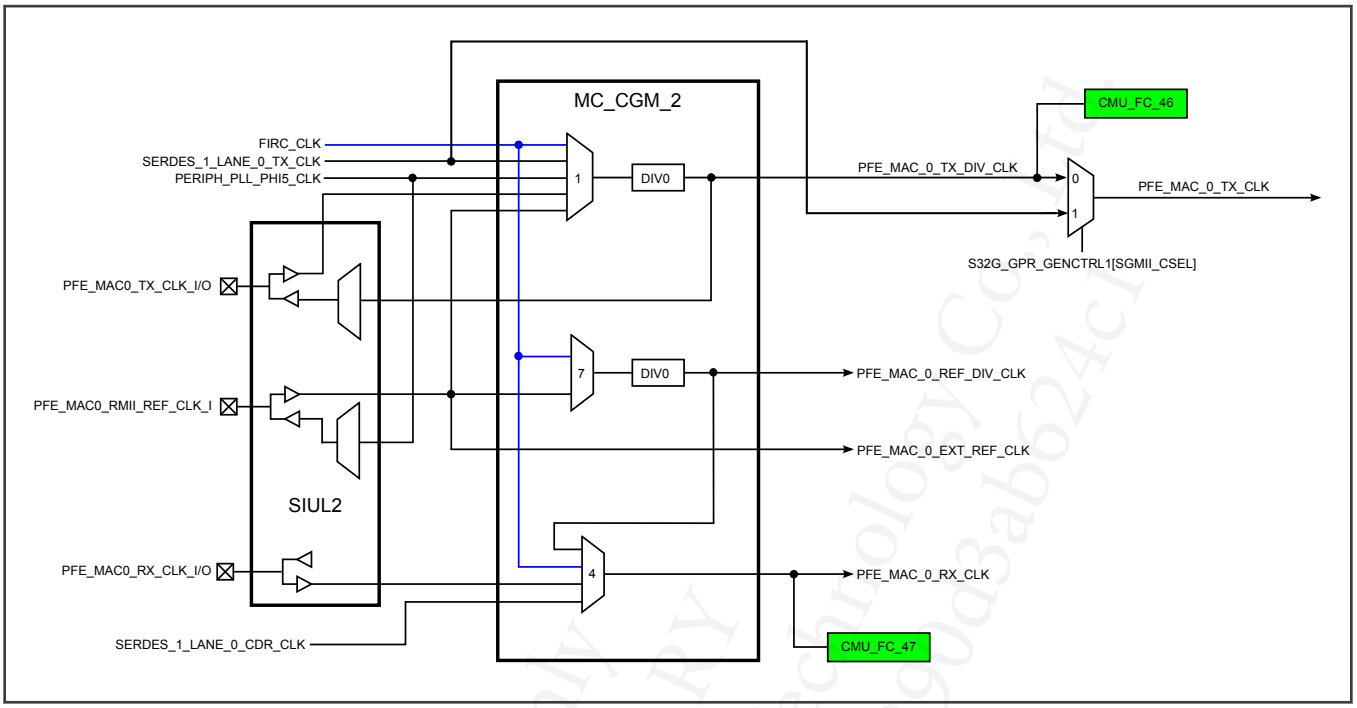
Table 77. SerDes_*m* to PFE_MAC_*n* clock signal relationship

SerDes_*m* clock signal	PFE_MAC_*n* destination
SERDES_1_LANE_0_TX/CDR_CLK	PFE_MAC_0
SERDES_1_LANE_1_TX/CDR_CLK	PFE_MAC_1
SERDES_0_LANE_1_TX/CDR_CLK	PFE_MAC_2

Table 78. Clock selection at multiplexer

Mode	MC_CGM_2 mux clock select (RX_CLK)	MC_CGM_2 mux clock select (TX_CLK)
MII	PFE_MAC*n*_RX_CLK_I	PFE_MAC*n*_TX_CLK_I
RGMII	PFE_MAC*n*_RX_CLK_I	PERIPH_PLL_PHI5_CLK
SGMII	SERDES_*m*_LANE_*k*_CDR_CLK	SERDES_*m*_LANE_*k*_TX_CLK
RMII	PFE_MAC*n*_RMII_REF_CLK_I	PFE_MAC*n*_RMII_REF_CLK_I

23.7.2.3.1 PFE_MAC_0 clocking overview



NOTE

Before switching clock source for PFE_MAC_0_TX_CLK, both mux clock inputs must be disabled (for example, PFE_MAC_0_TX_DIV_CLK and SERDES_1_LANE_0_TX_CLK).

23.7.2.3.1.1 PFE_MAC_0 RGMII configuration

Figure 68 shows the PFE_MAC_0 RMII Ethernet clocking, and Table 79 shows the SIUL2 clock signal configuration for RMII.

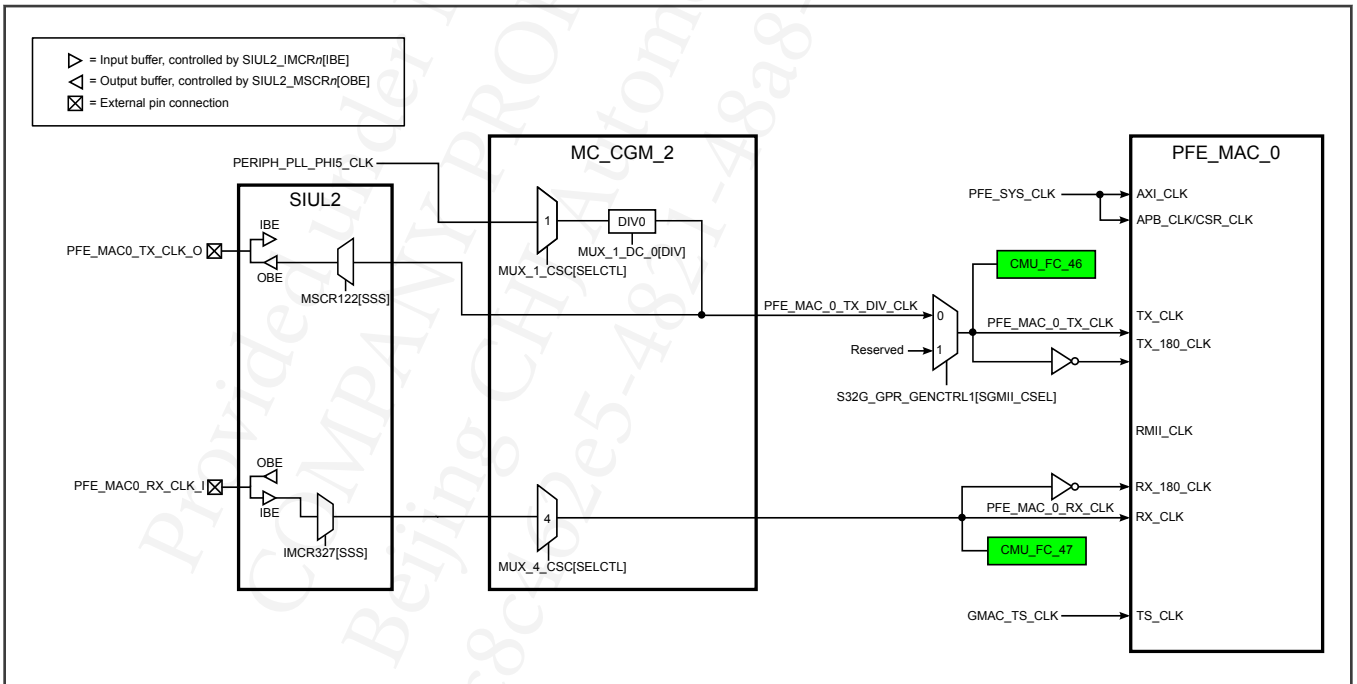


Figure 68. PFE_MAC_0 RGMII clocking

Table 79. PFE_MAC_0 RGMII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	See Ethernet (GMAC_0) RGMII configuration							
PFE_MAC0_TX_CLK_O	PFE_MAC_0_TX_DIV_CLK	PH_10	122	1	0	001b	—	X
PFE_MAC_0_RX_CLK	PFE_MAC0_RX_CLK_I	PH_04	116	0	1	X	327	010b

23.7.2.3.1.2 PFE_MAC_0 RMII configuration

Figure 69 shows the PFE_MAC_0 RMII Ethernet clocking, and Table 80 shows the SIUL2 clock signal configuration for RMII.

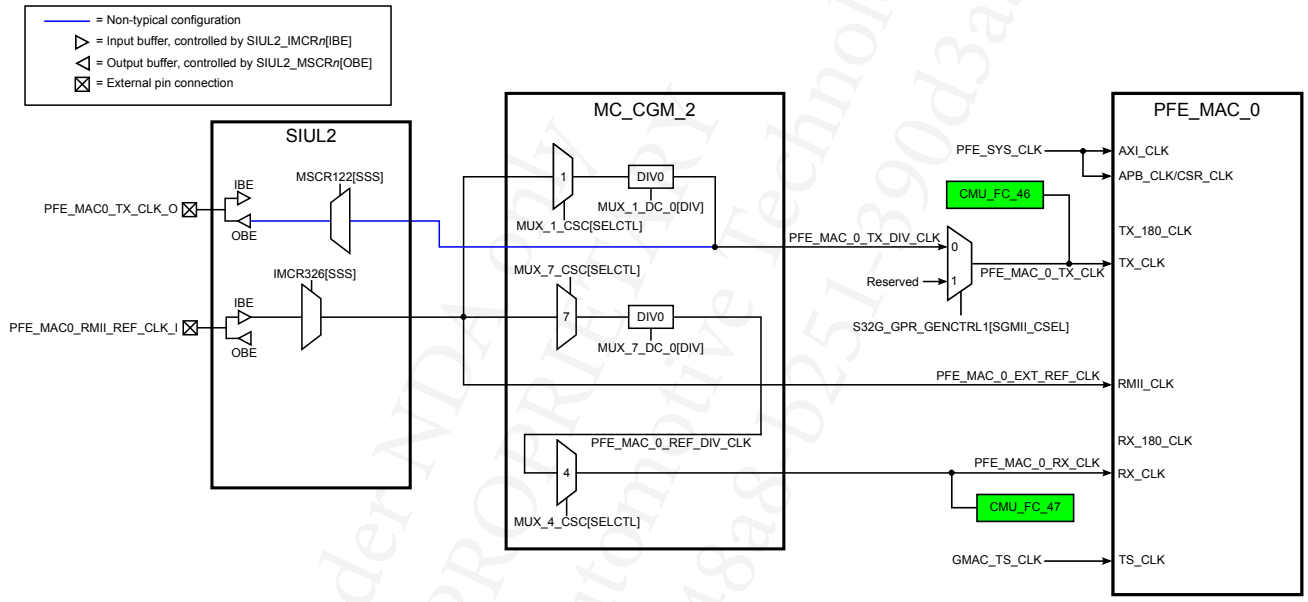


Figure 69. PFE_MAC_0 RMII clocking

Table 80. PFE_MAC_0 RMII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	See Ethernet (GMAC_0) RMII configuration							
PFE_MAC0_TX_CLK_O	PFE_MAC_0_TX_DIV_CLK	PH_10	122	1	0	001b	—	X
PFE_MAC_0_EXT_REF_CLK	PFE_MAC0_RMII_REF_CLK_I	PH_03	7	0	1	X	326	010b

Table continues on the next page...

Table 80. PFE_MAC_0 RMII clock configuration (continued)

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
PFE_MAC_0_TX_CLK								
PFE_MAC_0_RX_CLK								

23.7.2.3.1.3 PFE_MAC_0 MII configuration

Figure 70 shows the PFE_MAC_0 MII Ethernet clocking, and PFE_MAC_0 MII configuration shows the SIUL2 clock signal configuration for MII.

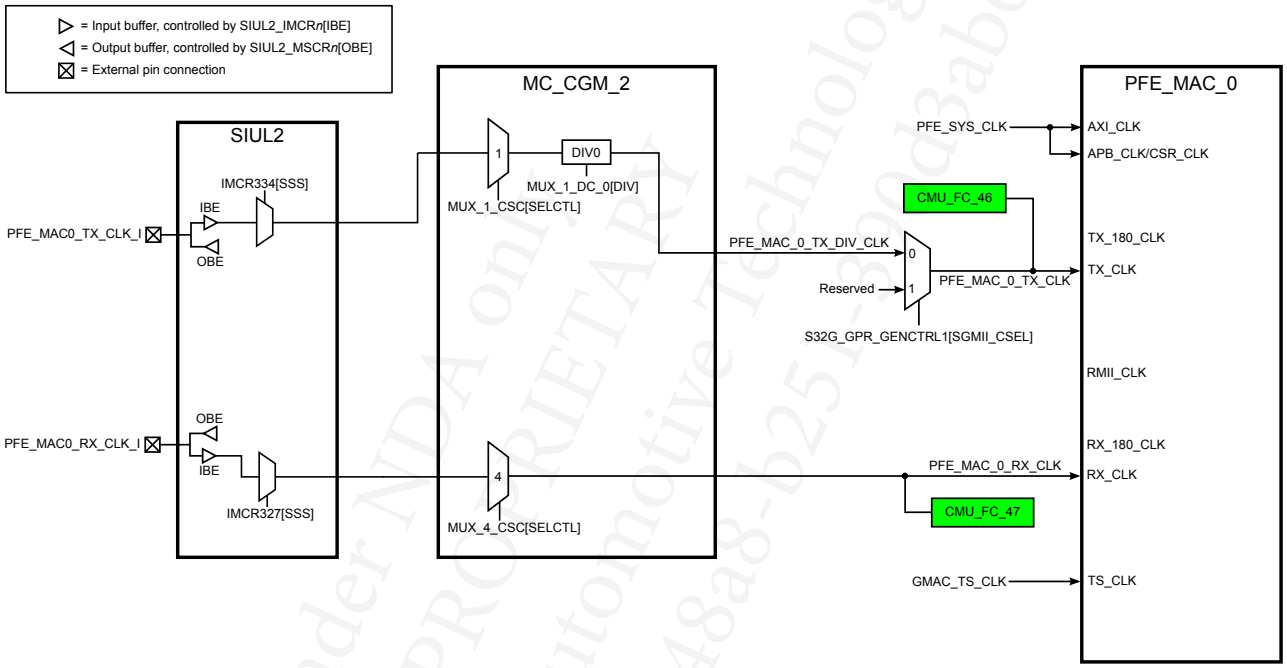


Figure 70. PFE_MAC_0 MII clocking

Table 81. PFE_MAC_0 MII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	See Ethernet (GMAC_0) MII configuration							
PFE_MAC_0_TX_CLK	PFE_MAC0_TX_CLK_I	PH_10	122	0	1	X	334	010b
PFE_MAC_0_RX_CLK	PFE_MAC0_RX_CLK_I	PH_04	116	0	1	X	327	010b

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8a42e25-4821-48a8-b251-300a3ab62ac1

23.7.2.3.1.4 PFE_MAC_0 SGMII configuration

Figure 71 shows the PFE_MAC_0 SGMII Ethernet clocking, and Table 82 shows the SIUL2 clock signal configuration for SGMII.

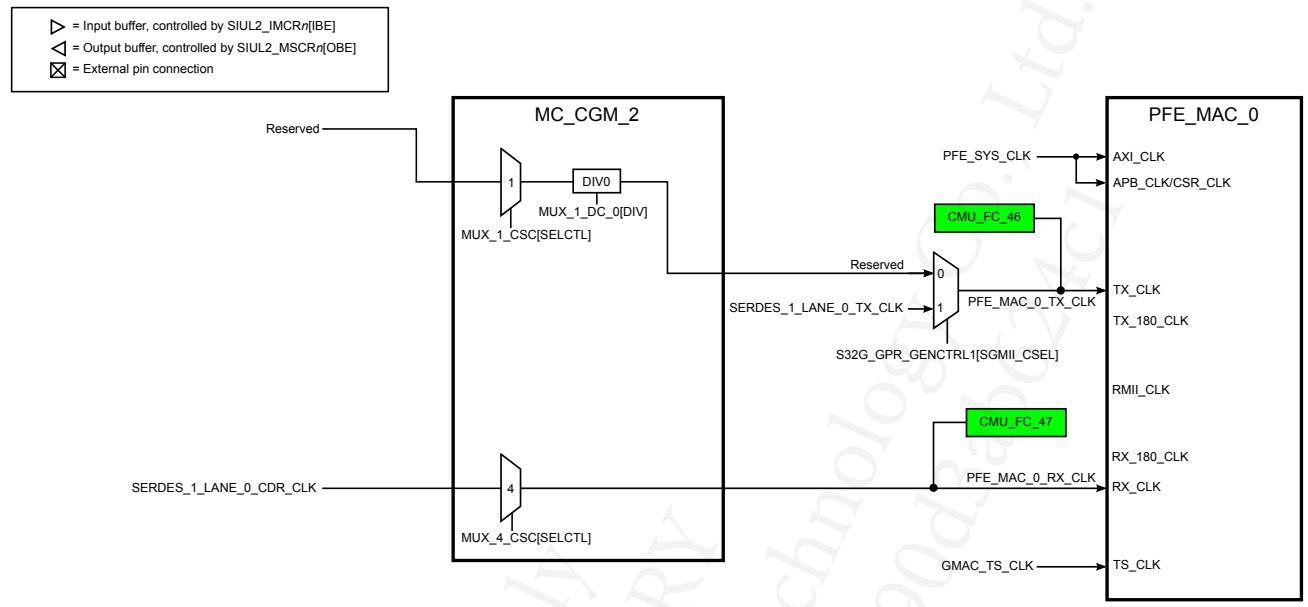


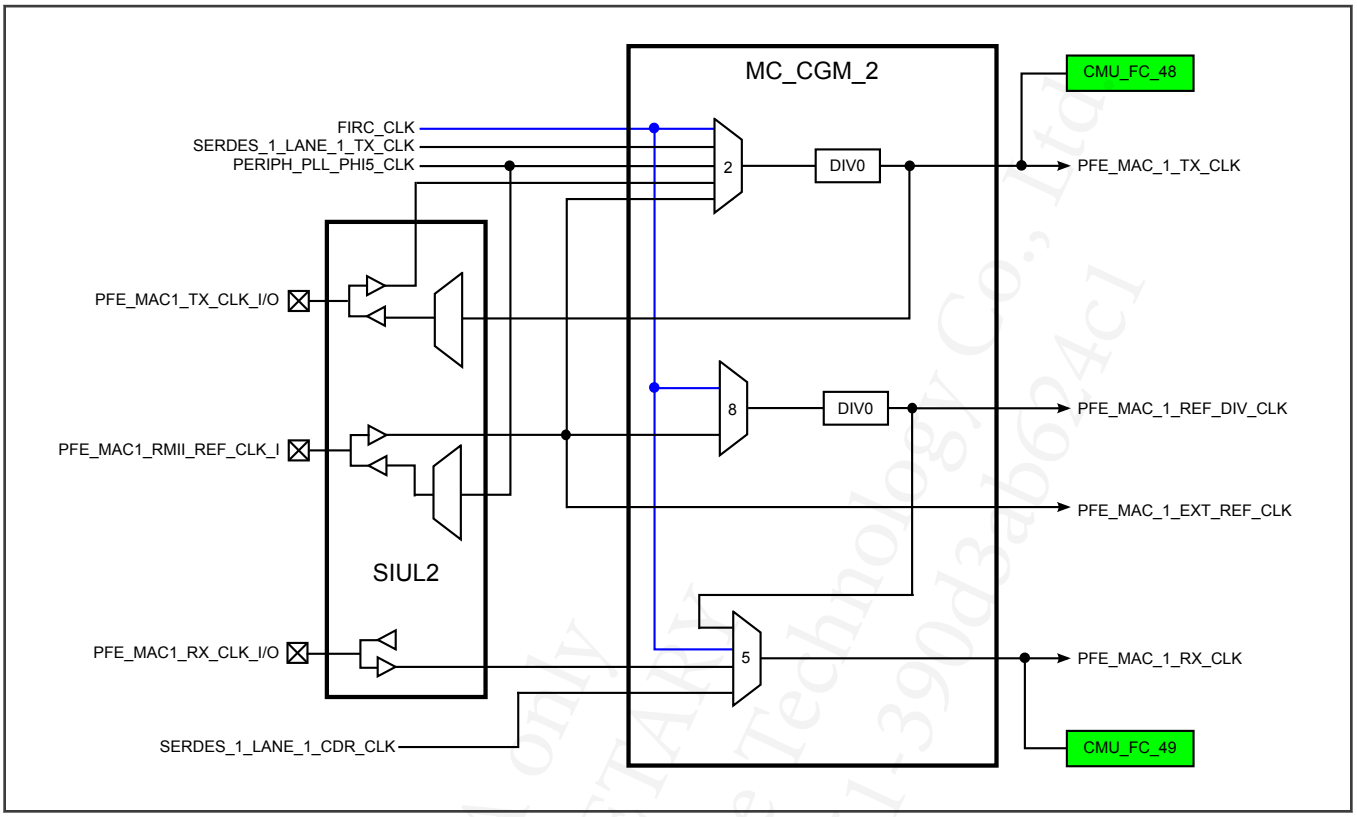
Figure 71. PFE_MAC_0 SGMII clocking

Table 82. PFE_MAC_0 SGMII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	See Ethernet (GMAC_0) SGMII configuration							
PFE_MAC_0_TX_CLK	SERDES_1_LANE_0_TX_CLK	—						
PFE_MAC_0_RX_CLK	SERDES_1_LANE_0_CDR_CLK	—						

Provided under IDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a8b-8251-390d1ab624c1

23.7.2.3.2 PFE_MAC_1 clocking overview



23.7.2.3.2.1 PFE_MAC_1 RGMII configuration

Figure 72 shows the PFE_MAC_1 RGMII Ethernet clocking, and Table 83 shows the SIUL2 clock signal configuration for RGMII.

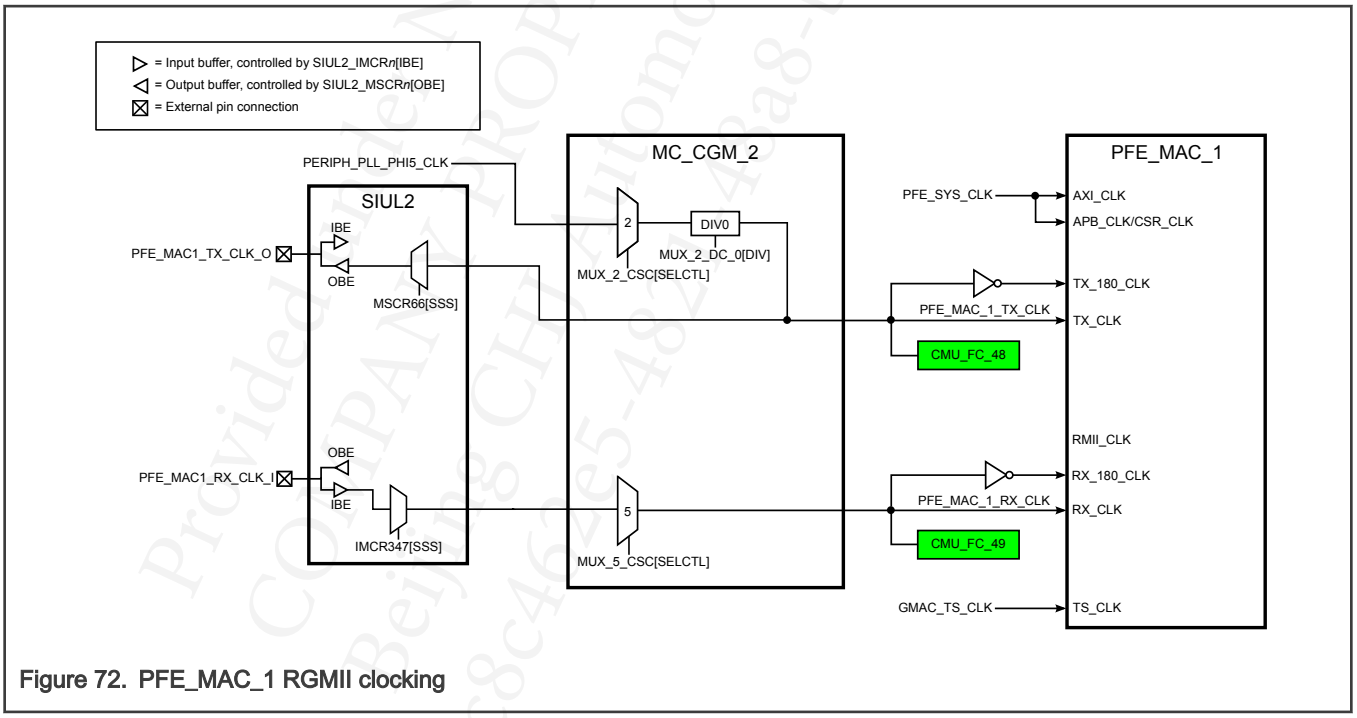


Figure 72. PFE_MAC_1 RGMII clocking

Table 83. PFE_MAC_1 RGMII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	See Ethernet (GMAC_0) RGMII configuration							
PFE_MAC_1_TX_CLK	PFE_MAC1_TX_CLK_I	PH_10	122	1	0	001b	—	X
PFE_MAC_1_RX_CLK	PFE_MAC1_RX_CLK_I	PH_04	116	0	1	X	327	010b

23.7.2.3.2.2 PFE_MAC_1 RMII configuration

Figure 73 shows the PFE_MAC_1 RMII Ethernet clocking, and Table 84 shows the SIUL2 clock signal configuration for RMII.

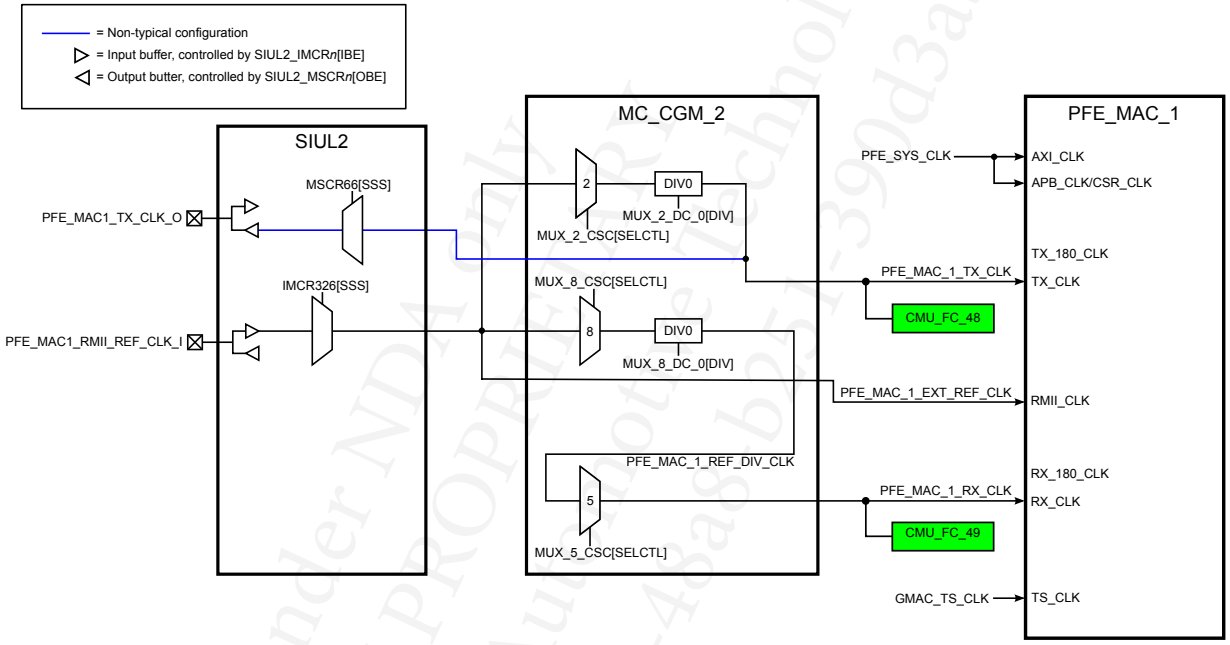


Figure 73. PFE_MAC_1 RMII clocking

Table 84. PFE_MAC_1 RMII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	See Ethernet (GMAC_0) RMII configuration							
PFE_MAC_1_TX_CLK	PFE_MAC1_TX_CLK_I	PH_10	122	1	0	001b	—	X
PFE_MAC_1_EXT_REF_CLK	PFE_MAC1_RMII_REF_CLK_I	PE_07	71	0	1	X	326	010b

Table continues on the next page...

Table 84. PFE_MAC_1 RMII clock configuration (continued)

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
PFE_MAC_1_TX_CLK								
PFE_MAC_1_RX_CLK								

23.7.2.3.2.3 PFE_MAC_1 MII configuration

Figure 74 shows the PFE_MAC_1 MII Ethernet clocking, and Table 85 shows the SIUL2 clock signal configuration for PFE_MAC_1 MII.

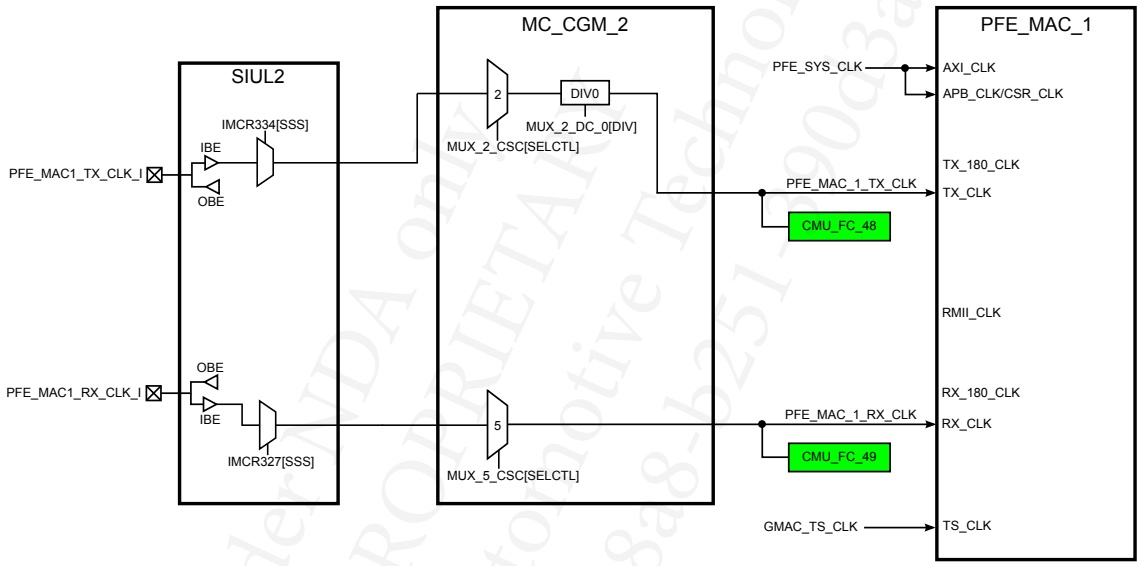


Figure 74. PFE_MAC_1 MII clocking

Table 85. PFE_MAC_1 MII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	See Ethernet (GMAC_0) MII configuration							
PFE_MAC_1_TX_CLK	PFE_MAC1_TX_CLK_I	PE_02	66	0	1	X	354	010b
PFE_MAC_1_RX_CLK	PFE_MAC1_RX_CLK_I	PH_04	116	0	1	X	327	010b

23.7.2.3.2.4 PFE_MAC_1 SGMII configuration

Figure 75 shows the PFE_MAC_1 SGMII Ethernet clocking, and Table 86 shows the SIUL2 clock signal configuration for PFE_MAC_1 SGMII.

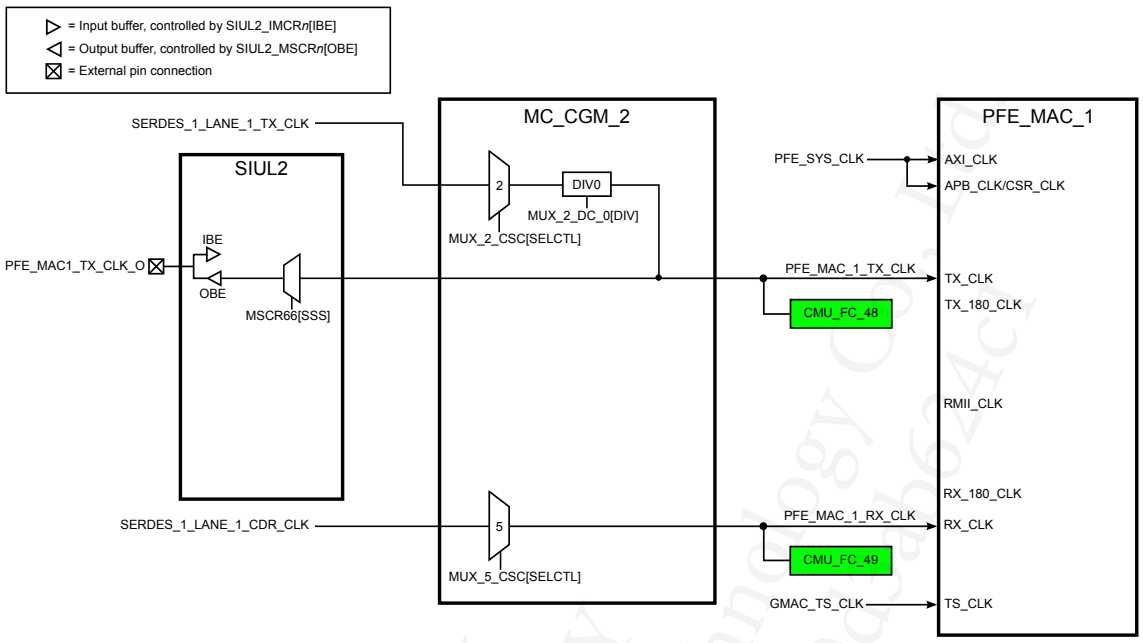


Figure 75. PFE_MAC_1 SGMII clocking

Table 86. PFE_MAC_1 SGMII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	See Ethernet (GMAC_0) SGMII configuration							
PFE_MAC_1_TX_CLK	SERDES_1_LANE_1_TX_CLK	—						
PFE_MAC_1_TX_CLK	PFE_MAC1_TX_CLK_I	PH_10	122	1	0	001b	—	X
PFE_MAC_1_RX_CLK	SERDES_1_LANE_1_CDR_CLK	—						

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4aa8-8251-390a3b624c7

23.7.2.3.3 PFE_MAC_2 clocking overview

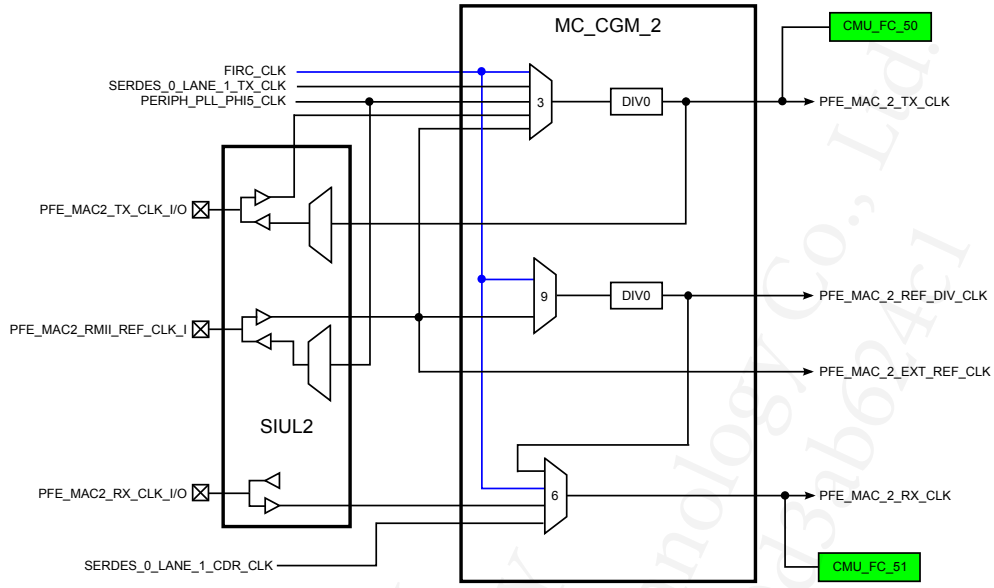


Figure 76. PFE_MAC_2 clocking

23.7.2.3.3.1 PFE_MAC_2 RGMII configuration

Figure 77 shows the PFE_MAC_2 RGMII Ethernet clocking, and Table 87 shows the SIUL2 clock signal configuration for RGMII.

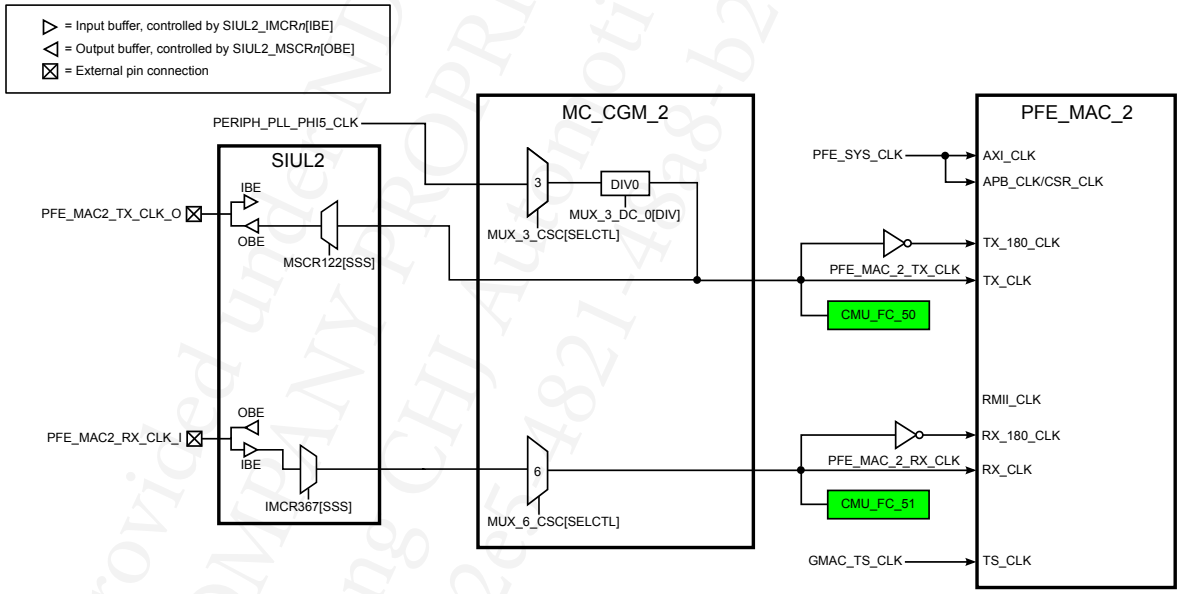


Figure 77. PFE_MAC_2 RGMII clocking

Table 87. PFE_MAC_2 RGMII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	See Ethernet (GMAC_0) RGMII configuration							
PFE_MAC_2_TX_CLK	PERIPH_PLL_PHI5_CLK	—						
PFE_MAC2_TX_CLK_O	PFE_MAC_2_TX_CLK	PH_10	122	1	0	010b	—	X
PFE_MAC_2_RX_CLK	PFE_MAC2_RX_CLK_I	PH_04	116	0	1	X	327	010b
PFE_MAC_2_RX_CLK	PFE_MAC2_RX_CLK_I	PE_00	64	0	1	X	367	010b

23.7.2.3.3.2 PFE_MAC_2 RMII configuration

Figure 78 shows the PFE_MAC_2 RMII Ethernet clocking, and Table 88 shows the SIUL2 clock signal configuration for RMII.

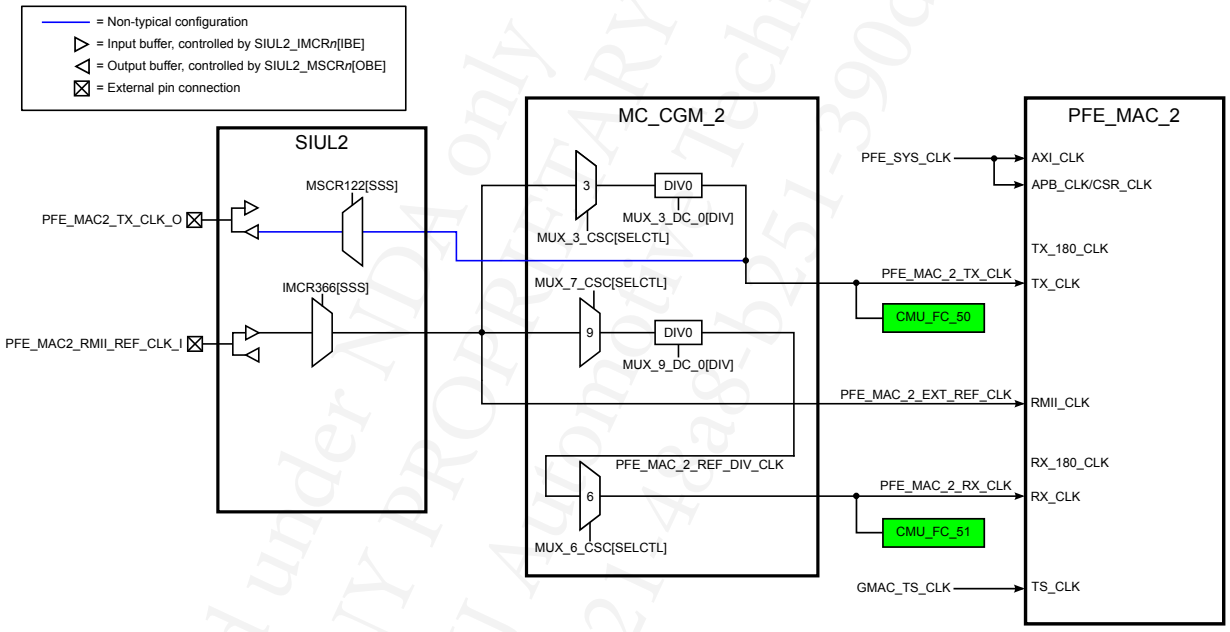


Figure 78. PFE_MAC_2 RMII clocking

Table 88. PFE_MAC_2 RMII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	See Ethernet (GMAC_0) RMII configuration							
PFE_MAC2_TX_CLK_O	PFE_MAC_2_TX_CLK	PH_10	122	1	0	010b	—	X

Table continues on the next page...

Table 88. PFE_MAC_2 RMII clock configuration (continued)

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
PFE_MAC_2_EXT_REF_CLK	PFE_MAC2_RMII_REF_CLK_I	PH_03	115	0	1	X	326	010b
PFE_MAC_2_TX_CLK								
PFE_MAC_2_RX_CLK								

23.7.2.3.3.3 PFE_MAC_2 MII configuration

Figure 79 shows the PFE_MAC_2 MII Ethernet clocking, and Table 89 shows the SIUL2 clock signal configuration for PFE_MAC_2 MII.

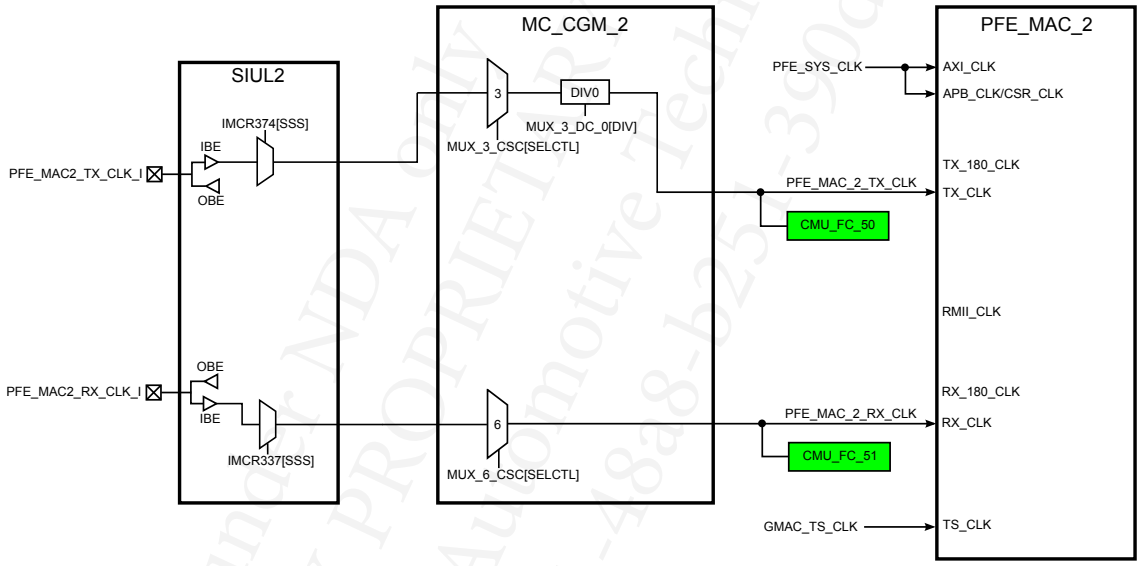


Figure 79. PFE_MAC_2 MII clocking

Table 89. PFE_MAC_2 MII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	See Ethernet (GMAC_0) MII configuration							
PFE_MAC_2_TX_CLK	PFE_MAC2_TX_CLK_I	PH_10	122	0	1	X	374	011b
		PL_08	184	0	1	X		010b

Table continues on the next page...

Table 89. PFE_MAC_2 MII clock configuration (continued)

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
PFE_MAC_2_RX_CLK	PFE_MAC2_RX_CLK_I	PH_04	116	0	1	X	327	010b
PFE_MAC_2_RX_CLK	PFE_MAC2_RX_CLK_I	PE_00	64	0	1	X	367	010b

23.7.2.3.3.4 PFE_MAC_2 SGMII configuration

Figure 80 shows the PFE_MAC_2 SGMII Ethernet clocking, and Table 90 shows the SIUL2 clock signal configuration for PFE_MAC_2 SGMII.

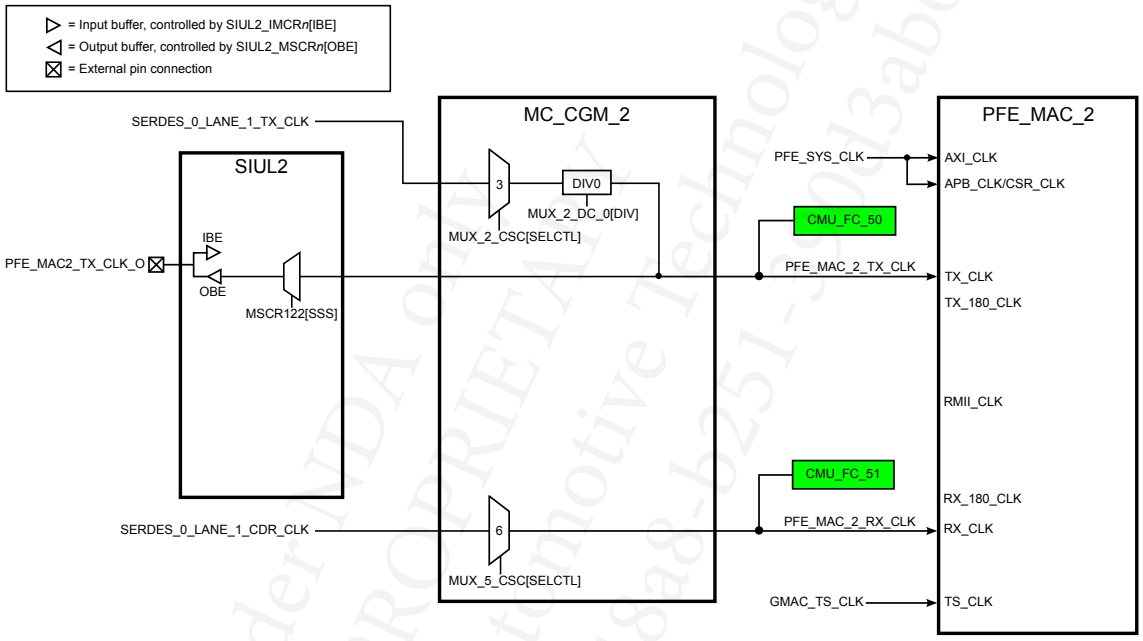


Figure 80. PFE_MAC_2 SGMII clocking

Table 90. PFE_MAC_2 SGMII clock configuration

Destination clock	Source clock	I/O Port	SIUL2					
			MSCR*a*	MSCR*a* fields			IMCR*b*	IMCR*b*[SSS]
				OBE	IBE	SSS		
GMAC_TS_CLK	See Ethernet (GMAC_0) SGMII configuration							
PFE_MAC_2_TX_CLK	SERDES_0_LANE_1_TX_CLK	—						
PFE_MAC2_TX_CLK_O	PFE_MAC_2_TX_CLK	PH_10	122	1	0	010b	—	X
PFE_MAC_2_RX_CLK	SERDES_0_LANE_1_CDR_CLK	—						

23.7.2.4 LLCE subsystem clocking overview

Figure 81 shows the LLCE clocking, and Table 91 shows the SIUL2 clock signal configuration for LLCE LPSPI*n*.

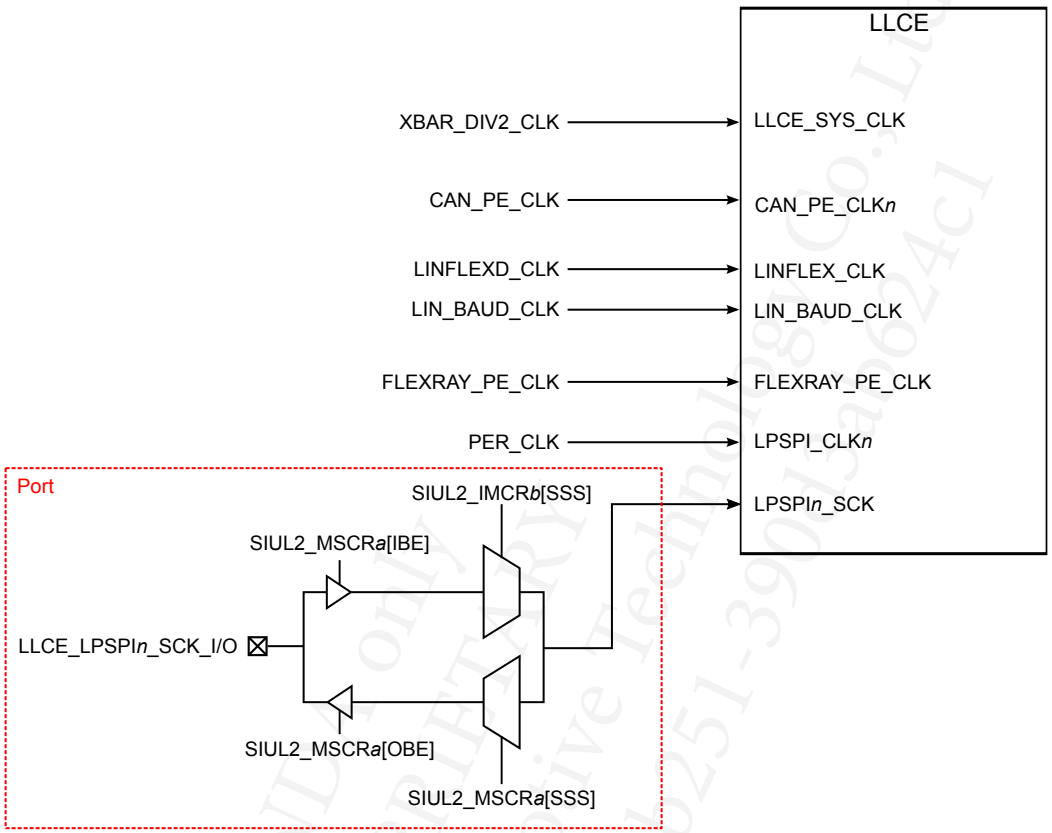


Figure 81. LLCE clocking

Table 91. SIUL2 options for LLCE_LPSPI*n* I/O clocks

LPSPI*n*	SPI mode	I/O Signal	Port	MSCR*a*	MSCR fields			IMCR*b*	IMCR[SSS]
					OBE	IBE	SSS		
LPSPI0_SCK	Slave	LLCE_LPSPI0_SCK_I	PA_09	9	0	1	X	288	010b
	Master	LLCE_LPSPI0_SCK_O			1	0	001b		X
LPSPI1_SCK	Slave	LLCE_LPSPI1_SCK_I	PH_06	118	0	1	X	296	011b
	Master	LLCE_LPSPI1_SCK_O			1	0	010b		X
	Slave	LLCE_LPSPI1_SCK_I	PL_13	189	0	1	X		010b

Table continues on the next page...

Table 91. SIUL2 options for LLCE_LPSPI*n* I/O clocks (continued)

LPSPI*n*	SPI mode	I/O Signal	Port	MSCR*a*	MSCR fields			IMCR*b*	IMCR[SSS]
					OBE	IBE	SSS		
	Master	LLCE_LPSPI1_SCK_O			1	0	101b		X
LPSPI2_SCK	Slave	LLCE_LPSPI2_SCK_I	PB_11	27	0	1	X	304	010b
	Master	LLCE_LPSPI2_SCK_O			1	0	100b		X
LPSPI3_SCK	Slave	LLCE_LPSPI3_SCK_I	PL_01	177	0	1	X	312	010b
	Master	LLCE_LPSPI3_SCK_O			1	0	010b		X

23.7.2.5 Serial peripheral interface (SPI) clocking

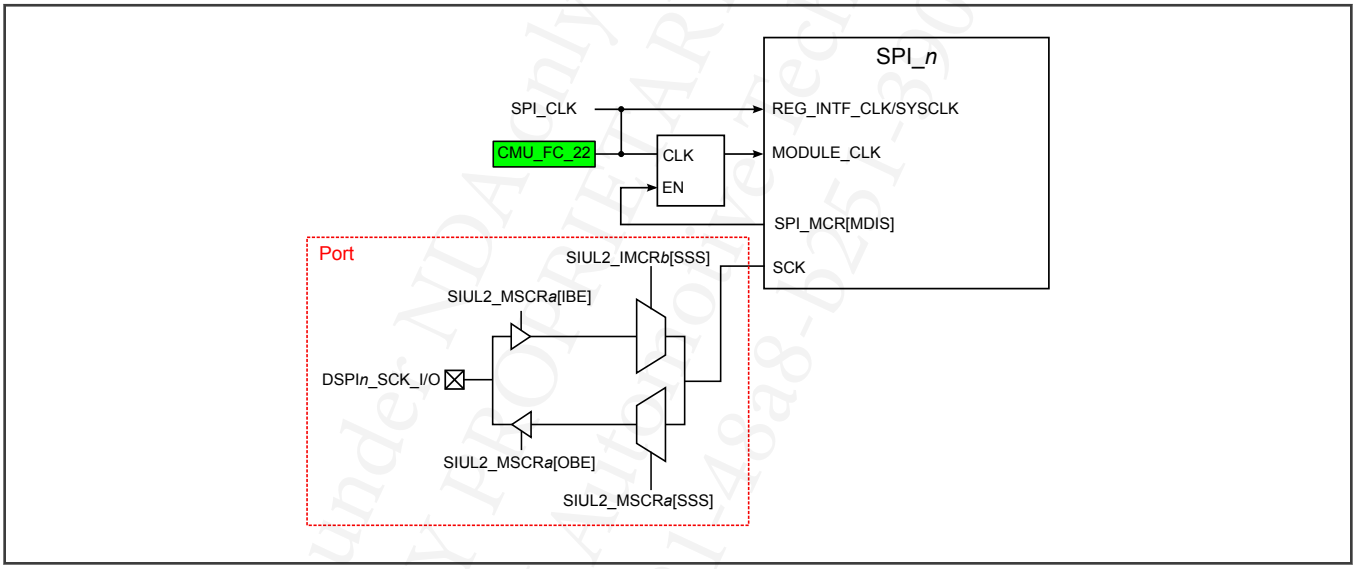


Table 92. SIUL2 options for SPI*n*_SCK_I/O clock

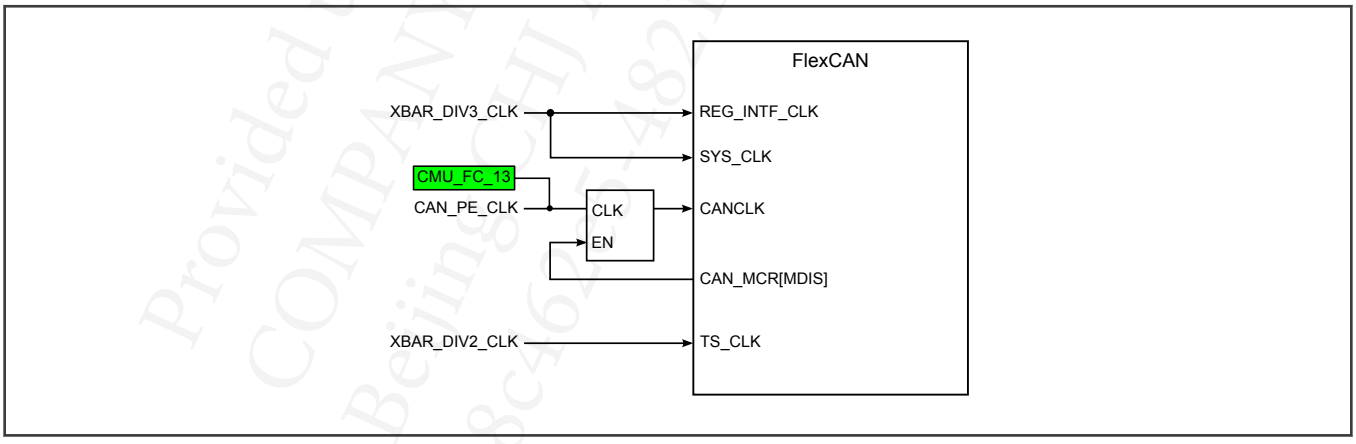
SPI_*n* signal	SPI mode	I/O signal	Port	MSCR *a*	MSCR fields			IMCR *b*	IMCR[SSS]
					OBE	IBE	SSS		
SPI_0 SCK	Slave	SPI0_SCK_I	PA_13	13	0	1	X	469	010b
	Master	SPI0_SCK_O			1	0	001b		X
SPI_1 SCK	Slave	SPI1_SCK_I	PA_08	8	0	1	X	474	010b
	Master	SPI1_SCK_O			1	0	011b		X
SPI_2 SCK	Slave	SPI2_SCK_I	PI_09	153	0	1	X	479	011b

Table continues on the next page...

Table 92. SIUL2 options for SPI*n*_SCK_I/O clock (continued)

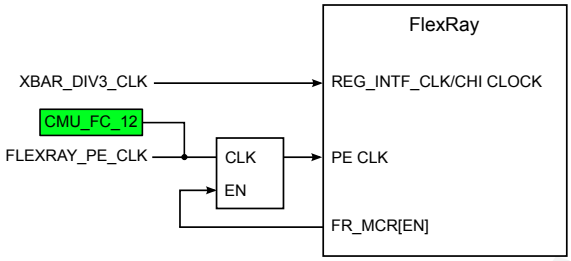
SPI_*n* signal	SPI mode	I/O signal	Port	MSCR *a*	MSCR fields			IMCR *b*	IMCR[SSS]
					OBE	IBE	SSS		
	Master	SPI2_SCK_O			1	0	010b		X
	Slave	SPI2_SCK_I	PI_08	184	0	1	X		010b
	Master	SPI2_SCK_O			1	0	001b		X
	Slave	SPI2_SCK_I	PJ_12	156	0	1	X		100b
SPI_3 SCK	Slave	SPI3_SCK_I	PJ_15	159	0	1	X	484	011b
	Master	SPI3_SCK_O			1	0	010b		X
	Slave	SPI3_SCK_I	PL_13	189	0	1	X		010b
	Master	SPI3_SCK_O			1	0	010b		X
	Slave	SPI3_SCK_I	PK_02	162	0	1	X		100b
SPI_4 SCK	Slave	SPI4_SCK_I	PC_09	41	0	1	X	489	010b
	Master	SPI4_SCK_O			1	0	100b		X
	Slave	SPI4_SCK_I	PK_05	165	0	1	X		011b
	Master	SPI4_SCK_O			1	0	010b		X
	Slave	SPI4_SCK_I	PK_06	162	0	1	X		100b
SPI_5 SCK	Slave	SPI5_SCK_I	PA_09	9	0	1	X	494	010b
	Master	SPI5_SCK_O			1	0	011b		X
	Slave	SPI5_SCK_I	PK_15	175	0	1	X		100b
	Master	SPI5_SCK_O			1	0	100b		X
	Slave	SPI5_SCK_I	PK_12	172	0	1	X		011b

23.7.2.6 FlexCAN clocking



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46f-4821-4848-b251-39f0da06240d

23.7.2.7 FlexRay communication controller clocking



23.7.2.8 I²C clocking

Figure 82 shows I²C_*n* clocking and Figure 83 shows the I²C SIUL2 configuration.

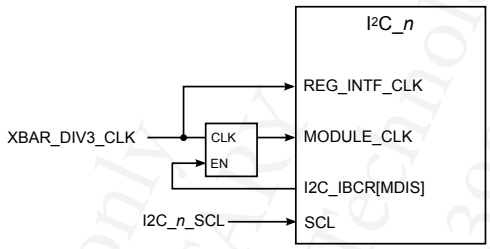


Figure 82. I²C_*n* clocking

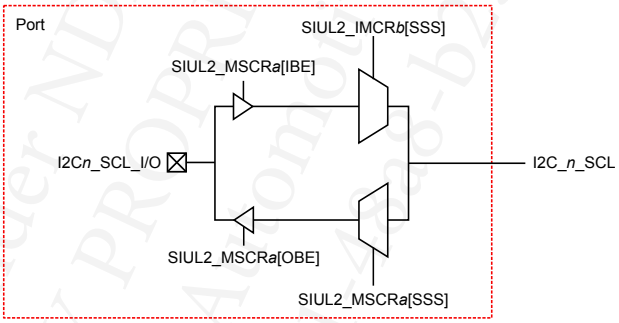


Figure 83. I²C_*n* SIUL2 clock configuration

Table 93. SIUL2 options for I2C*n*_SCL_I/O clock

I²C_*n* signal	I²C mode	I/O signal	Port	MSCR*a*	MSCR fields			IMCR*b*	IMCR[SSS]
					OBE	IBE	SSS		
I2C_0_SCL	Slave	I2C0_SCL_I	PB_01	17	0	1	X	54	010b
	Master	I2C0_SCL_O			1	0	001b		X
	Slave	I2C0_SCL_I	PC_00	32	0	1	X		011b
	Master	I2C0_SCL_O			1	0	001b		X
	Slave	I2C0_SCL_I	PD_07	55	0	1	X		100b

Table continues on the next page...

Table 93. SIUL2 options for I2C*n*_SCL_I/O clock (continued)

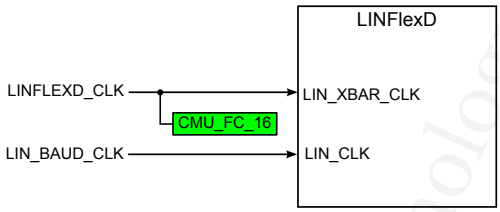
I²C_*n* signal	I²C mode	I/O signal	Port	MSCR*a*	MSCR fields			IMCR*b*	IMCR[SSS]
					OBE	IBE	SSS		
	Master	I2C0_SCL_O			1	0	011b		X
	Slave	I2C0_SCL_I	PJ_00	144	0	1	X		101b
	Master	I2C0_SCL_O			1	0	011b		X
	Slave	I2C0_SCL_I	PJ_00	15	0	1	X		110b
	Master	I2C0_SCL_O			1	0	011b		X
I2C_1_SCL	Slave	I2C1_SCL_I	PB_03	19	0	1	X	205	010b
	Master	I2C1_SCL_O			1	0	001b		X
	Slave	I2C1_SCL_I	PD_01	49	0	1	X		011b
	Master	I2C1_SCL_O			1	0	001b		X
	Slave	I2C1_SCL_I	PH_02	114	0	1	X		100b
	Master	I2C1_SCL_O			1	0	100b		X
	Slave	I2C1_SCL_I	PK_00	163	0	1	X		101b
	Master	I2C1_SCL_O			1	0	011b		X
	Slave	I2C1_SCL_I	PK_15	175	0	1	X		110b
	Master	I2C1_SCL_O			1	0	101b		X
I2C_2_SCL	Slave	I2C2_SCL_I	PB_05	21	0	1	X	207	010b
	Master	I2C2_SCL_O			1	0	001b		X
	Slave	I2C2_SCL_I	PD_00	48	0	1	X		011b
	Master	I2C2_SCL_O			1	0	011b		X
	Slave	I2C2_SCL_I	PH_08	120	0	1	X		100b
	Master	I2C2_SCL_O			1	0	001b		X
I2C_3_SCL	Slave	I2C3_SCL_I	PB_07	23	0	1	X	209	010b
	Master	I2C3_SCL_O			1	0	001b		X
	Slave	I2C3_SCL_I	PB_14	30	0	1	X		011b
	Master	I2C3_SCL_O			1	0	100b		X
	Slave	I2C3_SCL_I	PD_03	51	0	1	X		100b
	Master	I2C3_SCL_O			1	0	011b		X
	Slave	I2C3_SCL_I	PE_15	79	0	1	X		101b
	Master	I2C3_SCL_O			1	0	011b		X
I2C_4_SCL	Slave	I2C4_SCL_I	PB_01	17	0	1	X	211	010b

Table continues on the next page...

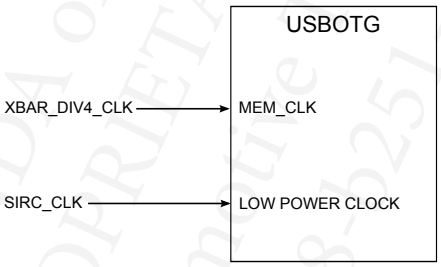
Table 93. SIUL2 options for I2C*n*_SCL_I/O clock (continued)

I²C_*n* signal	I²C mode	I/O signal	Port	MSCR*a*	MSCR fields			IMCR*b*	IMCR[SSS]
					OBE	IBE	SSS		
	Master	I2C4_SCL_O			1	0	100b		X
	Slave	I2C4_SCL_I	PC_02	34	0	1	X		011b
	Master	I2C4_SCL_O			1	0	010b		X

23.7.2.9 LINFlexD clocking



23.7.2.10 Universal serial bus (USB) clocking



23.7.3 Timer modules

Figure 84 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 94 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK, uses the same source for REG_INTF_CLK as used by the MODULE_CLK.

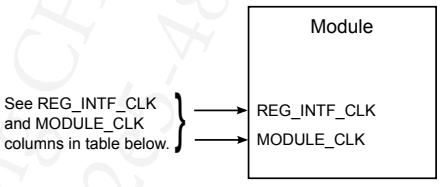


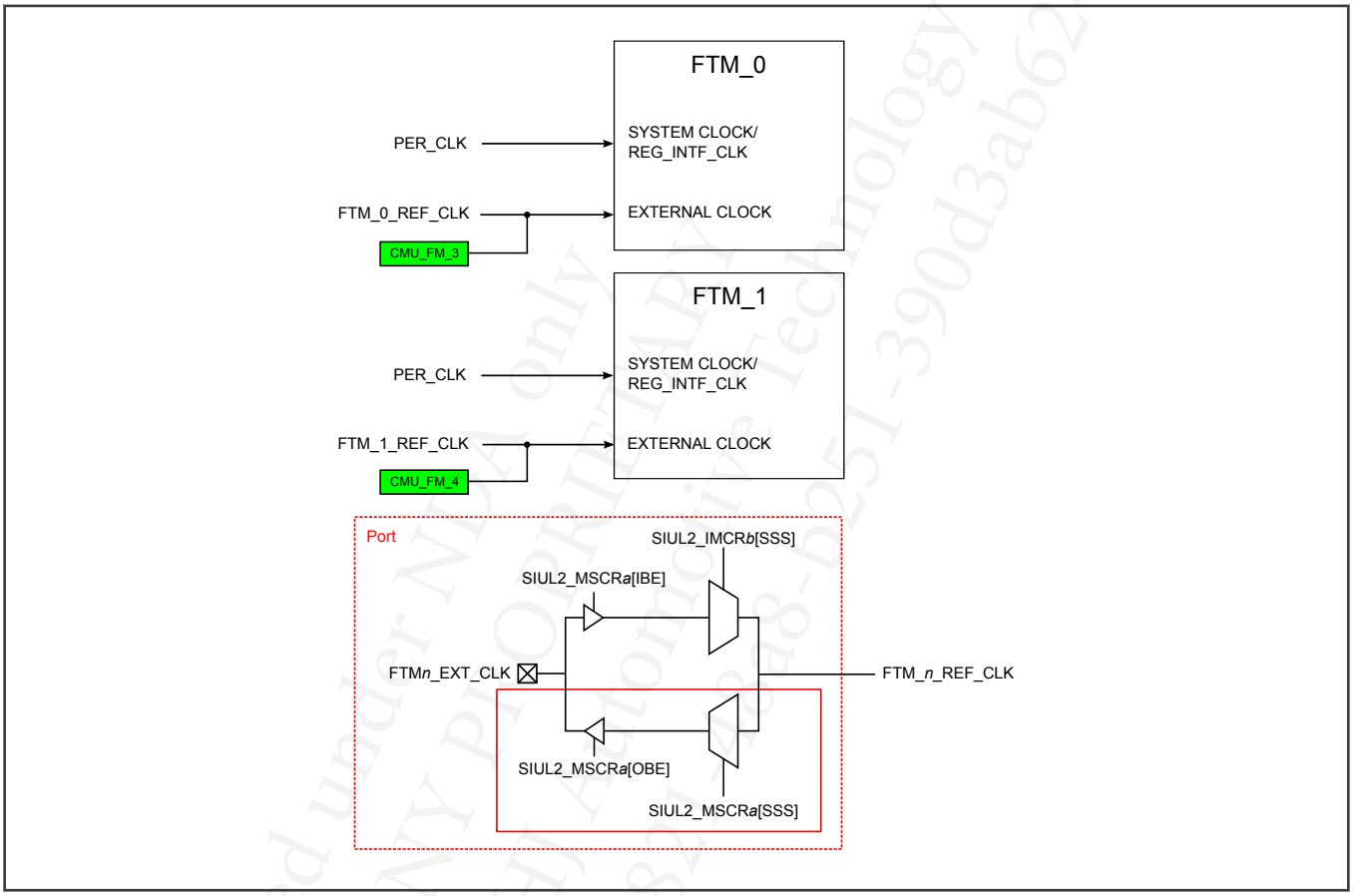
Figure 84. Timer module clocks

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHI Automotive Technology Co., Ltd.
c8c4620c-4824-4888-b251-390d3ab624c1

Table 94. Timer module clocking

Module	MODULE_CLK	REG_INTF_CLK
FTM	See FTM clocking	
PIT	XBAR_DIV3_CLK	
RTC	See RTC clocking	
STM	XBAR_DIV3_CLK	
SWT	See SWT clocking	

23.7.3.1 FTM clocking



NOTE

Output mux and output buffer in solid lined red box does not apply to port PJ_08.

Table 95. SIUL2 options for FTM_*n*_REF_CLK clock

FTM_*n*_REF_CLK signal	FTM_*n*_SC[CLKS]	I/O Signal	Port	MSCR*a*	MSCR fields		IMCR*b*	IMCR[SSS]
					OBE	IBE		
FTM_0_REF_CLK	11b	FTM0_EXT_CLK	PA_13	13	0	1	661	010b
	!(11b)				X	X		X

Table continues on the next page...

Table 95. SIUL2 options for FTM_*n*_REF_CLK clock (continued)

FTM_*n*_REF_CLK signal	FTM_*n*_SC[CLKS]	I/O Signal	Port	MSCR*a*	MSCR fields		IMCR*b*	IMCR[SSS]
					OBE	IBE		
	11b		PH_04	116	0	1		010b
	!(11b)				X	X		X
	11b		PJ_08	—	—	—		100b
	!(11b)							X
FTM_1_REF_CLK	11b	FTM1_EXT_CLK	PB_11	27	0	1	671	010b
	!(11b)				X	X		X
	11b		PD_15	63	0	1		011b
	!(11b)				X	X		X

23.7.3.2 RTC clocking

Figure 85 shows the RTC clock configuration and Figure 86 shows the SIUL2 port configuration for the external RTC clock.

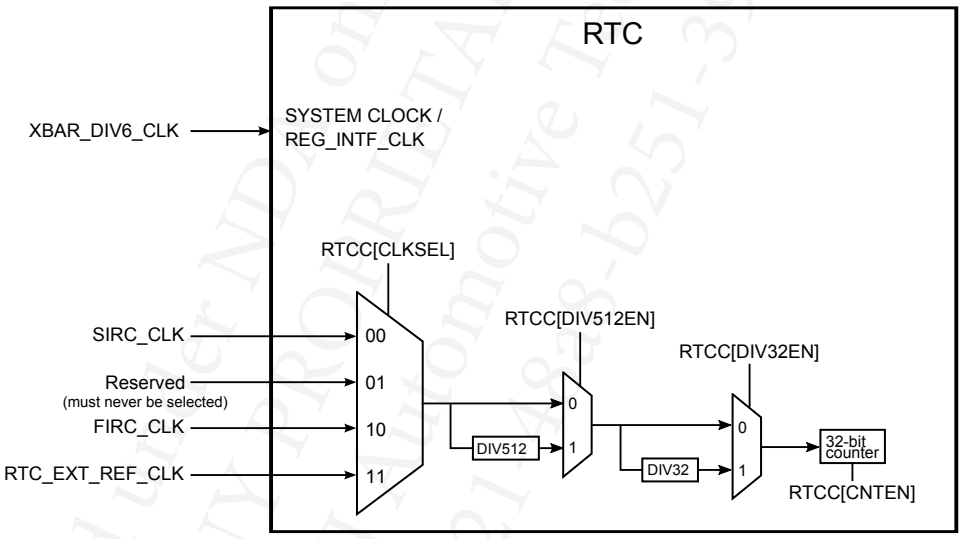


Figure 85. RTC clock configuration

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4828-b251-390d3ab6f4d1

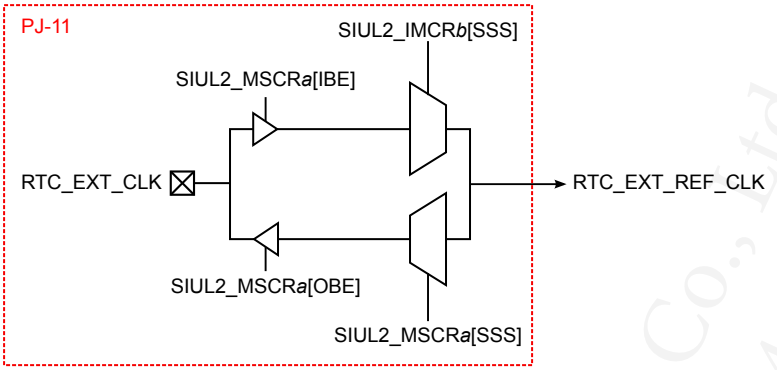


Figure 86. SIUL2 port configuration for RTC_EXT_REF_CLK

WARNING

RTCC[CLKSEL] must never equal 01b.

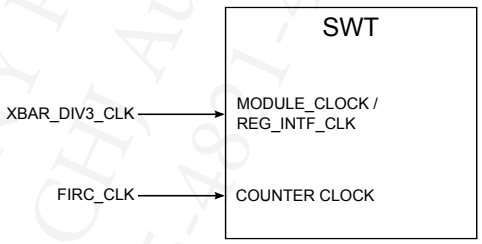
Table 96. SIUL2 options for RTC_EXT_REF_CLK

RTC_RTCC[CLKSEL]	Port	MSCR*a*	MSCR fields		IMCR*b*	IMCR[SSS]
			OBE	IBE		
11b	PJ_11	155	0	1	979	010b
!(11b)			X	X		X

NOTE

When in Standby mode, SIRC_CLK is the only available clock. RTC can only use SIRC_CLK or FIRC_CLK as clock sources if you want to preserve RTC contents across functional resets. The chip reconfigures the RTC_EXT_REF_CLK pin multiplexing on reset.

23.7.3.3 SWT clocking



23.7.4 Generic modules

Figure 87 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 97 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK, uses the same source for REG_INTF_CLK as used by the MODULE_CLK.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8-6285-4991-48a8-251390d3ab624c1

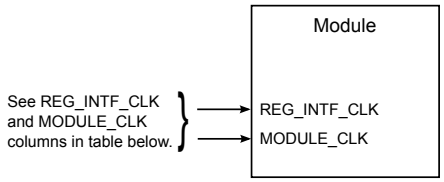
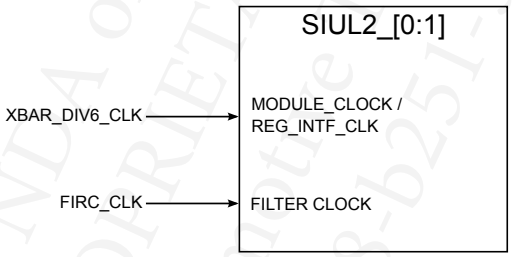


Figure 87. Generic module clocks

Table 97. Generic module clocking

Module	MODULE_CLK	REG_INTF_CLK
WKPU	XBAR_DIV6_CLK	
SRC	XBAR_DIV6_CLK	
SRC_TOP	XBAR_DIV6_CLK	
SIUL2_[0:1]	See SIUL2 clocking	
TMU	XBAR_DIV3_CLK	

23.7.4.1 SIUL2 clocking



23.7.5 Analog modules

Figure 88 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 98 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK, uses the same source for REG_INTF_CLK as used by the MODULE_CLK.

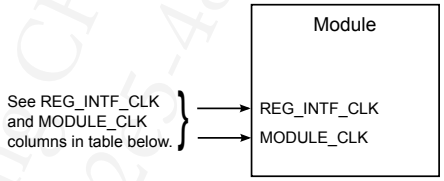


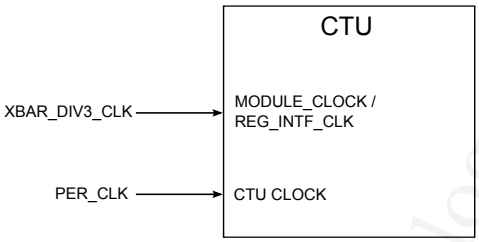
Figure 88. Analog module clocks

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHIPSEA Automotive Technology Co., Ltd.
c8c4626-482-48a8-b251-390d3ab624c1

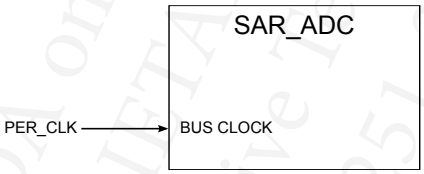
Table 98. Analog module clocking

Module	MODULE_CLK	REG_INTF_CLK
CTU	See CTU clocking	
SAR_ADC	See SAR_ADC clocking	

23.7.5.1 CTU clocking



23.7.5.2 SAR_ADC clocking



23.7.6 Debug modules

Please see the debug chapter for details.

23.7.6.1 Debug trace unit clocking

The following diagram shows an overview of Debug trace clocking in the Accelerator subsystem.

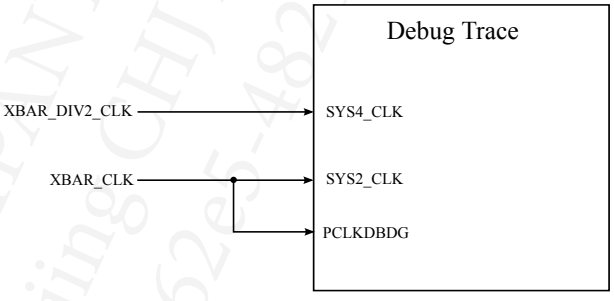


Figure 89. Debug trace clocking overview

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8ab624c1

23.7.7 Memory-controller modules

Figure 90 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 99 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK, uses the same source for REG_INTF_CLK as used by the MODULE_CLK.

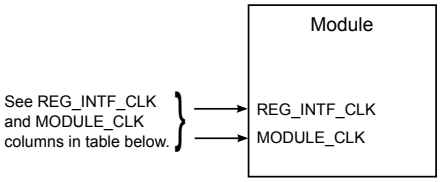
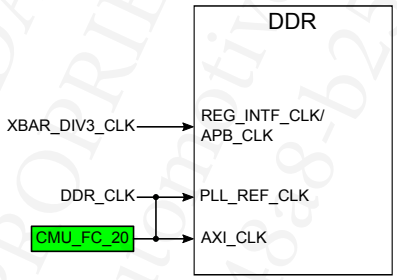


Figure 90. Memory-controller module clocks

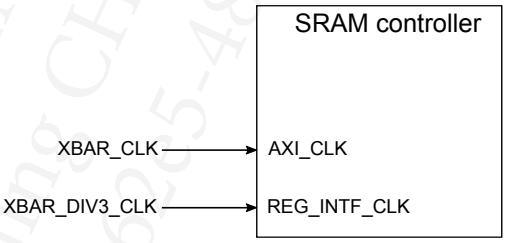
Table 99. Memory-controller module clocking

Module	MODULE_CLK	REG_INTF_CLK
DDR	See DDR controller clocking	
SRAM controller	See SRAM controller clocking	
uSDHC	See uSDHC clocking	
QuadSPI	See QuadSPI clocking	

23.7.7.1 DDR controller clocking

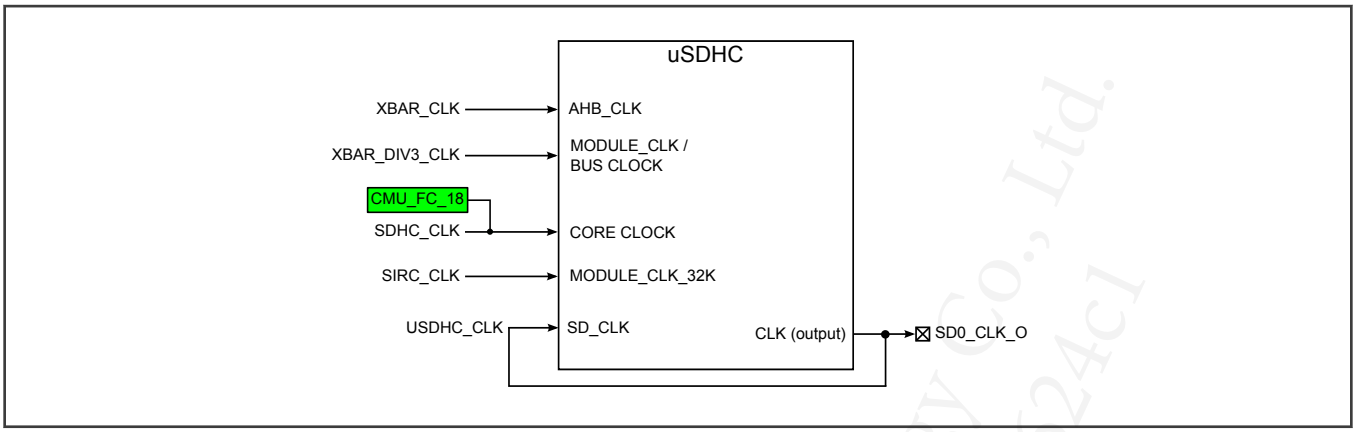


23.7.7.2 SRAM controller clocking

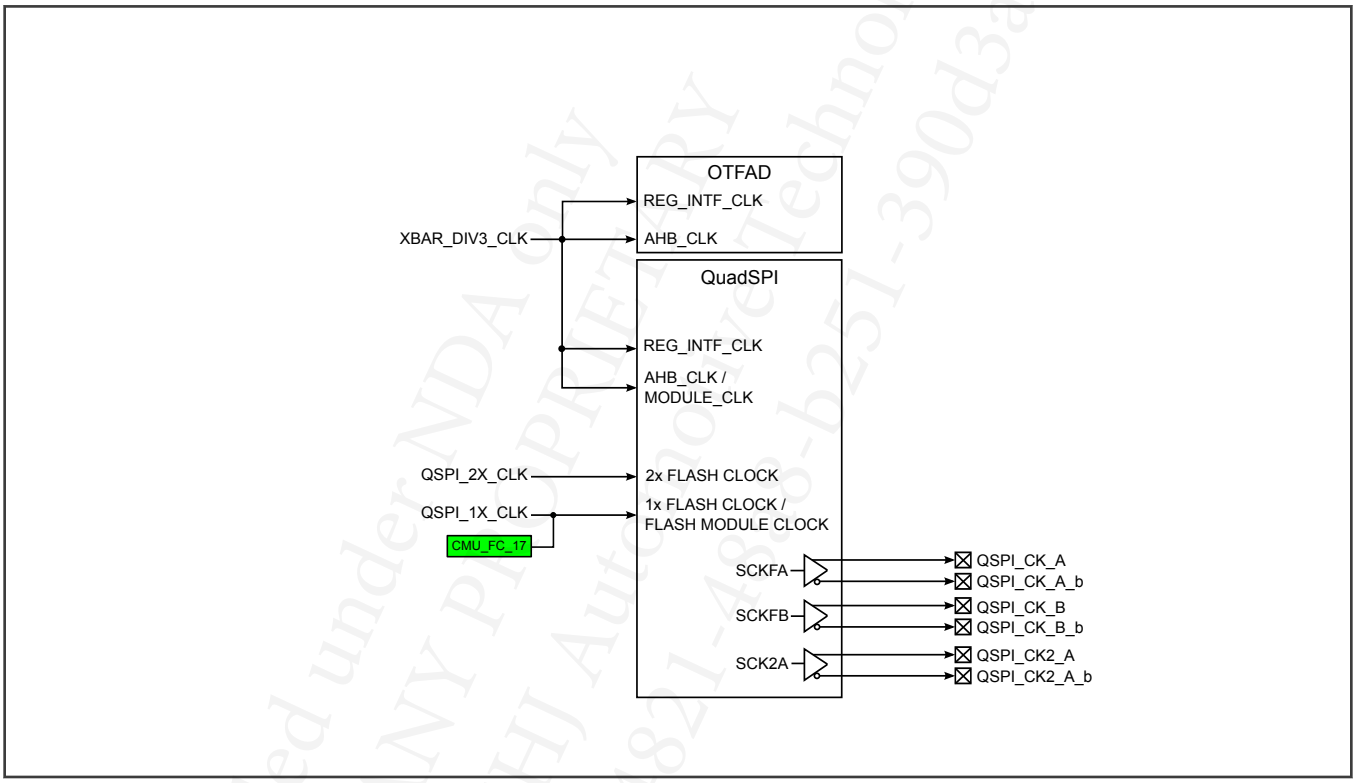


Provided under NDA of COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c4625-4821-4a8-b251-390d3ab624c1

23.7.7.3 uSDHC clocking



23.7.7.4 QuadSPI clocking



23.7.8 Security modules

Figure 91 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 100 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK, uses the same source for REG_INTF_CLK as used by the MODULE_CLK.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4675-4821-4886-b251-390d3ab624c1

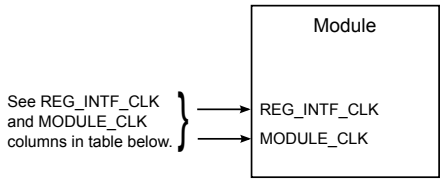


Figure 91. Security module clocks

Table 100. Security module clocking

Module	MODULE_CLK	REG_INTF_CLK
Hardware Security Engine (HSE_H)	See Hardware Security Engine (HSE_H)	
OCOTP	FIRC_CLK	

23.7.8.1 Hardware Security Engine (HSE_H)

The following figure shows the HSE_H clocking configuration.

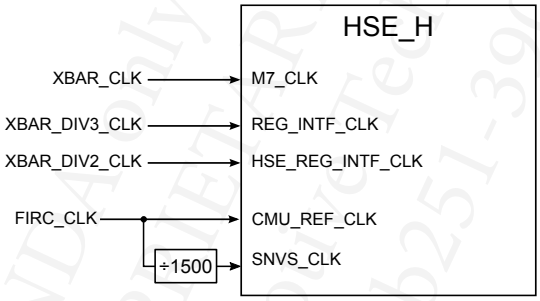


Figure 92. HSE_H clocking

23.7.9 Clocking modules

Figure 93 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 101 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK, uses the same source for REG_INTF_CLK as used by the MODULE_CLK.

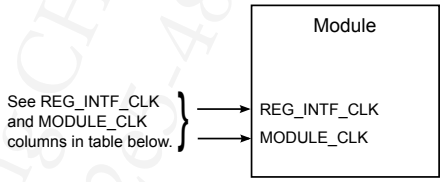


Figure 93. Clocking module clocks

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHTC Automotive Technology Co., Ltd.
c8c4626e-4824-48a8-b251-390d3ab624c1

Table 101. Clocking module clocking

Module	MODULE_CLK	REG_INTF_CLK
CMU	XBAR_DIV6_CLK (See Clock input sources for reference and monitored clock details.)	
DFS	XBAR_DIV6_CLK	
FIRC	See FIRC for details.	
MC_CGM	XBAR_DIV6_CLK	
PLLDIG	XBAR_DIV6_CLK	
SIRC	See SIRC for details.	
FXOSC	XBAR_DIV6_CLK (see FXOSC for details)	

23.7.10 Reset modules

Figure 94 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 102 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK, uses the same source for REG_INTF_CLK as used by the MODULE_CLK.

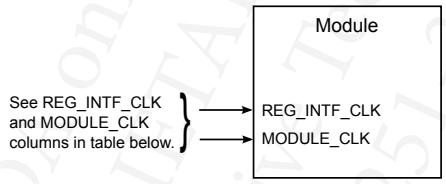
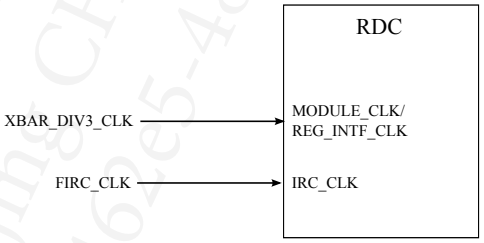


Figure 94. Reset module clocks

Table 102. Reset module clocking

Module	MODULE_CLK	REG_INTF_CLK
MC_RGM	FIRC_CLK	
RDC	See RDC clocking	

23.7.10.1 RDC clocking



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4848-b251-390cab624c1

23.7.11 Power-modes modules

Figure 95 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 103 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK, uses the same source for REG_INTF_CLK as used by the MODULE_CLK.

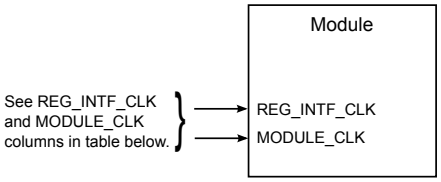
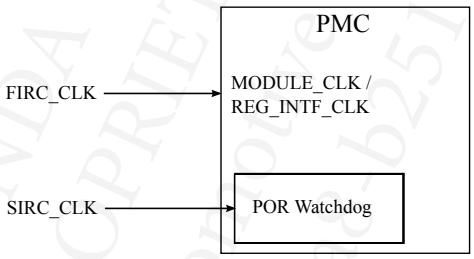


Figure 95. Power module clocks

Table 103. Power module clocking

Module	MODULE_CLK	REG_INTF_CLK
MC_ME	XBAR_DIV6_CLK	
PMC	See PMC clocking	
MC_PCU	SIRC_CLK	—

23.7.11.1 PMC clocking



23.7.12 Core platform modules

Figure 96 shows the REG_INTF_CLK and MODULE_CLK connections, and Table 104 shows the REG_INTF_CLK and MODULE_CLK signals used by these modules. Any module diagram that does not explicitly show a REG_INTF_CLK, uses the same source for REG_INTF_CLK as used by the MODULE_CLK.

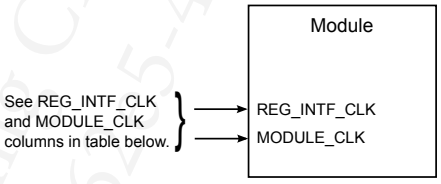


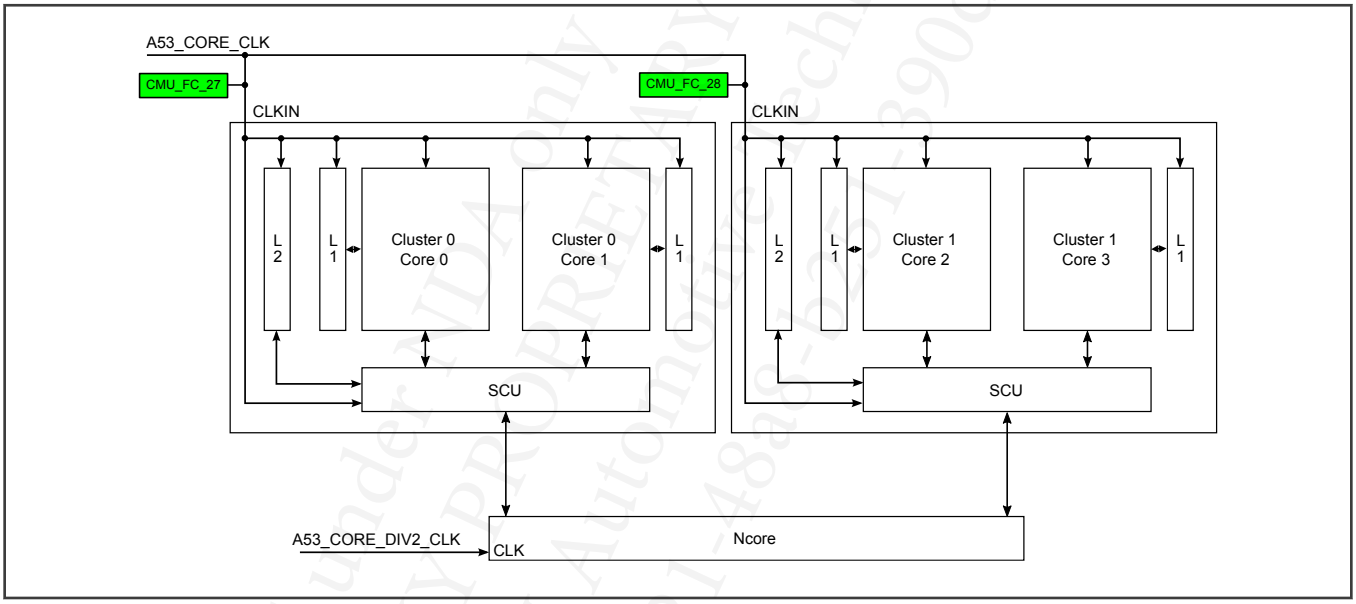
Figure 96. Core platform module clocks

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHANGAN Automotive Technology Co., Ltd.
c8c4625-46a-484c-b251-390d3ab624c1

Table 104. Core platform module clocking

Module	MODULE_CLK	REG_INTF_CLK
Cortex-A53 cluster	See Cortex-A53 cluster clocking	
Cortex-M7	See Cortex-M7 clocking	
DMAMUX	XBAR_DIV3_CLK	
eDMA	See eDMA clocking	
Arm generic interrupt controller (GIC500)	See Arm generic interrupt controller (GIC500) clocking for details.	
MSCM	XBAR_DIV3_CLK	
SEMA42	XBAR_DIV3_CLK	
XRDC_0	XBAR_DIV6_CLK	
XRDC_1	XBAR_DIV6_CLK	

23.7.12.1 Cortex-A53 cluster clocking

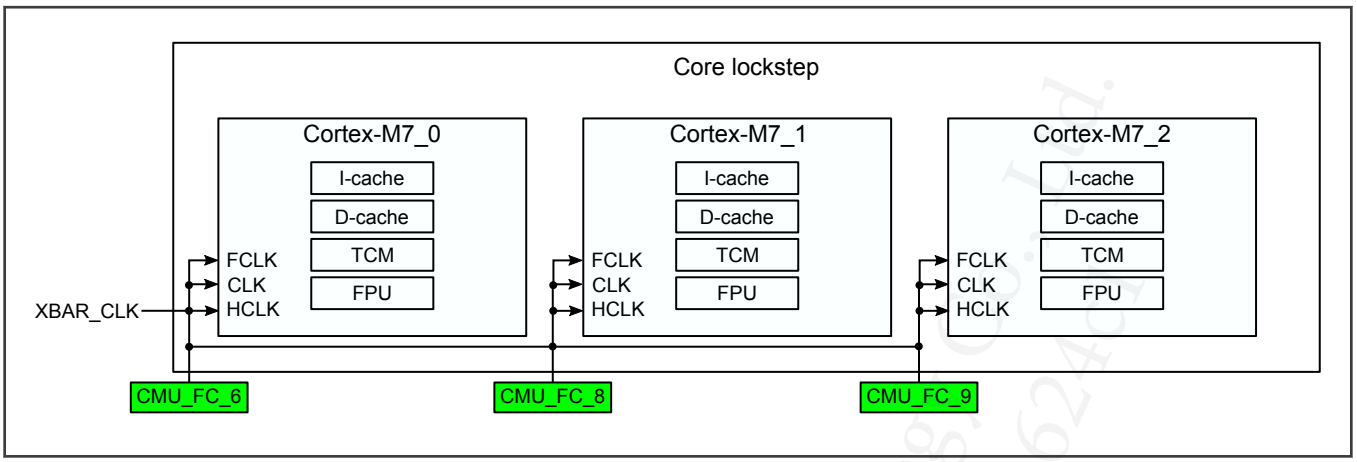


NOTE

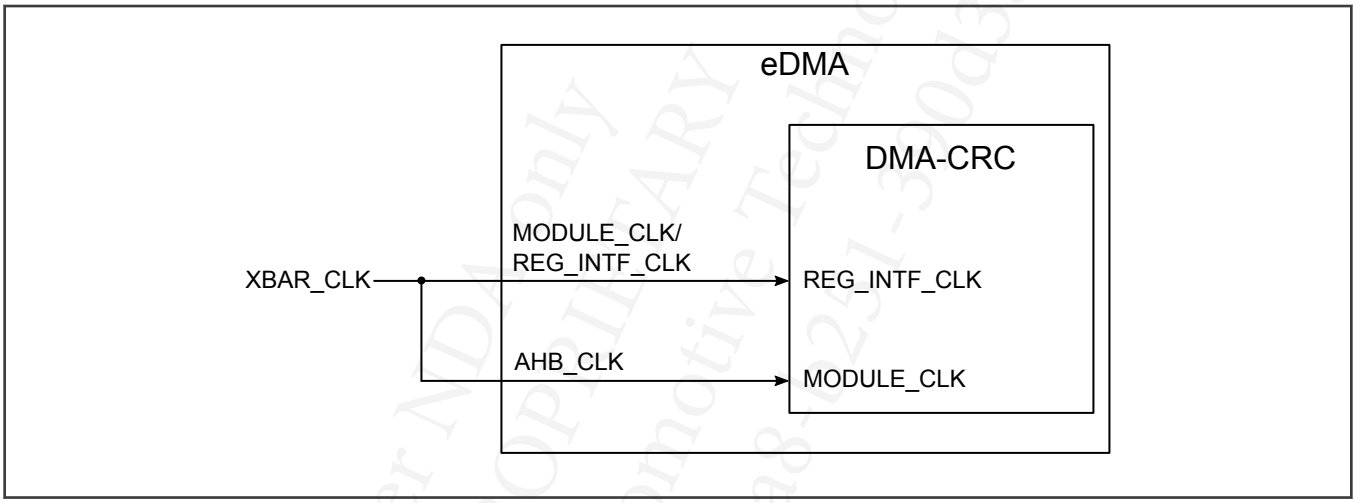
Generic timer uses FXOSC_CLK.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CLN Automotive Technology Co., Ltd.
c8c462e5-4e21-4848-8251-390d3ab624c4

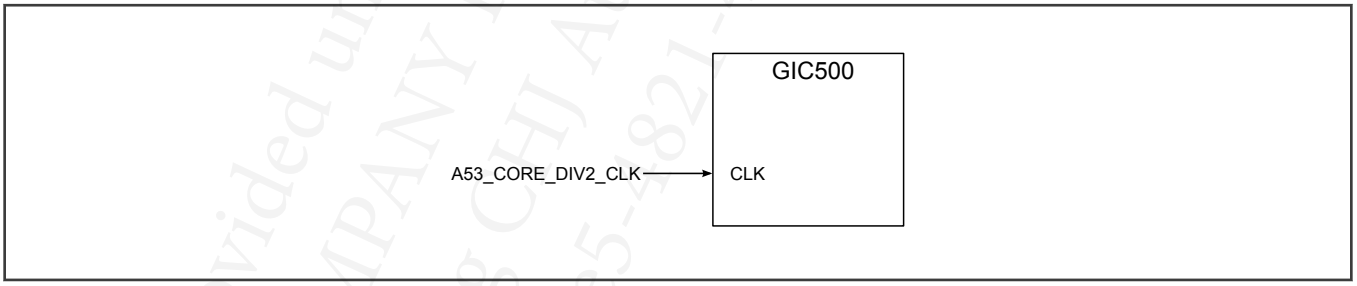
23.7.12.2 Cortex-M7 clocking



23.7.12.3 eDMA clocking



23.7.12.4 Arm generic interrupt controller (GIC500) clocking



23.8 Clock monitoring

23.8.1 Introduction

For all safety critical clocks, the chip must detect a missing clock or incorrect frequency. The use of CMUs achieve this result. The reference clock used depends on the CMU instance (either FIRC_CLK or FXOSC_CLK). For detailed information on CMUs, see the "Clock monitor unit (CMU)" chapter.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cc462e5-4821-48a8-8251-3a0d3ab624c4

NOTE

If the FIRC is not operating within operating parameters, recovery must be done with a POR. The methods used to trigger the POR are as follows:

- The software executing on the master safety core detects the FIRC that is not operating correctly (based on the CMU_FM metering FIRC_CLK with FXOSC_CLK as reference clock). Then it sends a POR command to the external PMIC via SPI (assuming that the software is able to issue the reset).
- If the software stops functioning correctly because of FIRC failure, or if it stops PMIC servicing, the PMIC watchdog timer expires (within the FTTI) and a POR asserts.

NOTE

Configure peripheral clocks before enabling peripherals. To verify that the peripheral clocks are operating within the expected parameters, check CMUs corresponding to the peripheral clocks. If a CMU shows an incorrect frequency, despite correct configuration, software should issue a destructive reset to correct the problem.

23.8.2 Clock input sources

The table below shows the clocks monitored by each CMU. These signals are connected internally on the chip, but are not accessible via pins on the boundary of the chip. The second column of the table shows the address offset for the corresponding CMU instance.

NOTE

All "CMU instances" shown in paretheses are for future use and their associated register space is reserved.

Table 105. Clock input sources

CMU instance	Address offset	Reference clock	Monitored clock	Error response
CMU_FC_0	0000h	FIRC_CLK	FXOSC_CLK	FXOSC_FAIL (MC_RGM destructive reset)
CMU_FM_1	0020h	FXOSC_CLK	FIRC_CLK	—
CMU_FM_2	0040h	FXOSC_CLK	SIRC_CLK	—
CMU_FM_3	0060h	FXOSC_CLK	FTM_0_REF_CLK	—
CMU_FM_4	0080h	FXOSC_CLK	FTM_1_REF_CLK	—
CMU_FC_5	00A0h	FIRC_CLK	XBAR_DIV3_CLK	XBAR_DIV3_CLK_FAIL (MC_RGM destructive reset)
CMU_FC_6	00C0h	FIRC_CLK	XBAR_CLK (M7_0)	CMU_REPORT_0 (FCCU NCF)
CMU_FC_7	00E0h	FXOSC_CLK	XBAR_DIV3_CLK	CMU_REPORT_0 (FCCU NCF)
CMU_FC_8	0100h	FIRC_CLK	XBAR_CLK (M7_1)	CMU_REPORT_0 (FCCU NCF)
CMU_FC_9	0120h	FIRC_CLK	XBAR_CLK (M7_2)	CMU_REPORT_0 (FCCU NCF)
CMU_FC_10	0140h	FIRC_CLK	PER_CLK	CMU_REPORT_0 (FCCU NCF)
CMU_FC_11	0160h	FXOSC_CLK	SERDES_REF_CLK	CMU_REPORT_0 (FCCU NCF)
CMU_FC_12	0180h	FXOSC_CLK	FLEXRAY_PE_CLK	CMU_REPORT_0 (FCCU NCF)
CMU_FC_13	01A0h	FXOSC_CLK	CAN_PE_CLK	CMU_REPORT_0 (FCCU NCF)
CMU_FC_14	01C0h	FXOSC_CLK	GMAC_0_TX_CLK	CMU_REPORT_0 (FCCU NCF)

Table continues on the next page...

Table 105. Clock input sources (continued)

CMU instance	Address offset	Reference clock	Monitored clock	Error response
CMU_FC_15	01E0h	FXOSC_CLK	GMAC_TS_CLK	CMU_REPORT_0 (FCCU NCF)
CMU_FC_16	0200h	FXOSC_CLK	LINFLEXD_CLK	CMU_REPORT_0 (FCCU NCF)
CMU_FC_17	0220h	FXOSC_CLK	QSPI_1X_CLK	CMU_REPORT_0 (FCCU NCF)
CMU_FC_18	0240h	FXOSC_CLK	SDHC_CLK	CMU_REPORT_0 (FCCU NCF)
(19)	0260h	Reserved		
CMU_FC_20	0280h	FIRC_CLK	DDR_CLK	CMU_REPORT_0 (FCCU NCF)
CMU_FC_21	02A0h	FXOSC_CLK	GMAC_0_RX_CLK	CMU_REPORT_0 (FCCU NCF)
CMU_FC_22	02C0h	FXOSC_CLK	SPI_CLK	CMU_REPORT_0 (FCCU NCF)
(23)	02E0h	Reserved		
(24)	0300h	Reserved		
(25)	0320h	Reserved		
(26)	0340h	Reserved		
CMU_FC_27	0360h	FXOSC_CLK	A53_CORE_CLK (cluster 0)	CMU_REPORT_1 (FCCU NCF)
CMU_FC_28	0380h	FIRC_CLK	A53_CORE_CLK (cluster 1)	CMU_REPORT_1 (FCCU NCF)
(29)	03A0h	Reserved		
(30)	03C0h	Reserved		
(31)	03E0h	Reserved		
(32)	0400h	Reserved		
(33)	0420h	Reserved		
(34)	0440h	Reserved		
(35)	0460h	Reserved		
(36)	0480h	Reserved		
(37)	04A0h	Reserved		
(38)	04C0h	Reserved		
CMU_FC_39	04E0h	FXOSC_CLK	PFE_SYS_CLK	CMU_REPORT_3 (FCCU NCF)
(40)	0500h	Reserved		
(41)	0520h	Reserved		
(42)	0540h	Reserved		
(43)	0560h	Reserved		
(44)	0580h	Reserved		
(45)	05A0h	Reserved		
CMU_FC_46	05C0h	FXOSC_CLK	PFE_MAC_0_TX_DIV_CLK	CMU_REPORT_3 (FCCU NCF)

Table continues on the next page...

Table 105. Clock input sources (continued)

CMU instance	Address offset	Reference clock	Monitored clock	Error response
CMU_FC_47	05E0h	FXOSC_CLK	PFE_MAC_0_RX_CLK	CMU_REPORT_3 (FCCU NCF)
CMU_FC_48	0600h	FXOSC_CLK	PFE_MAC_1_TX_CLK	CMU_REPORT_3 (FCCU NCF)
CMU_FC_49	0620h	FXOSC_CLK	PFE_MAC_1_RX_CLK	CMU_REPORT_3 (FCCU NCF)
CMU_FC_50	0640h	FXOSC_CLK	PFE_MAC_2_TX_CLK	CMU_REPORT_3 (FCCU NCF)
CMU_FC_51	0660h	FXOSC_CLK	PFE_MAC_2_RX_CLK	CMU_REPORT_3 (FCCU NCF)
(52)	0680h	Reserved		
(53)	06A0h	Reserved		
(54)	06C0h	Reserved		

23.8.3 Enabling and disabling CMU_FC_0 and CMU_FC_5

CMU_FC_0 and CMU_FC_5 need to be enabled and running in order to properly detect loss of clock events.

- CMU_FC_0:
  - You must enable CMU_FC_0 as soon as the FXOSC has been turned on and is stable.
  - You must disable CMU_FC_0 just prior to turning off the FXOSC.
- CMU_FC_5:
  - You must enable CMU_FC_5 as soon as the XBAR_DIV3_CLK has been configured, that is, after source configuration and selection.
  - You must disable CMU_FC_5 just prior to reconfiguring the XBAR_DIV3_CLK, that is, prior to source reconfiguration or source selection change.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39003ab624d1

Chapter 24
Clock Generation Module (MC_CGM)

24.1 Chip-specific MC_CGM information

This chip has the following MC_CGM instances:

- MC_CGM_0 generates clocks for the system peripherals.
- MC_CGM_1 generates clocks for the Cortex-A53 cores.
- MC_CGM_2 generates clocks for PFE clocks.
- MC_CGM_5 generates clocks for the DDR peripheral.

Clock source numbers

In this chapter, each numbered clock source represented as clk_src_*n* refers to the clock source with the same index number in the "MC_CGM clock source mapping" table in the "Clock source mapping on MC_CGM clock selectors" section.

24.2 Chip-specific MC_CGM_2 information

24.2.1 MC_CGM feature configuration

MC_CGM instances on this chip do not implement Divider Trigger Control Registers or Divider Trigger Registers. Disregard their references in the MC_CGM_2 register descriptions.

24.3 Introduction

The clock generation module (MC_CGM) is used to set up the configurable clock domains used by various chip blocks as per the application needs. It includes the clock multiplexers that allow software to select the desired clock sources for these domains. This is managed by the MC_CGM to ensure that the changing of the clock selection from one source to another occurs in a glitch-less fashion. In addition, the MC_CGM includes the clock dividers that can be configured by software.

See following figure for the MC_CGM block diagram:

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-9151-590d3a9624c1

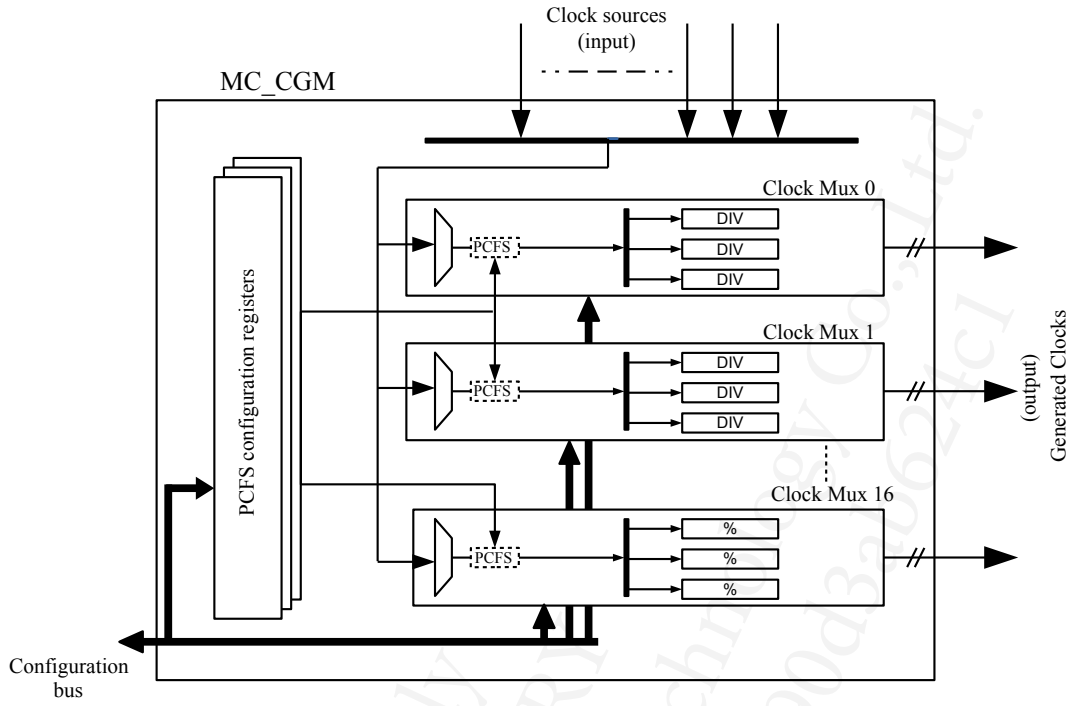


Figure 97. MC_CGM block diagram

NOTE

The block diagram is generic and does not necessarily reflect any specific MC_CGM implementation.

24.4 Features

MC_CGM includes the following features:

- Implements software configurable clock multiplexers for selecting from various clock sources
- Provides hardware-controlled multiplexers that guarantee glitchless transition, while the software-controlled multiplexers need a software sequence to ensure such a transition
- Provides software configurable automatic PCFS on certain clocks to minimize the impact of a sudden power consumption change through a gentle ramp-down and -up of the clock frequency when switching clock sources
- Implements software configurable clock dividers

24.5 MC_CGM_0 register descriptions

MC_CGM implements a set of clock multiplexers that share PCFS configuration registers. MC_CGM registers have the following properties:

- All registers are 32-bit wide.
- Only 32-bit read and write accesses are supported.
- Read/write accesses of less than 32 bits do not terminate with an error, and hardware behavior is unpredictable.
- Writes to read-only register fields in writable registers are ignored and do not provide an error response.
- Writes to read-only registers are aborted with an error response.

24.5.1 MC_CGM_0 memory map

MC_CGM_0 base address: 4003_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PCFS Step Duration (PCFS_SDUR)	32	RW	0000_0000h
88h	PCFS Divider Change 12 Register (PCFS_DIVC12)	32	RW	0000_0000h
8Ch	PCFS Divider End 12 Register (PCFS_DIVE12)	32	RW	0000_03E7h
90h	PCFS Divider Start 12 Register (PCFS_DIVS12)	32	RW	0000_03E7h
300h	Clock Mux 0 Select Control Register (MUX_0_CSC)	32	RW	0000_0000h
304h	Clock Mux 0 Select Status Register (MUX_0_CSS)	32	RO	0008_0000h
308h	Clock Mux 0 Divider 0 Control Register (MUX_0_DC_0)	32	RW	0000_0000h
30Ch	Clock Mux 0 Divider 1 Control Register (MUX_0_DC_1)	32	RW	8005_0000h
33Ch	Clock Mux 0 Divider Update Status Register (MUX_0_DIV_UPD_STAT)	32	RO	0000_0000h
340h	Clock Mux 1 Select Control Register (MUX_1_CSC)	32	RW	0200_0000h
344h	Clock Mux 1 Select Status Register (MUX_1_CSS)	32	RO	0202_0000h
348h	Clock Mux 1 Divider 0 Control Register (MUX_1_DC_0)	32	RW	0000_0000h
37Ch	Clock Mux 1 Divider Update Status Register (MUX_1_DIV_UPD_STAT)	32	RO	0000_0000h
380h	Clock Mux 2 Select Control Register (MUX_2_CSC)	32	RW	0200_0000h
384h	Clock Mux 2 Select Status Register (MUX_2_CSS)	32	RO	0202_0000h
388h	Clock Mux 2 Divider 0 Control Register (MUX_2_DC_0)	32	RW	0000_0000h
3BCh	Clock Mux 2 Divider Update Status Register (MUX_2_DIV_UPD_STAT)	32	RO	0000_0000h
3C0h	Clock Mux 3 Select Control Register (MUX_3_CSC)	32	RW	0000_0000h
3C4h	Clock Mux 3 Select Status Register (MUX_3_CSS)	32	RO	0008_0000h
3C8h	Clock Mux 3 Divider 0 Control Register (MUX_3_DC_0)	32	RW	0000_0000h
3FCh	Clock Mux 3 Divider Update Status Register (MUX_3_DIV_UPD_STAT)	32	RO	0000_0000h
400h	Clock Mux 4 Select Control Register (MUX_4_CSC)	32	RW	0000_0000h
404h	Clock Mux 4 Select Status Register (MUX_4_CSS)	32	RO	0008_0000h
408h	Clock Mux 4 Divider 0 Control Register (MUX_4_DC_0)	32	RW	0000_0000h
43Ch	Clock Mux 4 Divider Update Status Register (MUX_4_DIV_UPD_STAT)	32	RO	0000_0000h
440h	Clock Mux 5 Select Control Register (MUX_5_CSC)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
444h	Clock Mux 5 Select Status Register (MUX_5_CSS)	32	RO	0008_0000h
448h	Clock Mux 5 Divider 0 Control Register (MUX_5_DC_0)	32	RW	0000_0000h
47Ch	Clock Mux 5 Divider Update Status Register (MUX_5_DIV_UPD_STAT)	32	RO	0000_0000h
480h	Clock Mux 6 Select Control Register (MUX_6_CSC)	32	RW	0000_0000h
484h	Clock Mux 6 Select Status Register (MUX_6_CSS)	32	RO	0008_0000h
488h	Clock Mux 6 Divider 0 Control Register (MUX_6_DC_0)	32	RW	0000_0000h
4BCh	Clock Mux 6 Divider Update Status Register (MUX_6_DIV_UPD_STAT)	32	RO	0000_0000h
4C0h	Clock Mux 7 Select Control Register (MUX_7_CSC)	32	RW	0000_0000h
4C4h	Clock Mux 7 Select Status Register (MUX_7_CSS)	32	RO	0008_0000h
500h	Clock Mux 8 Select Control Register (MUX_8_CSC)	32	RW	0000_0000h
504h	Clock Mux 8 Select Status Register (MUX_8_CSS)	32	RO	0008_0000h
540h	Clock Mux 9 Select Control Register (MUX_9_CSC)	32	RW	0000_0000h
544h	Clock Mux 9 Select Status Register (MUX_9_CSS)	32	RO	0008_0000h
548h	Clock Mux 9 Divider 0 Control Register (MUX_9_DC_0)	32	RW	0000_0000h
57Ch	Clock Mux 9 Divider Update Status Register (MUX_9_DIV_UPD_STAT)	32	RO	0000_0000h
580h	Clock Mux 10 Select Control Register (MUX_10_CSC)	32	RW	0000_0000h
584h	Clock Mux 10 Select Status Register (MUX_10_CSS)	32	RO	0008_0000h
588h	Clock Mux 10 Divider 0 Control Register (MUX_10_DC_0)	32	RW	0000_0000h
5BCh	Clock Mux 10 Divider Update Status Register (MUX_10_DIV_UPD_STAT)	32	RO	0000_0000h
5C0h	Clock Mux 11 Select Control Register (MUX_11_CSC)	32	RW	0000_0000h
5C4h	Clock Mux 11 Select Status Register (MUX_11_CSS)	32	RO	0008_0000h
600h	Clock Mux 12 Select Control Register (MUX_12_CSC)	32	RW	0000_0000h
604h	Clock Mux 12 Select Status Register (MUX_12_CSS)	32	RO	0008_0000h
608h	Clock Mux 12 Divider 0 Control Register (MUX_12_DC_0)	32	RW	0000_0000h
63Ch	Clock Mux 12 Divider Update Status Register (MUX_12_DIV_UPD_STAT)	32	RO	0000_0000h
680h	Clock Mux 14 Select Control Register (MUX_14_CSC)	32	RW	0000_0000h
684h	Clock Mux 14 Select Status Register (MUX_14_CSS)	32	RO	0008_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
688h	Clock Mux 14 Divider 0 Control Register (MUX_14_DC_0)	32	RW	0000_0000h
6BCh	Clock Mux 14 Divider Update Status Register (MUX_14_DIV_UPD_STAT)	32	RO	0000_0000h
6C0h	Clock Mux 15 Select Control Register (MUX_15_CSC)	32	RW	0000_0000h
6C4h	Clock Mux 15 Select Status Register (MUX_15_CSS)	32	RO	0008_0000h
6C8h	Clock Mux 15 Divider 0 Control Register (MUX_15_DC_0)	32	RW	0000_0000h
6FCh	Clock Mux 15 Divider Update Status Register (MUX_15_DIV_UPD_STAT)	32	RO	0000_0000h
700h	Clock Mux 16 Select Control Register (MUX_16_CSC)	32	RW	0000_0000h
704h	Clock Mux 16 Select Status Register (MUX_16_CSS)	32	RO	0008_0000h

24.5.2 PCFS Step Duration (PCFS_SDUR)

Offset

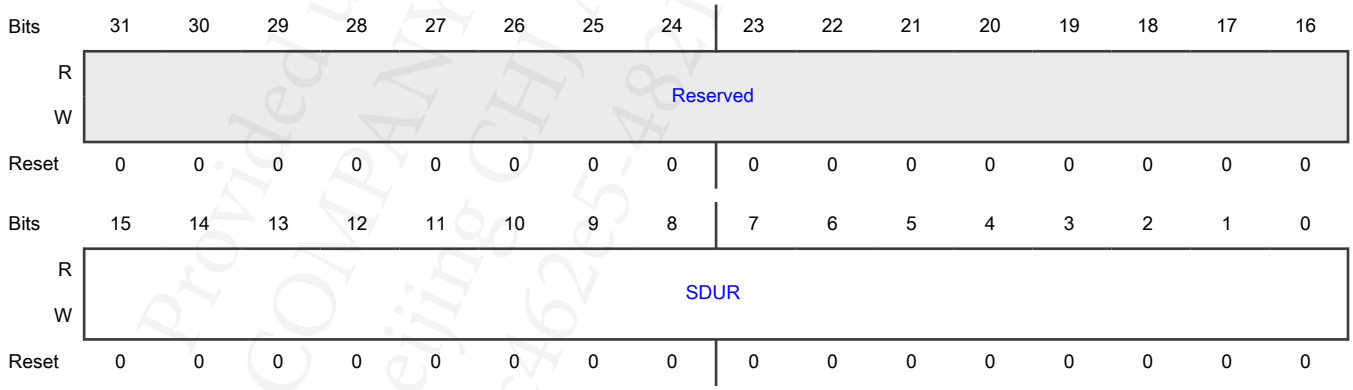
Register	Offset
PCFS_SDUR	0h

Function

This register specifies the step duration of each PCFS step. The value provided in this register specifies the PCFS step duration in terms of the number of cycles of clk_src_0.

This register is reset only by a destructive reset. For details, see PCFS.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c1625-4821-48a8-b251-390dab6242

Fields

Field	Function
31-16 —	This field is reserved and reads return zeros.
15-0 SDUR	Step duration Count value of the step duration

24.5.3 PCFS Divider Change 12 Register (PCFS_DIVC12)

Offset

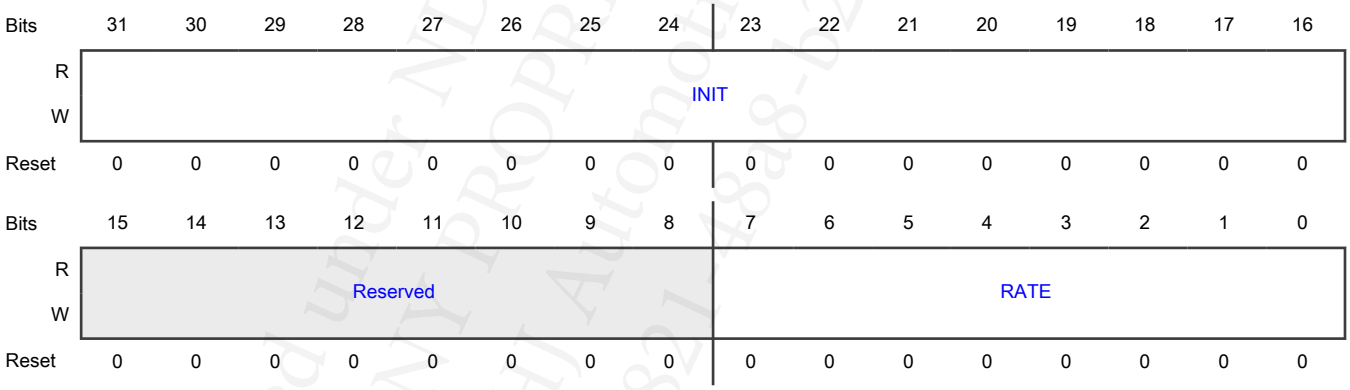
Register	Offset
PCFS_DIVC12	88h

Function

This register defines the rate of frequency change and initial change value on frequency ramp-up for the Progressive Clock Frequency switching of clk_src_12.

This register is reset only on destructive reset.

Diagram



Fields

Field	Function
31-16 INIT	Divider change initial value This field provides the initial change value of the clock divider for the clock ramp-up phase of clk_src_12.
15-8 —	This field is reserved and reads return zeros.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
7-0 RATE	Divider change rate This value controls the change value of the clock divider for the clock ramp-up and ramp-down phase of clk_src_12.

24.5.4 PCFS Divider End 12 Register (PCFS_DIVE12)

Offset

Register	Offset
PCFS_DIVE12	8Ch

Function

This register defines the final division value on frequency ramp-down for the progressive system clock switching of clk_src_12.

This registers is reset only on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved												DIVE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DIVE															
W																
Reset	0	0	0	0	0	0	1	1	1	1	1	0	0	1	1	1

Fields

Field	Function
31-20 —	This field is reserved and reads return zeros.
19-0 DIVE	Divider end value This field provides the end value of the clock divider for the clk_src_12 ramp-down phase.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462c54821-48a8-b251-3901-3ab624c1

24.5.5 PCFS Divider Start 12 Register (PCFS_DIVS12)

Offset

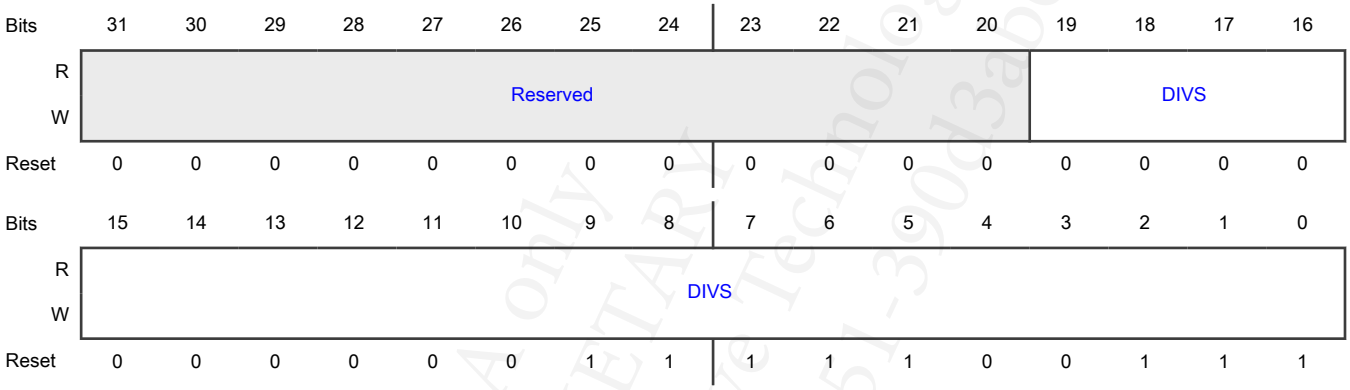
Register	Offset
PCFS_DIVS12	90h

Function

This register defines the initial division value on frequency ramp-up for the progressive system clock switching of clk_src_12.

This register is reset only on destructive reset.

Diagram



Fields

Field	Function
31-20 —	This field is reserved and reads return zeros.
19-0 DIVS	Divider start value This field provides the start value of the clock divider for the clk_src_12 ramp-up phase

24.5.6 Clock Mux 0 Select Control Register (MUX_0_CSC)

Offset

Register	Offset
MUX_0_CSC	300h

Function

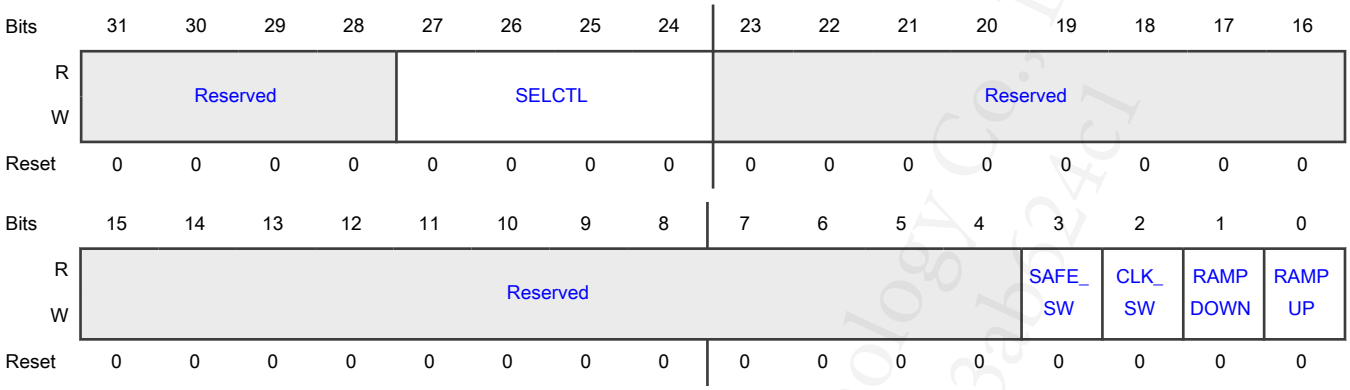
This register provides the clock source selection control for clock mux 0. Clock mux 0 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CTI Automotive Technology Co., Ltd.
8c462e5-4821-48a8-b251-39043a62401

This register is reset on destructive reset only.

An update to all the PCFS-related fields of this register must be an atomic write, which means a single write must update the CLK_SW, RAMPDOWN, and RAMPUP fields. It is necessary to set both RAMPUP and RAMPDOWN bits together even if you want to trigger either RAMPUP or RAMPDOWN process otherwise the desired PCFS sequence will not be executed.

Diagram



Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELCTL	Clock source selection control Selects the source clock for clock mux 0. The reserved values are not displayed. 0000b - clk_src_0 1100b - clk_src_12
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 0. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1 RAMPDOWN	PCFS ramp-down Writing 1 to this bit makes a PCFS ramp-down request to clock mux 0. After a PCFS ramp-down operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
0 RAMPUP	PCFS ramp-up Writing 1 to this bit makes a PCFS ramp-up request to clock mux 0. After a PCFS ramp-up operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.

24.5.7 Clock Mux 0 Select Status Register (MUX_0_CSS)

Offset

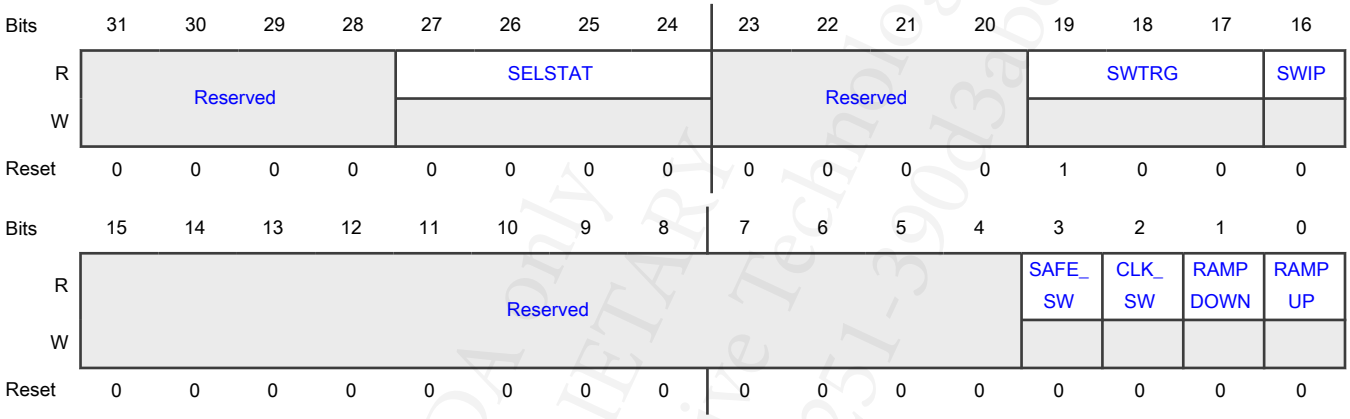
Register	Offset
MUX_0_CSS	304h

Function

This register provides the current clock source selection status for clock mux 0.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 0. The reserved values are not displayed. 0000b - clk_src_0 1100b - clk_src_12
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca162e5-4821-48a8-b251-390d3a624c1

Table continued from the previous page...

Field	Function
	NOTE If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 0. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 0. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1 RAMPDOWN	PCFS ramp-down

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field provides an indication of whether a PCFS ramp-down operation was requested during the previous/ongoing request on clock mux 0. **NOTE** In case of safe clock switching, the ramp-down operation runs internally, but the value of the corresponding status field is not set to 1. 0b - No ramp-down operation was requested. 1b - Ramp-down operation was requested.
0 RAMPUP	PCFS ramp-up This field provides an indication of whether a PCFS ramp-up operation was requested during the previous/ ongoing request on clock mux 0. 0b - No ramp-up operation was requested. 1b - Ramp-up operation was requested.

24.5.8 Clock Mux 0 Divider 0 Control Register (MUX_0_DC_0)

Offset

Register	Offset
MUX_0_DC_0	308h

Function

This register controls the clock divider 0 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DE	Reserved							DIV							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-8 —	This field is reserved and reads return zeros.
7-0 —	This field is reserved and reads return zeros.

24.5.9 Clock Mux 0 Divider 1 Control Register (MUX_0_DC_1)

Offset

Register	Offset
MUX_0_DC_1	30Ch

Function

This register controls the clock divider 1 for clock mux 0.

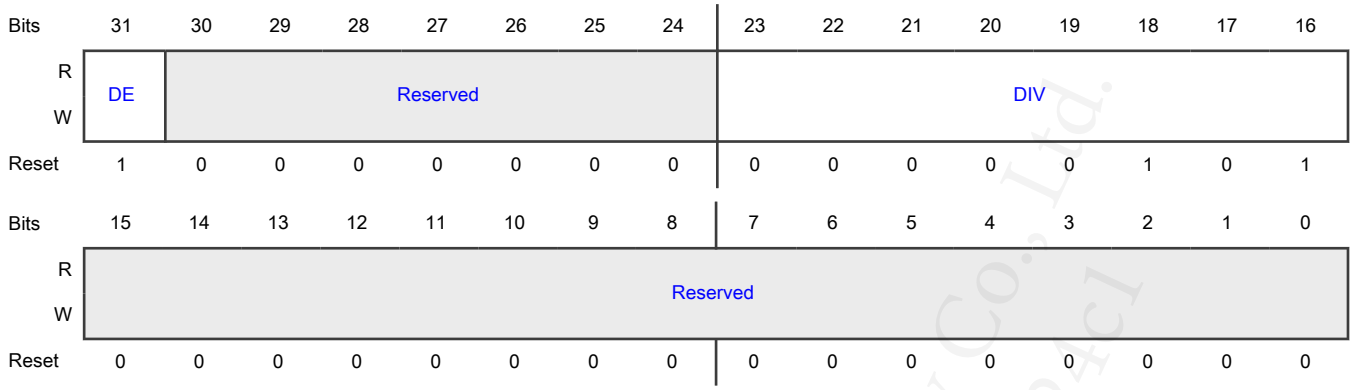
This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482f-48a8-b251-390d3ab604c1

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.10 Clock Mux 0 Divider Update Status Register (MUX_0_DIV_UPD_STAT)

Offset

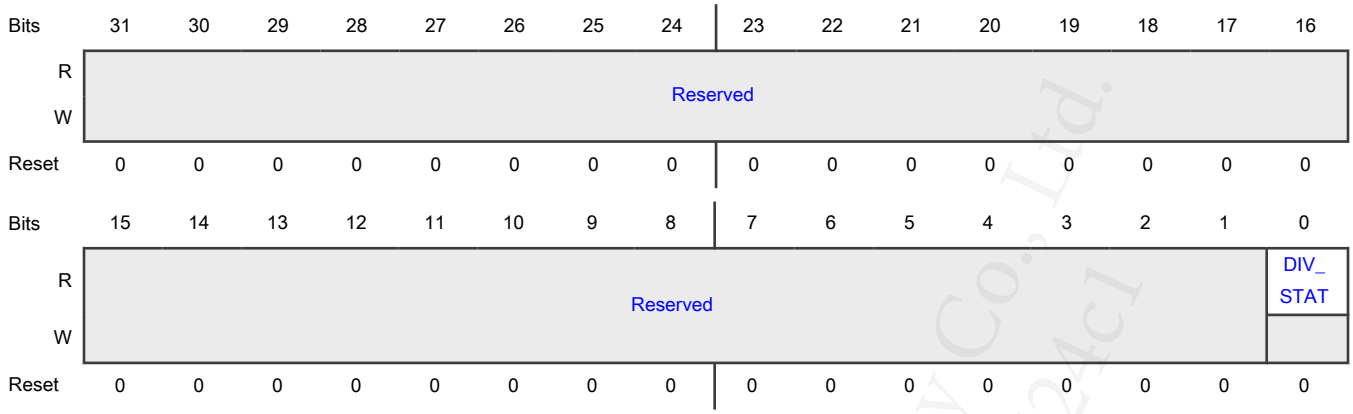
Register	Offset
MUX_0_DIV_UPD_STAT	33Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 0. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Provided under NDA only
BYD COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8a2e5-4821-48a8-9251-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 0 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.11 Clock Mux 1 Select Control Register (MUX_1_CSC)

Offset

Register	Offset
MUX_1_CSC	340h

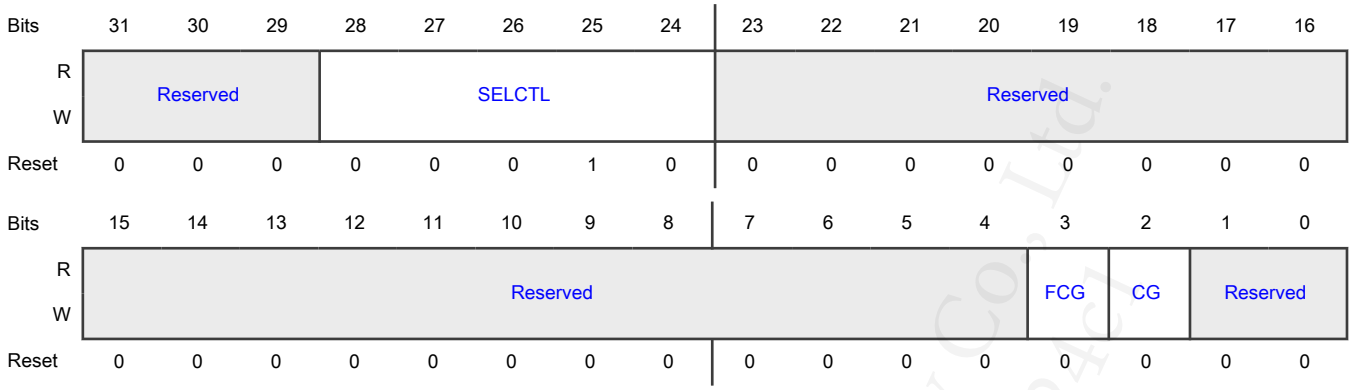
Function

This register provides the clock source selection control of clock mux 1. Clock mux 1 implements software control clock switching, and a graceful clock switch can be performed by executing a sequence of steps in software. See "Software-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e5-4821-48a8-b253-390d3ab624c1

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 1. The reserved values are not displayed. 00010b - clk_src_2 10010b - clk_src_18 11011b - clk_src_27 11110b - clk_src_30
23-4 —	This field is reserved and reads return zeros.
3 FCG	Force clock gate Writing 1 to this bit gates the clock at the output of clock mux 1 to logic-0 irrespective of the logic level of the currently selected clock. Clock gating using this bit should only be performed when it is insured that current clock source is inactive.
2 CG	Clock gate Writing 1 to this bit gates the clock at the output of clock mux 1 to logic-0. Using this bit it is insured that no glitches are resulted when gating the clock.
1-0 —	This field is reserved and reads return zeros.

Provided under NDA only
CONTAINS PROPRIETARY
Beijing HTC Automotive Technology Co., Ltd.
c8c46255-8214-48a8-b251-390d3ab624c1

24.5.12 Clock Mux 1 Select Status Register (MUX_1_CSS)

Offset

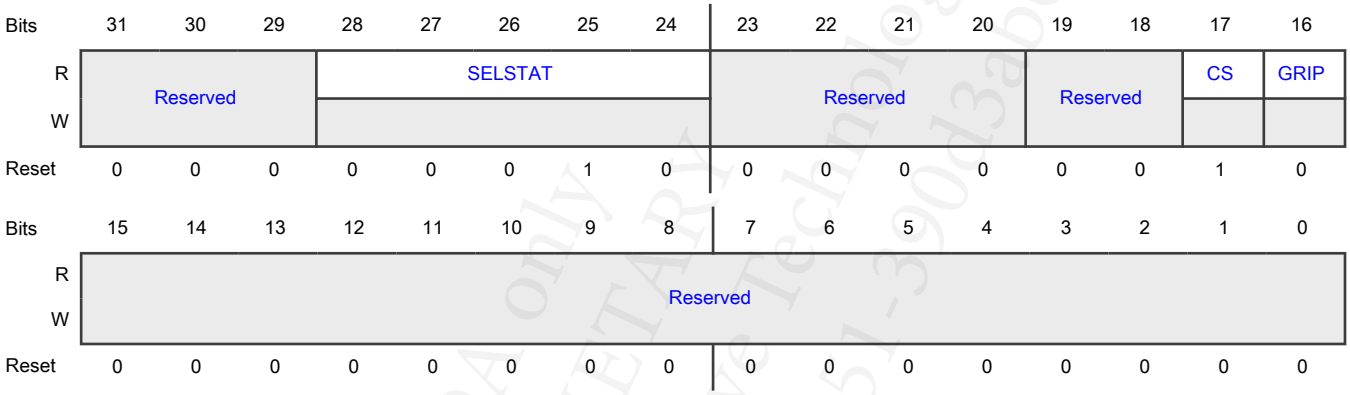
Register	Offset
MUX_1_CSS	344h

Function

This register provides the current clock source selection status for clock mux 1.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 1. The reserved values are not displayed. 00010b - clk_src_2 10010b - clk_src_18 11011b - clk_src_27 11110b - clk_src_30
23-20 —	This field is reserved and reads return zeros.
19-18 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 CS	Clock status This field indicates state of the clock at the output of the clock mux. 0b - Clock is gated to logic-0 at output of clock mux 1b - Clock mux is transparent. Active clock pulses at input of clock mux results in same number of pulses at its output
16 GRIP	Gating request is in progress. When a clock gate request is given this bit indicates if the clock gating at the output of mux has completed or not. 0b - Clock source gating or ungating has completed. 1b - Clock source gating or ungating is in progress.
15-0 —	This field is reserved and reads return zeros.

24.5.13 Clock Mux 1 Divider 0 Control Register (MUX_1_DC_0)

Offset

Register	Offset
MUX_1_DC_0	348h

Function

This register controls the clock divider 0 for clock mux 1.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DE	Reserved							DIV							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.14 Clock Mux 1 Divider Update Status Register (MUX_1_DIV_UPD_STAT)

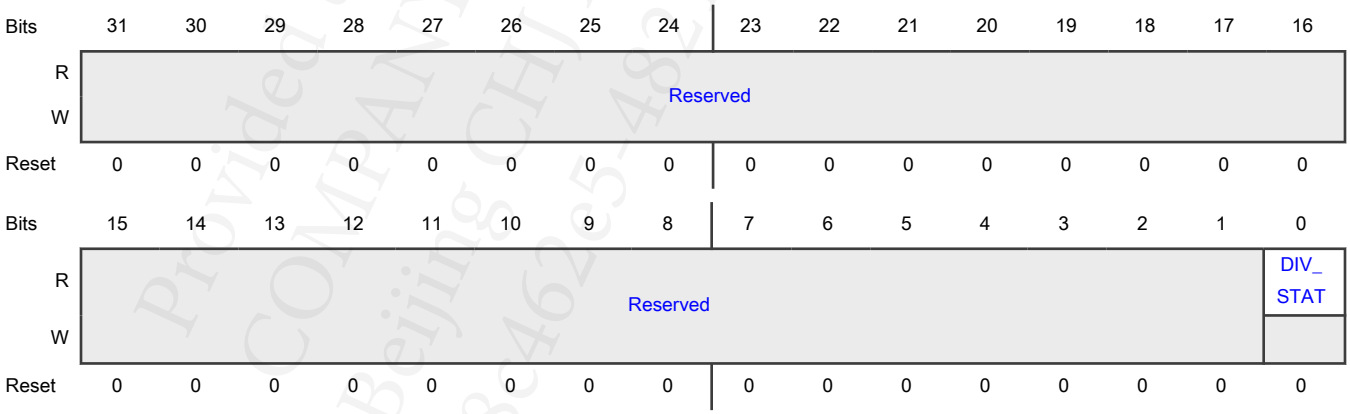
Offset

Register	Offset
MUX_1_DIV_UPD_STAT	37Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 1. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 1 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.15 Clock Mux 2 Select Control Register (MUX_2_CSC)

Offset

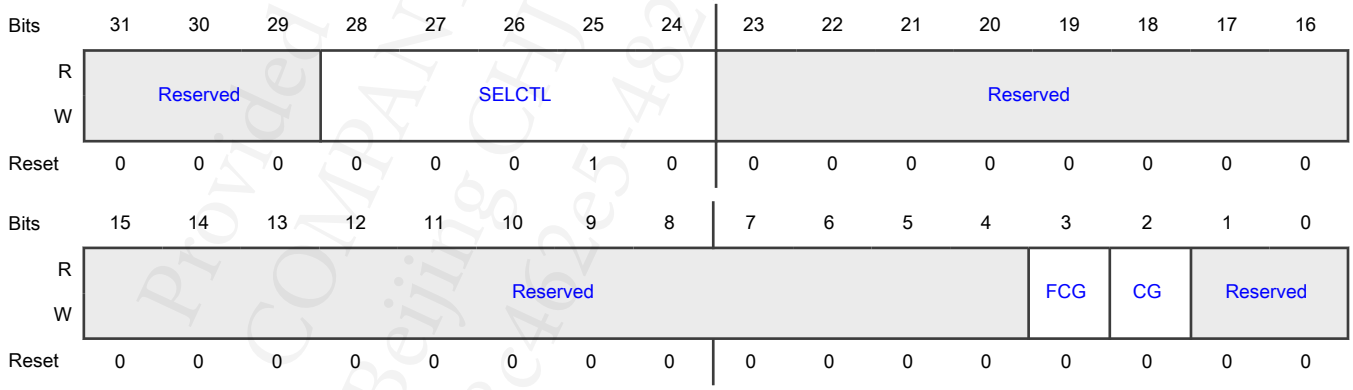
Register	Offset
MUX_2_CSC	380h

Function

This register provides the clock source selection control of clock mux 2. Clock mux 2 implements software control clock switching, and a graceful clock switch can be performed by executing a sequence of steps in software. See "Software-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c84e2e54-4821-48a1-b251-390a3ab624d1

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 2. The reserved values are not displayed. 00010b - clk_src_2 10010b - clk_src_18 11011b - clk_src_27 11110b - clk_src_30
23-4 —	This field is reserved and reads return zeros.
3 FCG	Force clock gate Writing 1 to this bit gates the clock at the output of clock mux 2 to logic-0 irrespective of the logic level of the currently selected clock. Clock gating using this bit should only be performed when it is insured that current clock source is inactive.
2 CG	Clock gate Writing 1 to this bit gates the clock at the output of clock mux 2 to logic-0. Using this bit it is insured that no glitches are resulted when gating the clock.
1-0 —	This field is reserved and reads return zeros.

24.5.16 Clock Mux 2 Select Status Register (MUX_2_CSS)

Offset

Register	Offset
MUX_2_CSS	384h

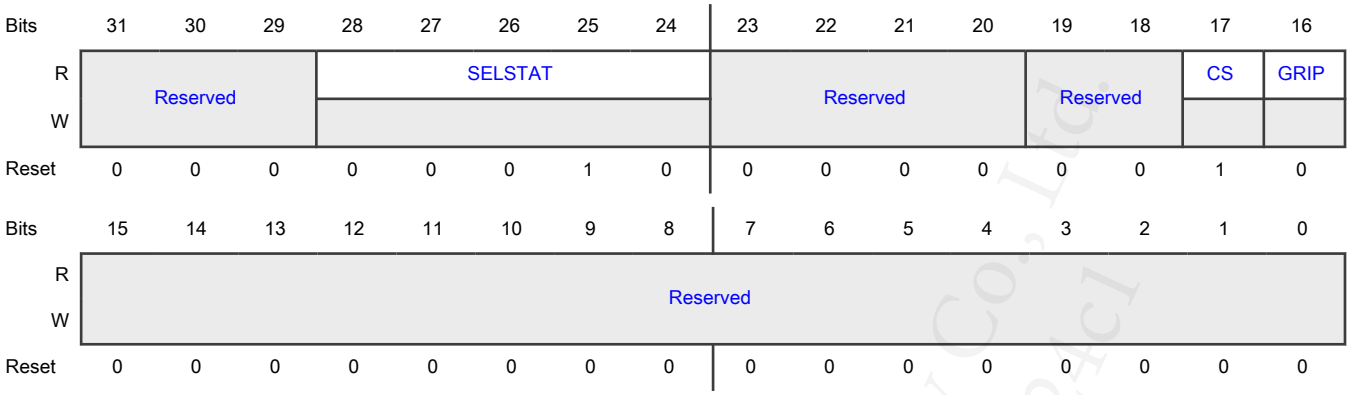
Function

This register provides the current clock source selection status for clock mux 2.

This register is reset on destructive reset only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-43a8-b251-30d3ab624c1

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 2. The reserved values are not displayed. 00010b - clk_src_2 10010b - clk_src_18 11011b - clk_src_27 11110b - clk_src_30
23-20 —	This field is reserved and reads return zeros.
19-18 —	This field is reserved and reads return zeros.
17 CS	Clock status This field indicates state of the clock at the output of the clock mux. 0b - Clock is gated to logic-0 at output of clock mux 1b - Clock mux is transparent. Active clock pulses at input of clock mux results in same number of pulses at its output
16 GRIP	Gating request is in progress. When a clock gate request is given this bit indicates if the clock gating at the output of mux has completed or not. 0b - Clock source gating or ungating has completed. 1b - Clock source gating or ungating is in progress.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 —	This field is reserved and reads return zeros.

24.5.17 Clock Mux 2 Divider 0 Control Register (MUX_2_DC_0)

Offset

Register	Offset
MUX_2_DC_0	388h

Function

This register controls the clock divider 0 for clock mux 2.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DE	Reserved							DIV							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.18 Clock Mux 2 Divider Update Status Register (MUX_2_DIV_UPD_STAT)

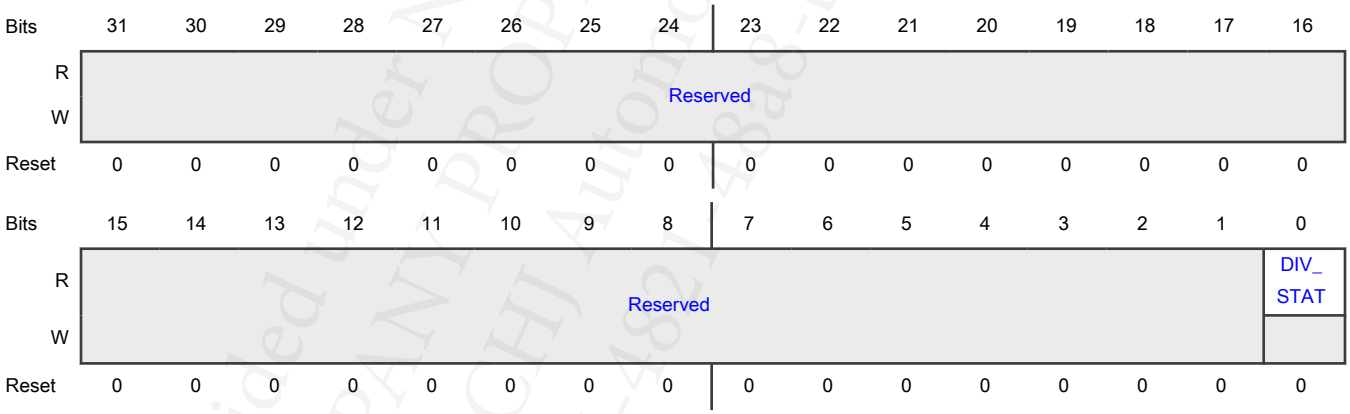
Offset

Register	Offset
MUX_2_DIV_UPD_STAT	3BCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 2. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0	Divider status for clock mux 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
DIV_STAT	On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.19 Clock Mux 3 Select Control Register (MUX_3_CSC)

Offset

Register	Offset
MUX_3_CSC	3C0h

Function

This register provides the clock source selection control for clock mux 3. Clock mux 3 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			SELCTL					Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												SAFE_SW	CLK_SW	Reserved	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 3. The reserved values are not displayed. 00000b - clk_src_0 10011b - clk_src_19
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 3. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.20 Clock Mux 3 Select Status Register (MUX_3_CSS)

Offset

Register	Offset
MUX_3_CSS	3C4h

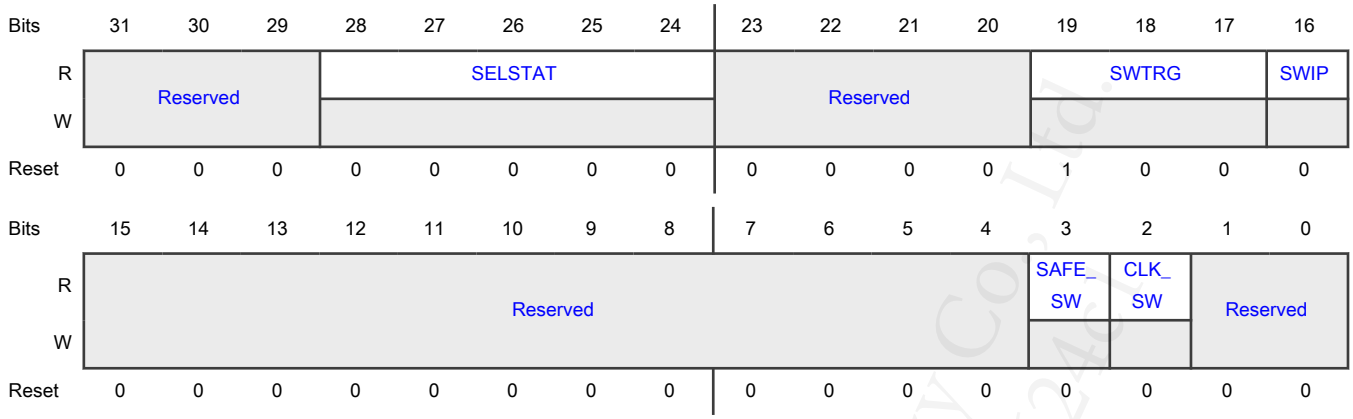
Function

This register provides the current clock source selection status for clock mux 3.

This register is reset on destructive reset only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-500deab624c1

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 3. The reserved values are not displayed. 00000b - clk_src_0 10011b - clk_src_19
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 3. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 3. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.21 Clock Mux 3 Divider 0 Control Register (MUX_3_DC_0)

Offset

Register	Offset
MUX_3_DC_0	3C8h

Function

This register controls the clock divider 0 for clock mux 3.

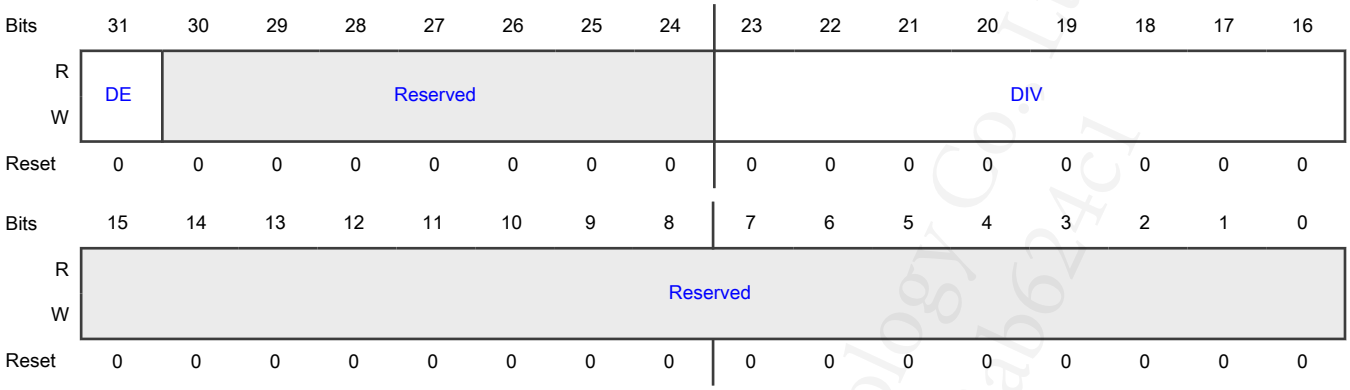
This divider is a 50% duty cycle divider.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-481-48a8-b251-390d3ab6240d

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.22 Clock Mux 3 Divider Update Status Register (MUX_3_DIV_UPD_STAT)

Offset

Register	Offset
MUX_3_DIV_UPD_STAT	3FCh

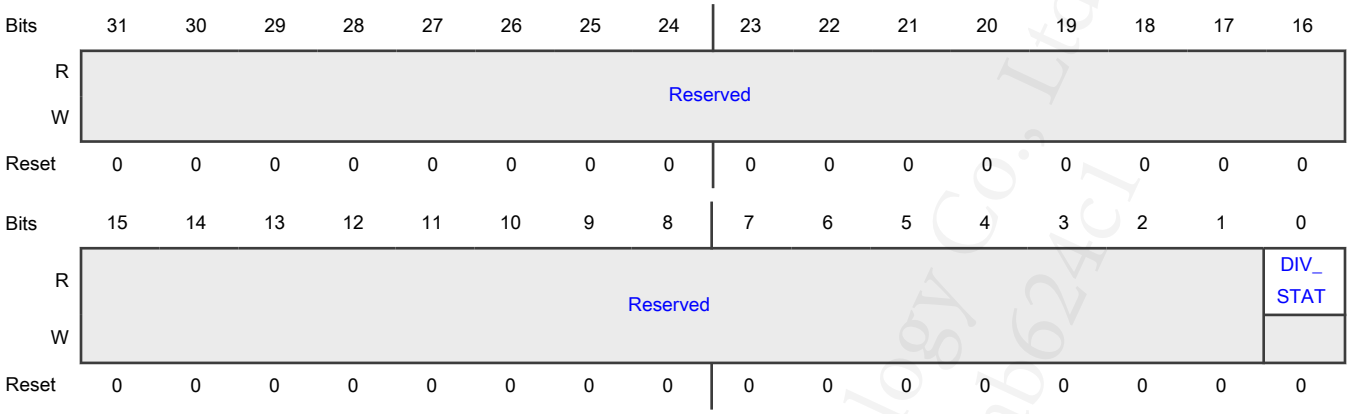
Function

This register provides the update status of the clock dividers corresponding to clock mux 3. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
c54625e5-4d21-4e8-b251-390d3ab624c1

clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 3 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.23 Clock Mux 4 Select Control Register (MUX_4_CSC)

Offset

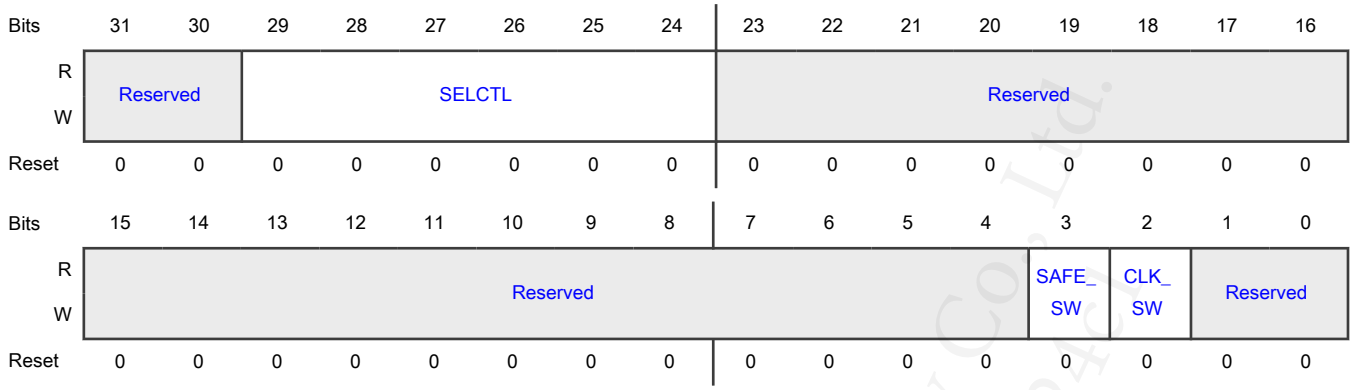
Register	Offset
MUX_4_CSC	400h

Function

This register provides the clock source selection control for clock mux 4. Clock mux 4 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 4. The reserved values are not displayed. 000000b - clk_src_0 010011b - clk_src_19 100010b - clk_src_34
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 4. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.24 Clock Mux 4 Select Status Register (MUX_4_CSS)

Offset

Register	Offset
MUX_4_CSS	404h

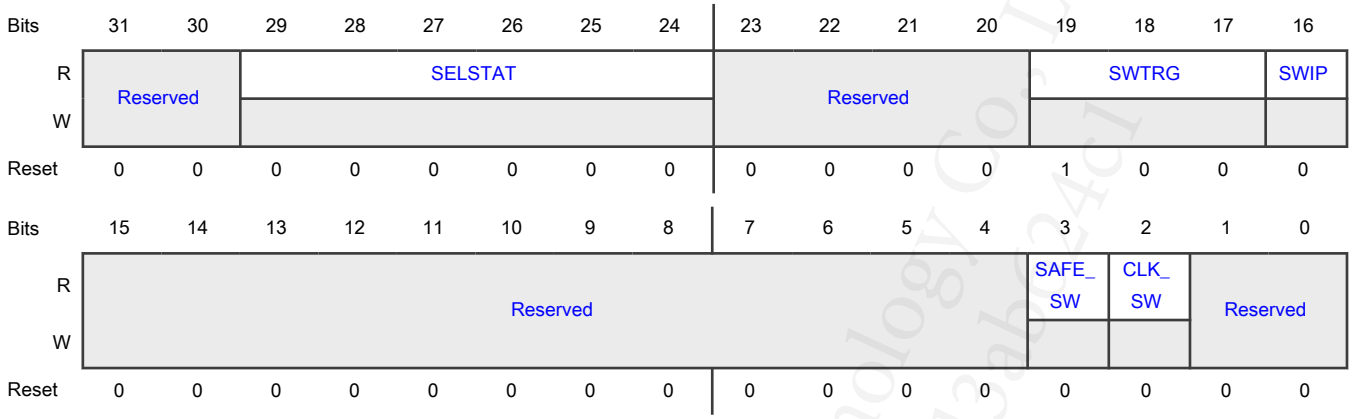
Provided for NDA only
COMPANY PROPRIETARY
Beijing CHN Automotive Technology Co., Ltd.
c8c4625-4a91-4a8-b251-390d3ab624

Function

This register provides the current clock source selection status for clock mux 4.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 4. The reserved values are not displayed. 000000b - clk_src_0 010011b - clk_src_19 100010b - clk_src_34
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 4. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 4. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.25 Clock Mux 4 Divider 0 Control Register (MUX_4_DC_0)

Offset

Register	Offset
MUX_4_DC_0	408h

Function

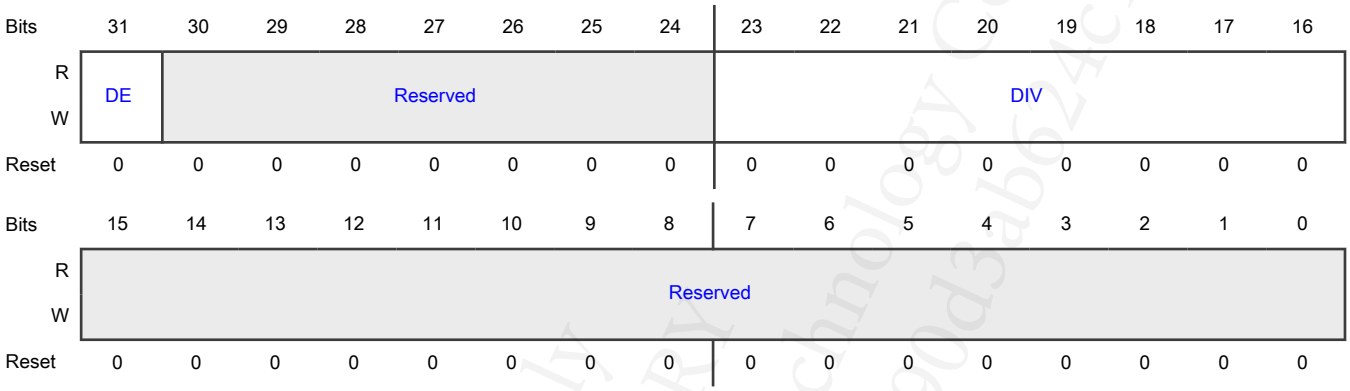
This register controls the clock divider 0 for clock mux 4.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.26 Clock Mux 4 Divider Update Status Register (MUX_4_DIV_UPD_STAT)

Offset

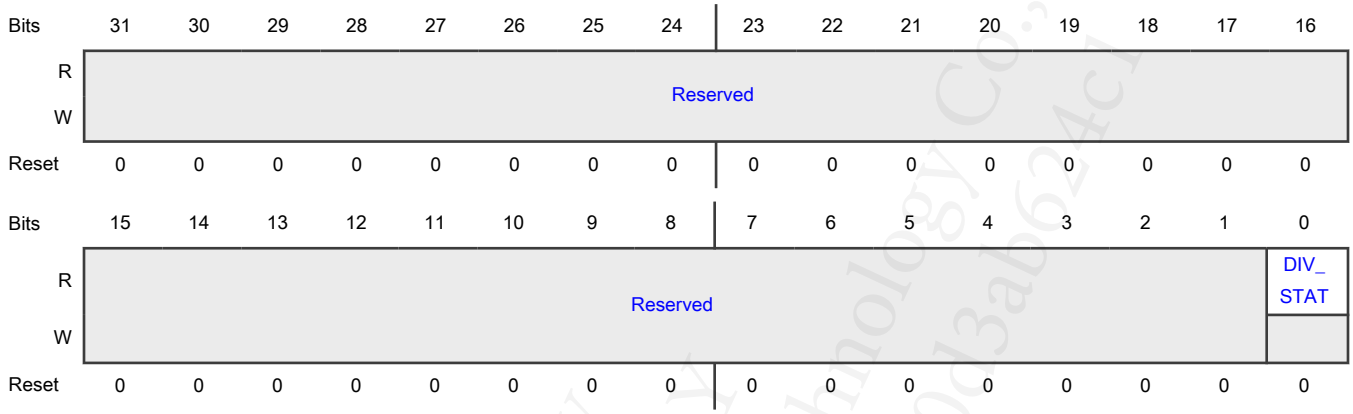
Register	Offset
MUX_4_DIV_UPD_STAT	43Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c46255-4a2-48a8-b251-390d3ab624c1

Function

This register provides the update status of the clock dividers corresponding to clock mux 4. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 4 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.27 Clock Mux 5 Select Control Register (MUX_5_CSC)

Offset

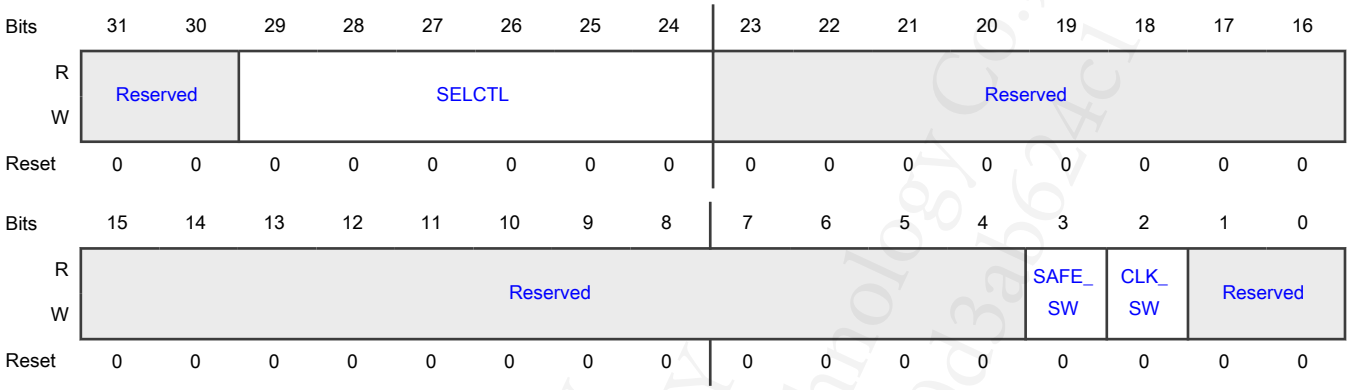
Register	Offset
MUX_5_CSC	440h

Function

This register provides the clock source selection control for clock mux 5. Clock mux 5 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 5. The reserved values are not displayed. 000000b - clk_src_0 010011b - clk_src_19 100011b - clk_src_35
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 5. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8402e48-21-48ac-b251-390a3ab624c1

24.5.28 Clock Mux 5 Select Status Register (MUX_5_CSS)

Offset

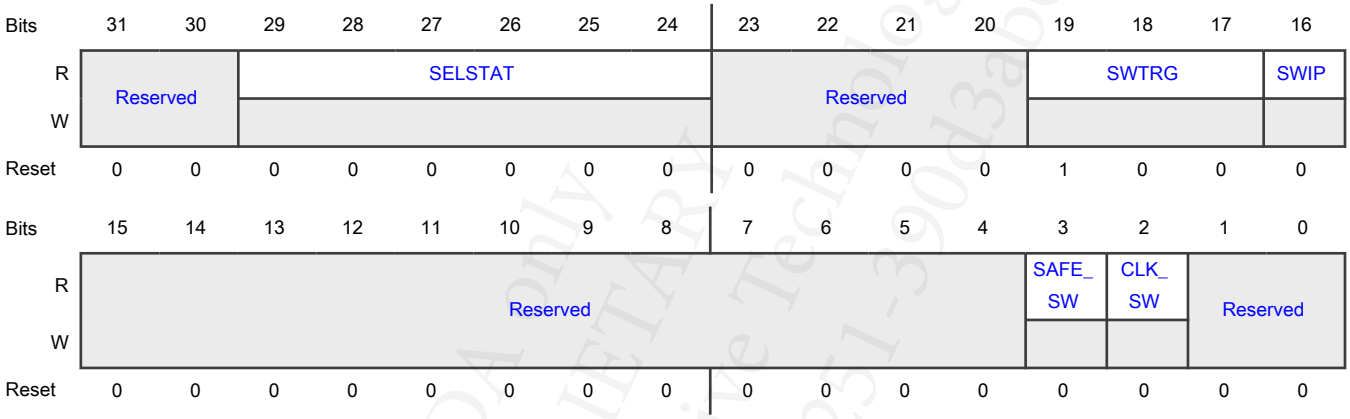
Register	Offset
MUX_5_CSS	444h

Function

This register provides the current clock source selection status for clock mux 5.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 5. The reserved values are not displayed. 000000b - clk_src_0 010011b - clk_src_19 100011b - clk_src_35
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 5. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 5. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

24.5.29 Clock Mux 5 Divider 0 Control Register (MUX_5_DC_0)

Offset

Register	Offset
MUX_5_DC_0	448h

Function

This register controls the clock divider 0 for clock mux 5.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DE	Reserved							DIV							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.30 Clock Mux 5 Divider Update Status Register (MUX_5_DIV_UPD_STAT)

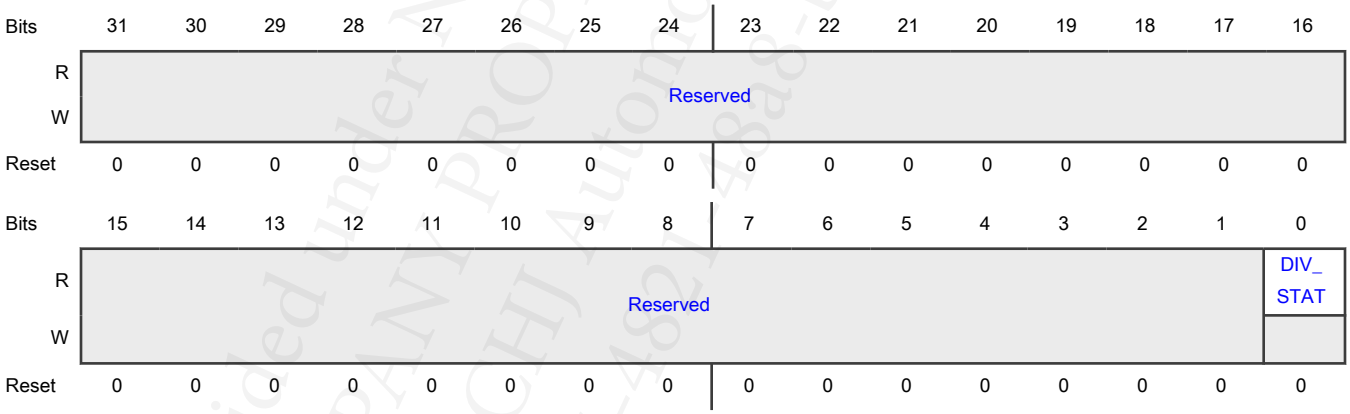
Offset

Register	Offset
MUX_5_DIV_UPD_STAT	47Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 5. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0	Divider status for clock mux 5

Table continues on the next page...

Table continued from the previous page...

Field	Function
DIV_STAT	On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.31 Clock Mux 6 Select Control Register (MUX_6_CSC)

Offset

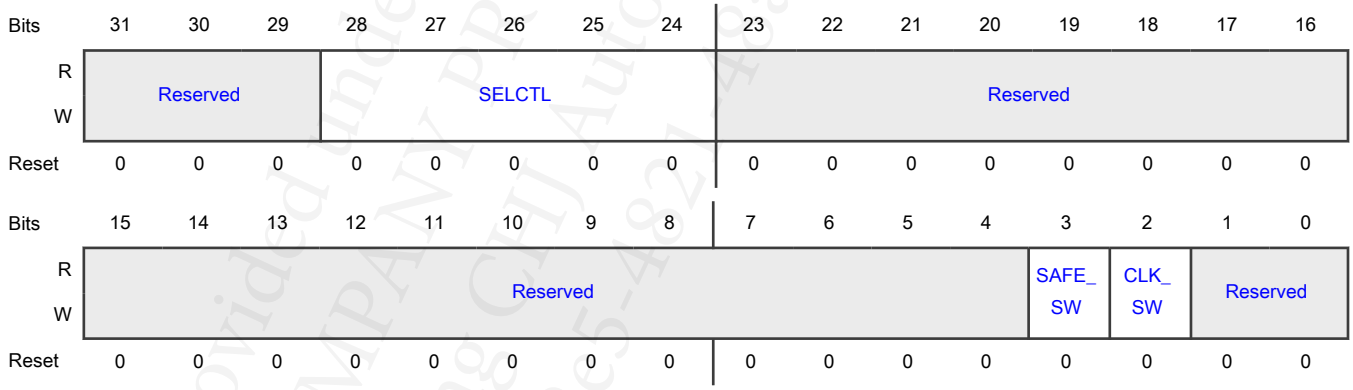
Register	Offset
MUX_6_CSC	480h

Function

This register provides the clock source selection control for clock mux 6. Clock mux 6 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 6. The reserved values are not displayed. 00000b - clk_src_0 00010b - clk_src_2 10011b - clk_src_19
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 6. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.32 Clock Mux 6 Select Status Register (MUX_6_CSS)

Offset

Register	Offset
MUX_6_CSS	484h

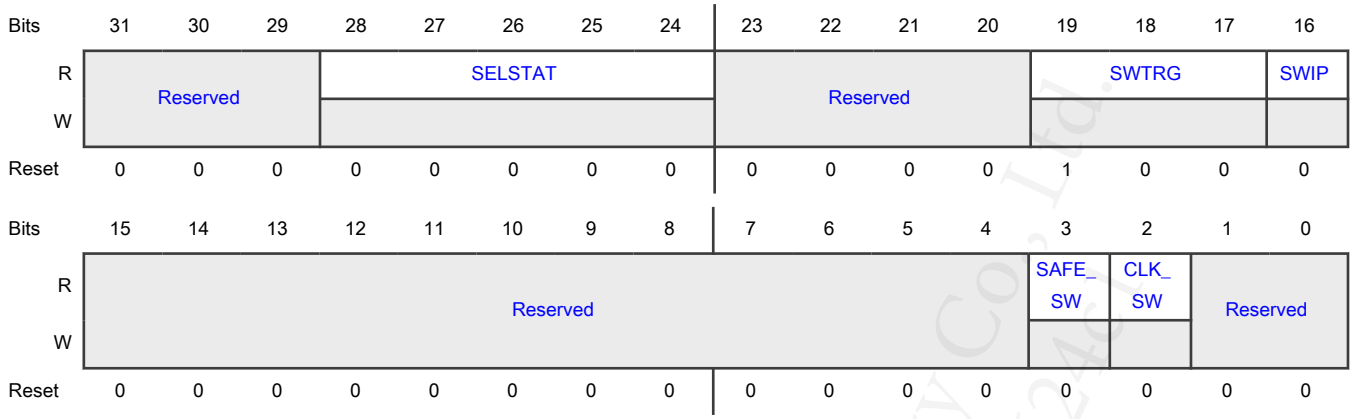
Function

This register provides the current clock source selection status for clock mux 6.

This register is reset on destructive reset only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-903ab624c1

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 6. The reserved values are not displayed. 00000b - clk_src_0 00010b - clk_src_2 10011b - clk_src_19
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 6. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 6. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.33 Clock Mux 6 Divider 0 Control Register (MUX_6_DC_0)

Offset

Register	Offset
MUX_6_DC_0	488h

Function

This register controls the clock divider 0 for clock mux 6.

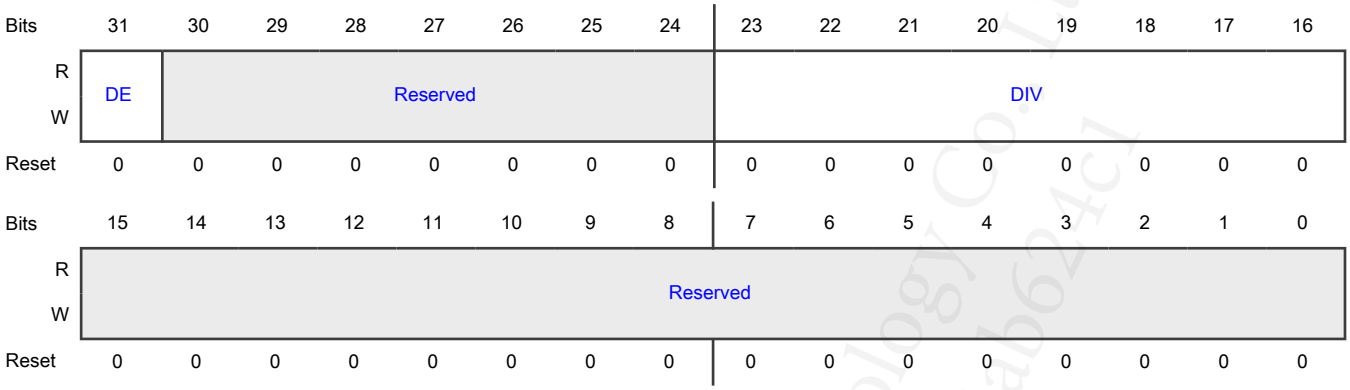
This divider is a 50% duty cycle divider.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-481-48a8-b251-390d3ab6240

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.34 Clock Mux 6 Divider Update Status Register (MUX_6_DIV_UPD_STAT)

Offset

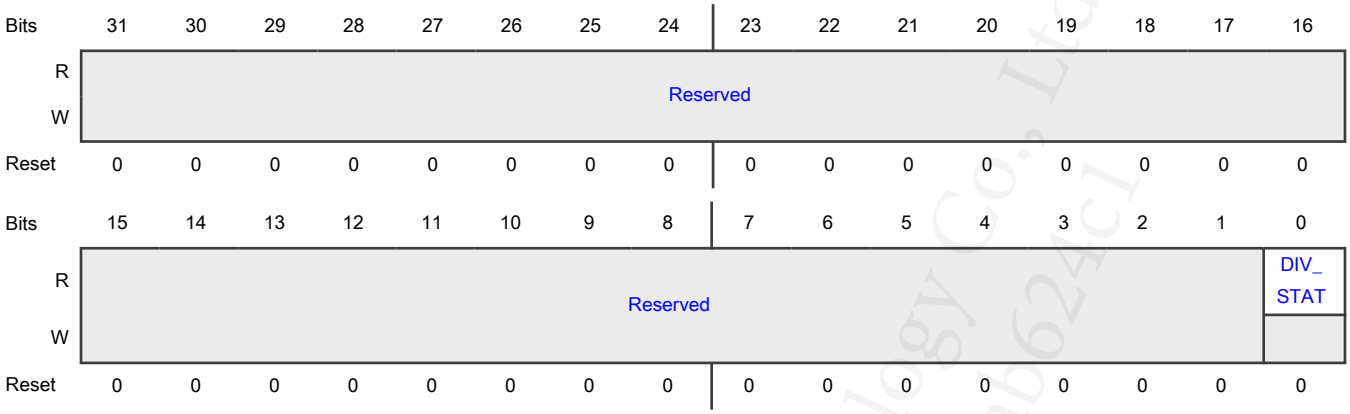
Register	Offset
MUX_6_DIV_UPD_STAT	4BCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 6. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other

clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 6 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.35 Clock Mux 7 Select Control Register (MUX_7_CSC)

Offset

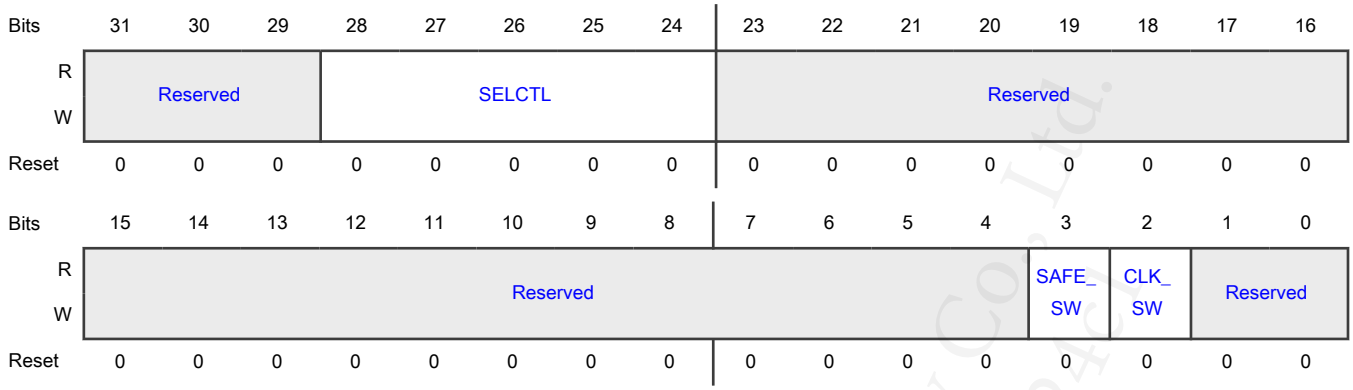
Register	Offset
MUX_7_CSC	4C0h

Function

This register provides the clock source selection control for clock mux 7. Clock mux 7 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 7. The reserved values are not displayed. 00000b - clk_src_0 00010b - clk_src_2 10100b - clk_src_20
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 7. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.36 Clock Mux 7 Select Status Register (MUX_7_CSS)

Offset

Register	Offset
MUX_7_CSS	4C4h

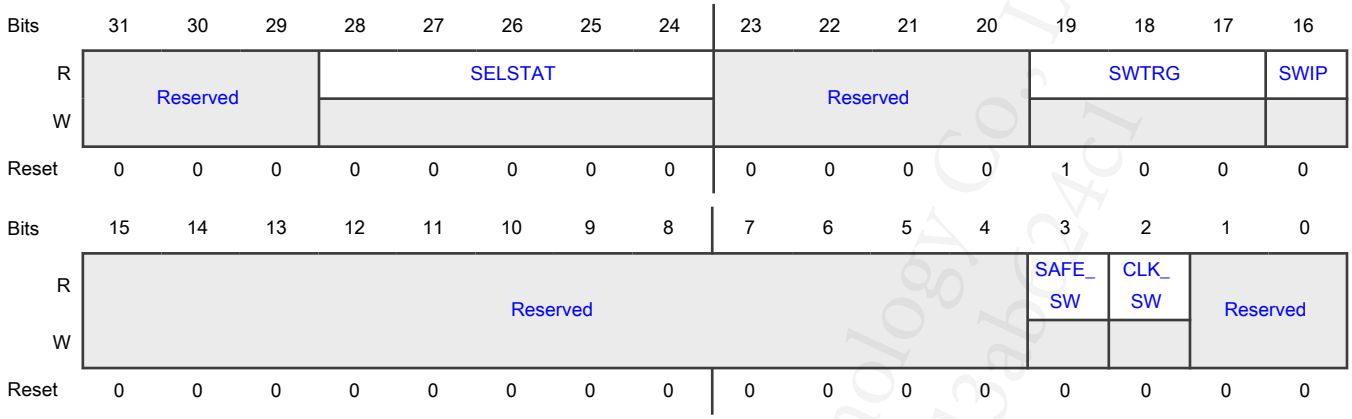
Provided for NXP USA only
COMPANY PROPRIETARY
Beijing CEC Automotive Technology Co., Ltd.
c8c4625-4a91-48a8-b251-390d3ab624c

Function

This register provides the current clock source selection status for clock mux 7.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 7. The reserved values are not displayed. 00000b - clk_src_0 00010b - clk_src_2 10100b - clk_src_20
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 7. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 7. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.37 Clock Mux 8 Select Control Register (MUX_8_CSC)

Offset

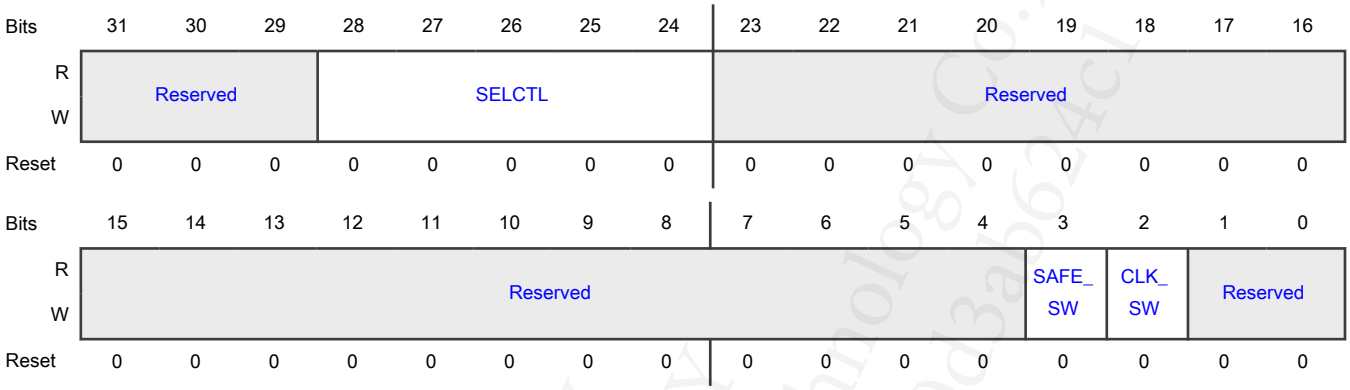
Register	Offset
MUX_8_CSC	500h

Function

This register provides the clock source selection control for clock mux 8. Clock mux 8 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 8. The reserved values are not displayed. 00000b - clk_src_0 00010b - clk_src_2 10101b - clk_src_21
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 8. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c84a2e3-4821-48ac-b251-3903ab624c1

24.5.38 Clock Mux 8 Select Status Register (MUX_8_CSS)

Offset

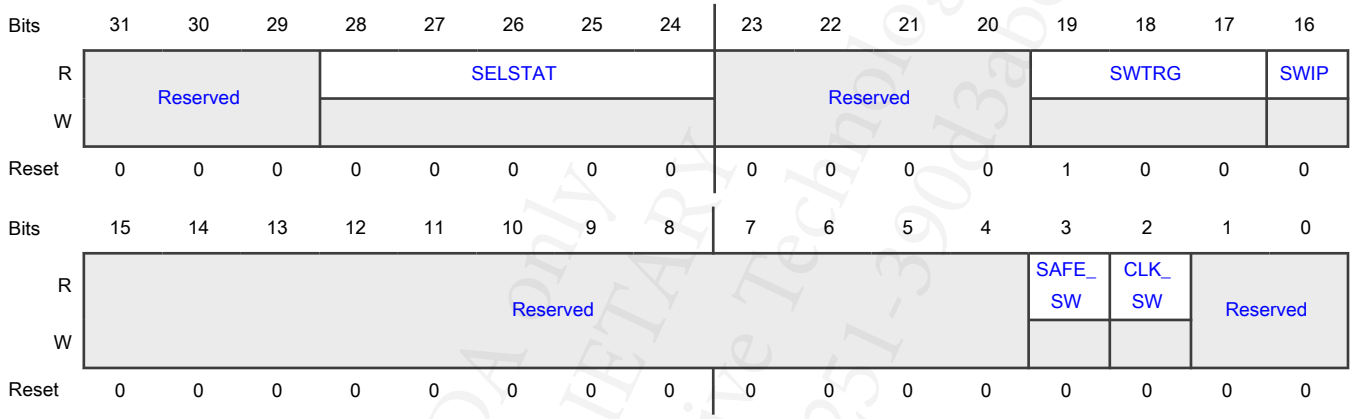
Register	Offset
MUX_8_CSS	504h

Function

This register provides the current clock source selection status for clock mux 8.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 8. The reserved values are not displayed. 00000b - clk_src_0 00010b - clk_src_2 10101b - clk_src_21
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 8. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 8. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

24.5.39 Clock Mux 9 Select Control Register (MUX_9_CSC)

Offset

Register	Offset
MUX_9_CSC	540h

Function

This register provides the clock source selection control for clock mux 9. Clock mux 9 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		SELCTL						Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												SAFE_SW	CLK_SW	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 9. The reserved values are not displayed. 000000b - clk_src_0 010110b - clk_src_22 101100b - clk_src_44

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 9. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.40 Clock Mux 9 Select Status Register (MUX_9_CSS)

Offset

Register	Offset
MUX_9_CSS	544h

Function

This register provides the current clock source selection status for clock mux 9.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		SELSTAT						Reserved				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												SAFE_SW	CLK_SW	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48ad-b251-390d3ab6240f

Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 9. The reserved values are not displayed. 000000b - clk_src_0 010110b - clk_src_22 101100b - clk_src_44
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 9. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 9. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.41 Clock Mux 9 Divider 0 Control Register (MUX_9_DC_0)

Offset

Register	Offset
MUX_9_DC_0	548h

Function

This register controls the clock divider 0 for clock mux 9.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DE	Reserved							DIV							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.42 Clock Mux 9 Divider Update Status Register (MUX_9_DIV_UPD_STAT)

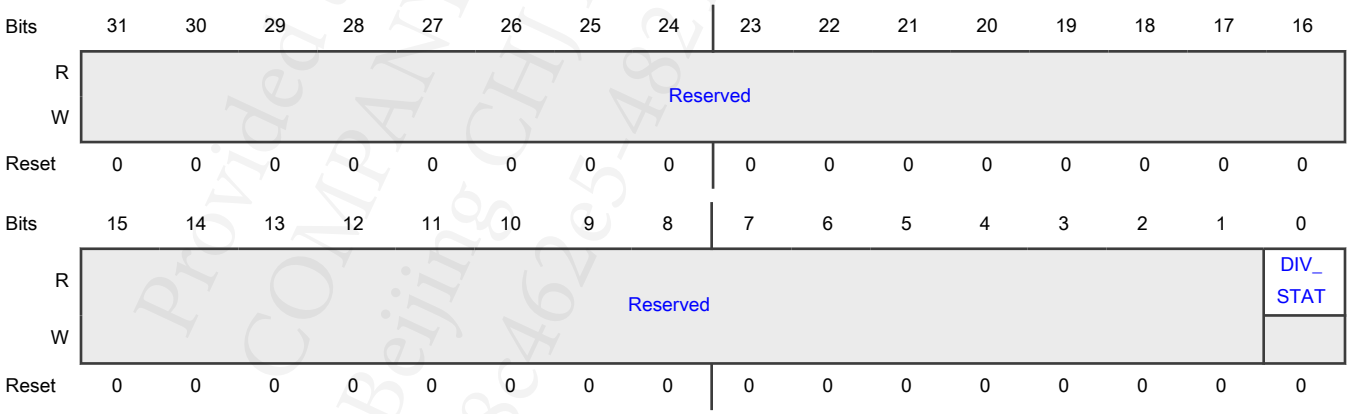
Offset

Register	Offset
MUX_9_DIV_UPD_STAT	57Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 9. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 9 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.43 Clock Mux 10 Select Control Register (MUX_10_CSC)

Offset

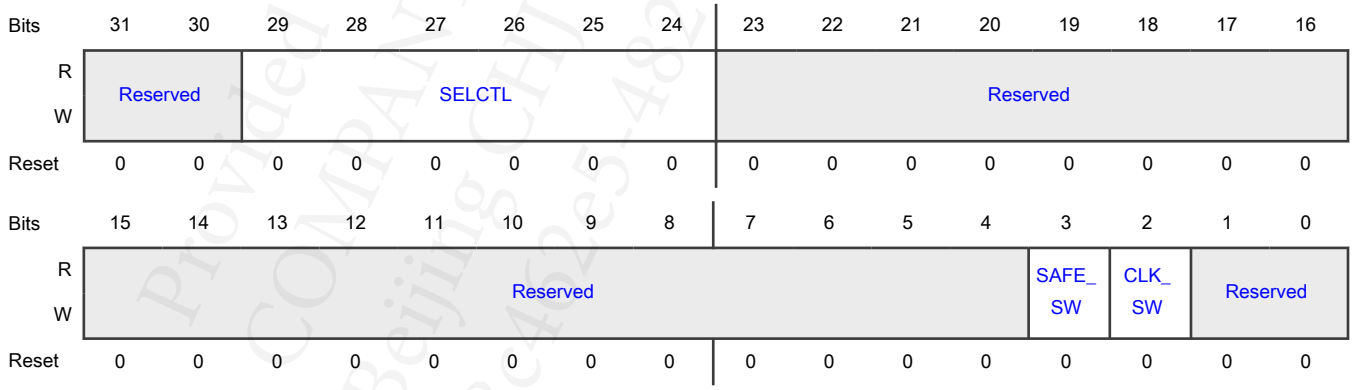
Register	Offset
MUX_10_CSC	580h

Function

This register provides the clock source selection control for clock mux 10. Clock mux 10 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8462e54-821-48a-b251-390a3ab624d

Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 10. The reserved values are not displayed. 000000b - clk_src_0 010111b - clk_src_23 100101b - clk_src_37 100111b - clk_src_39 101000b - clk_src_40
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 10. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.44 Clock Mux 10 Select Status Register (MUX_10_CSS)

Offset

Register	Offset
MUX_10_CSS	584h

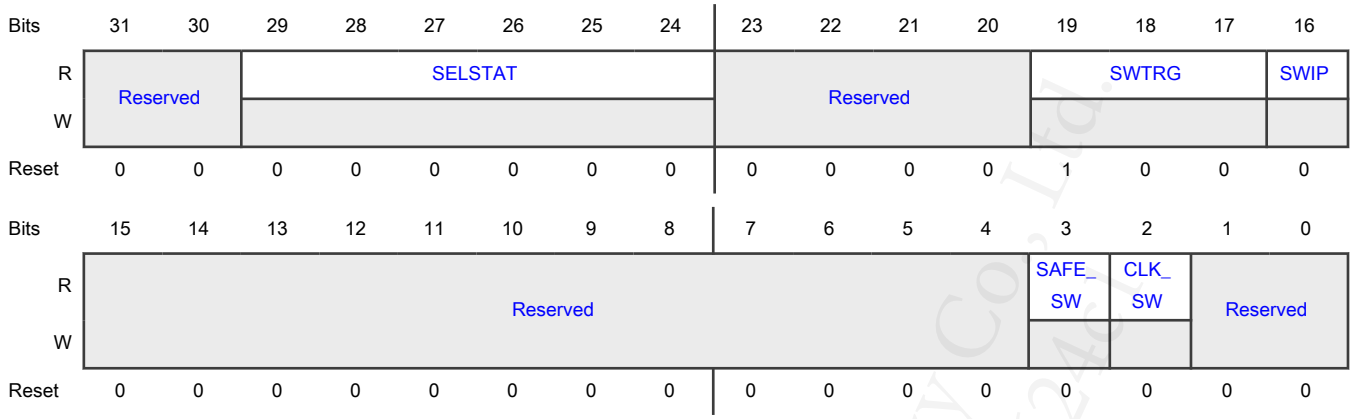
Function

This register provides the current clock source selection status for clock mux 10.

This register is reset on destructive reset only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b051-90d3ab624c1

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 10. The reserved values are not displayed. 000000b - clk_src_0 010111b - clk_src_23 100101b - clk_src_37 100111b - clk_src_39 101000b - clk_src_40
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 10. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 10. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.45 Clock Mux 10 Divider 0 Control Register (MUX_10_DC_0)

Offset

Register	Offset
MUX_10_DC_0	588h

Function

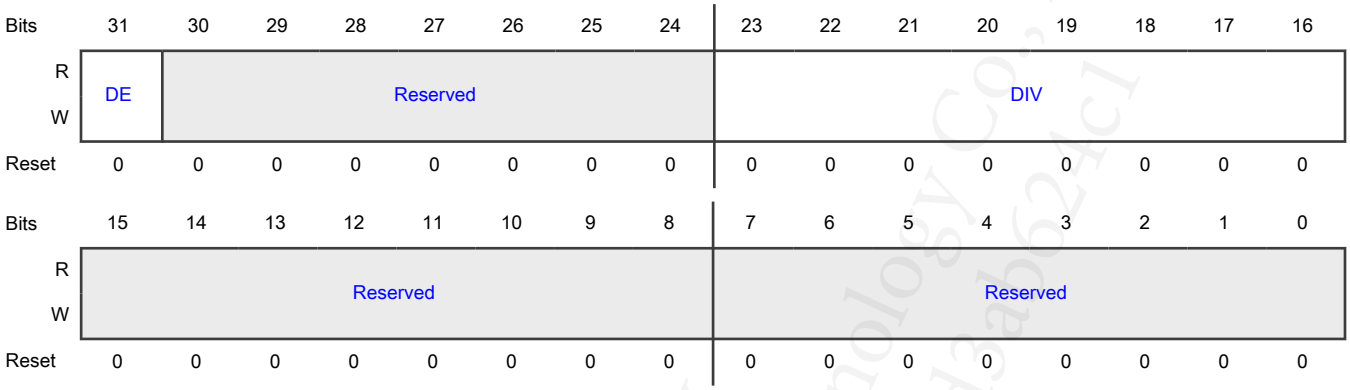
This register controls the clock divider 0 for clock mux 10.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-8 —	This field is reserved and reads return zeros.
7-0 —	This field is reserved and reads return zeros.

24.5.46 Clock Mux 10 Divider Update Status Register (MUX_10_DIV_UPD_STAT)

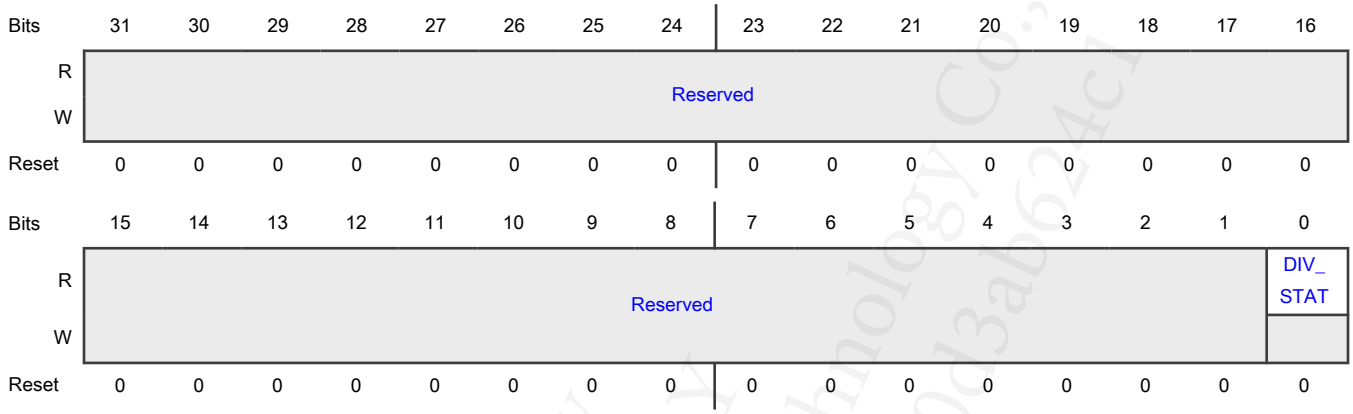
Offset

Register	Offset
MUX_10_DIV_UPD_STAT	5BCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 10. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 10 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.47 Clock Mux 11 Select Control Register (MUX_11_CSC)

Offset

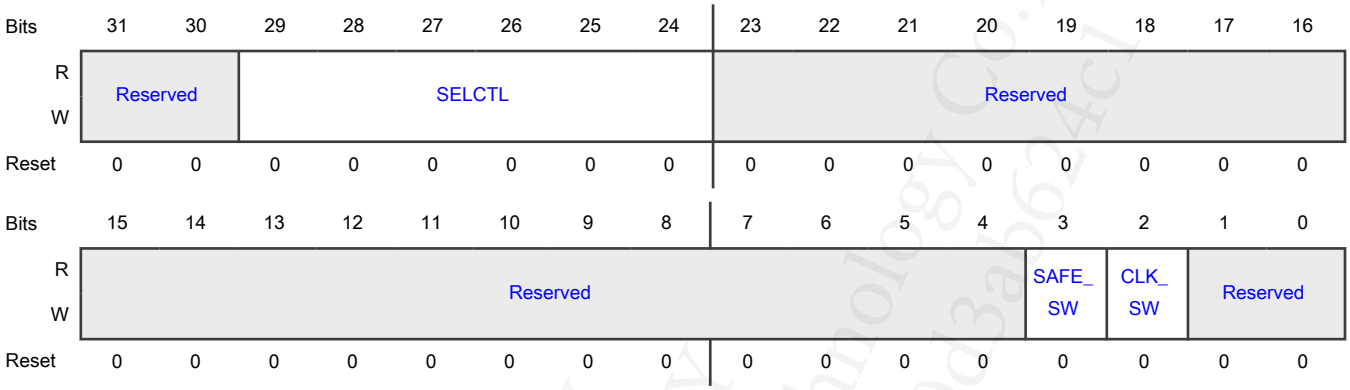
Register	Offset
MUX_11_CSC	5C0h

Function

This register provides the clock source selection control for clock mux 11. Clock mux 11 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 11. The reserved values are not displayed. 000000b - clk_src_0 100110b - clk_src_38 101001b - clk_src_41 101101b - clk_src_45
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 11. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.48 Clock Mux 11 Select Status Register (MUX_11_CSS)

Offset

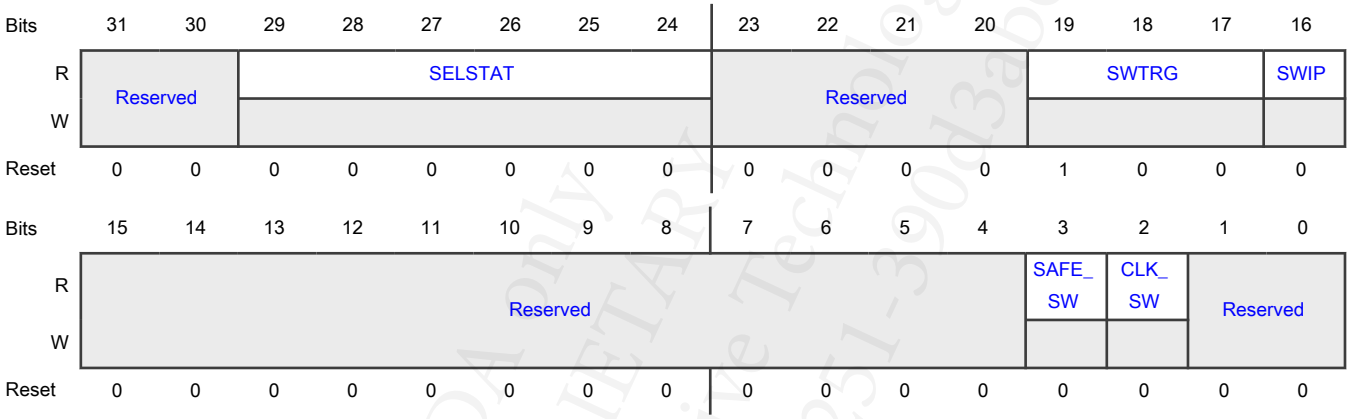
Register	Offset
MUX_11_CSS	5C4h

Function

This register provides the current clock source selection status for clock mux 11.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 11. The reserved values are not displayed. 000000b - clk_src_0 100110b - clk_src_38 101001b - clk_src_41 101101b - clk_src_45
23-20 —	This field is reserved and reads return zeros.
19-17	Switch trigger cause

Table continues on the next page...

Table continued from the previous page...

Field	Function
SWTRG	This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 11. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 11. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

24.5.49 Clock Mux 12 Select Control Register (MUX_12_CSC)

Offset

Register	Offset
MUX_12_CSC	600h

Function

This register provides the clock source selection control for clock mux 12. Clock mux 12 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			SELCTL					Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												SAFE_SW	CLK_SW	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 12. The reserved values are not displayed. 00000b - clk_src_0 11010b - clk_src_26

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 12. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.50 Clock Mux 12 Select Status Register (MUX_12_CSS)

Offset

Register	Offset
MUX_12_CSS	604h

Function

This register provides the current clock source selection status for clock mux 12.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			SELSTAT					Reserved				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												SAFE_SW	CLK_SW	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a6-b251-390a3ab6240f

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 12. The reserved values are not displayed. 00000b - clk_src_0 11010b - clk_src_26
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 12. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 12. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.51 Clock Mux 12 Divider 0 Control Register (MUX_12_DC_0)

Offset

Register	Offset
MUX_12_DC_0	608h

Function

This register controls the clock divider 0 for clock mux 12.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DE	Reserved							DIV							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-482-48a8-b251-390d3ab624c1

Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.52 Clock Mux 12 Divider Update Status Register (MUX_12_DIV_UPD_STAT)

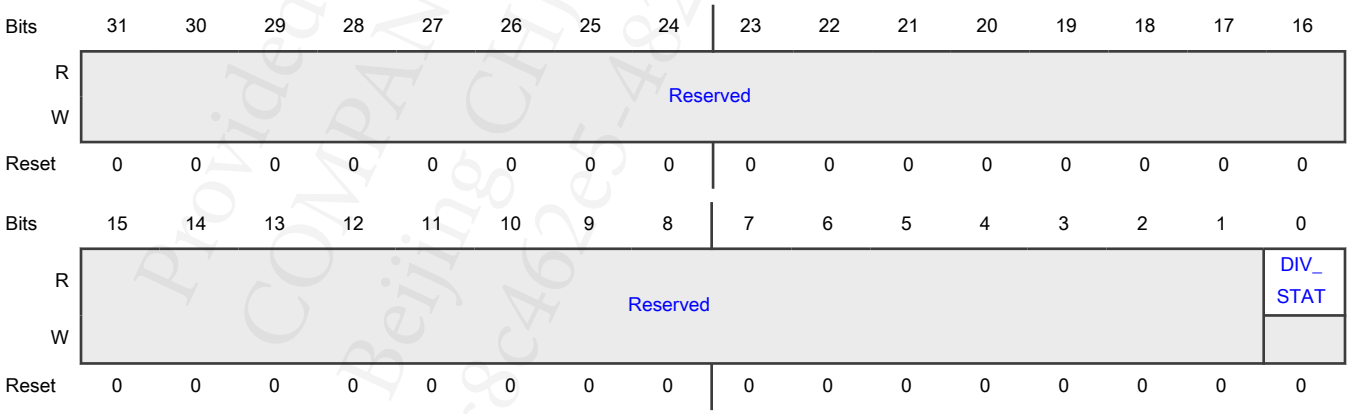
Offset

Register	Offset
MUX_12_DIV_UPD_STAT	63Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 12. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 12 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.53 Clock Mux 14 Select Control Register (MUX_14_CSC)

Offset

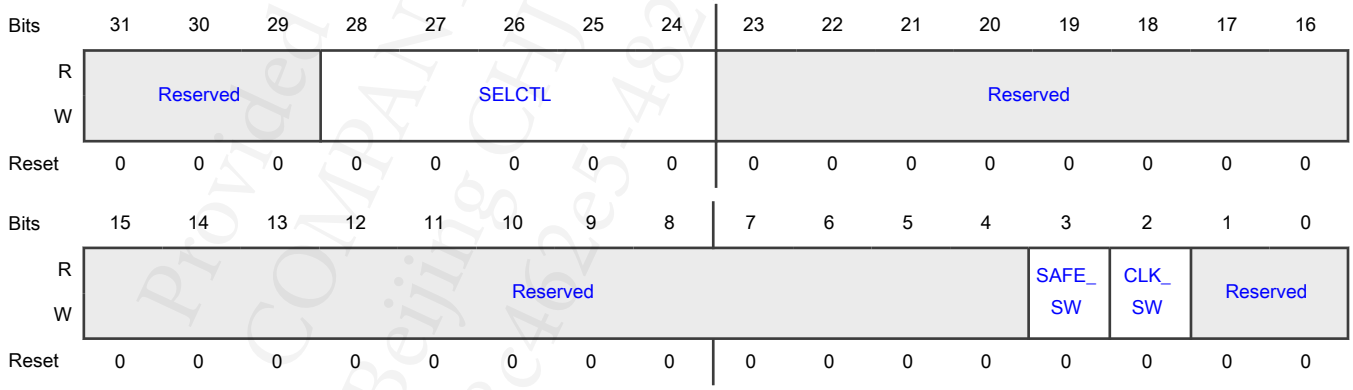
Register	Offset
MUX_14_CSC	680h

Function

This register provides the clock source selection control for clock mux 14. Clock mux 14 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e54-821-48a-b251-390a3ab624d1

Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 14. The reserved values are not displayed. 00000b - clk_src_0 11100b - clk_src_28
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 14. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.54 Clock Mux 14 Select Status Register (MUX_14_CSS)

Offset

Register	Offset
MUX_14_CSS	684h

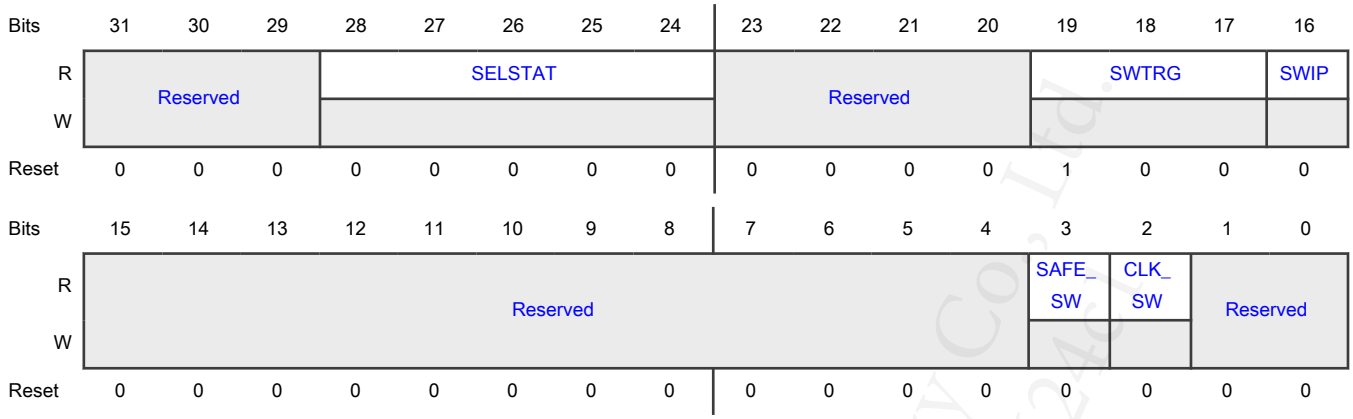
Function

This register provides the current clock source selection status for clock mux 14.

This register is reset on destructive reset only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-821-48a8-b251-903ab624c1

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 14. The reserved values are not displayed. 00000b - clk_src_0 11100b - clk_src_28
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 14. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 14. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.55 Clock Mux 14 Divider 0 Control Register (MUX_14_DC_0)

Offset

Register	Offset
MUX_14_DC_0	688h

Function

This register controls the clock divider 0 for clock mux 14.

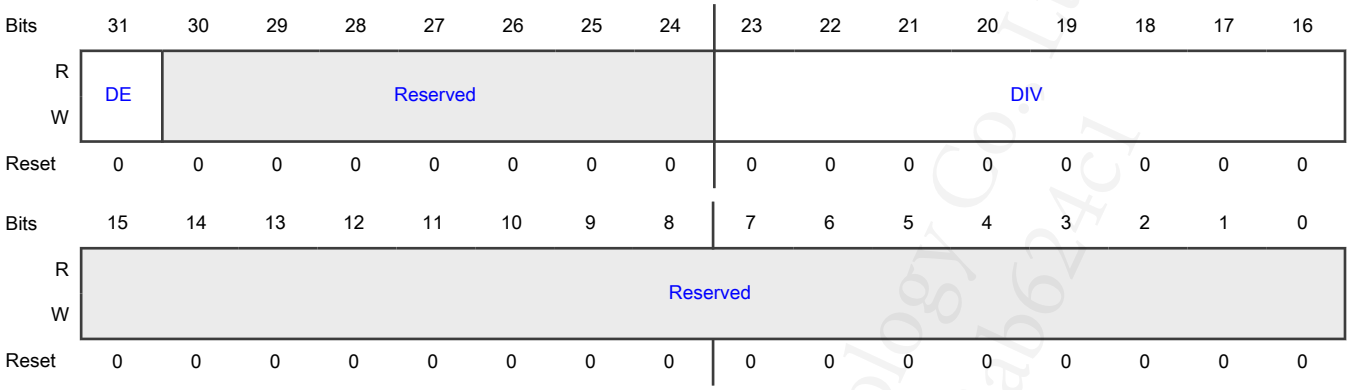
This divider is a 50% duty cycle divider.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8462e5-481-48a8-b251-390d3ab6240

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.56 Clock Mux 14 Divider Update Status Register (MUX_14_DIV_UPD_STAT)

Offset

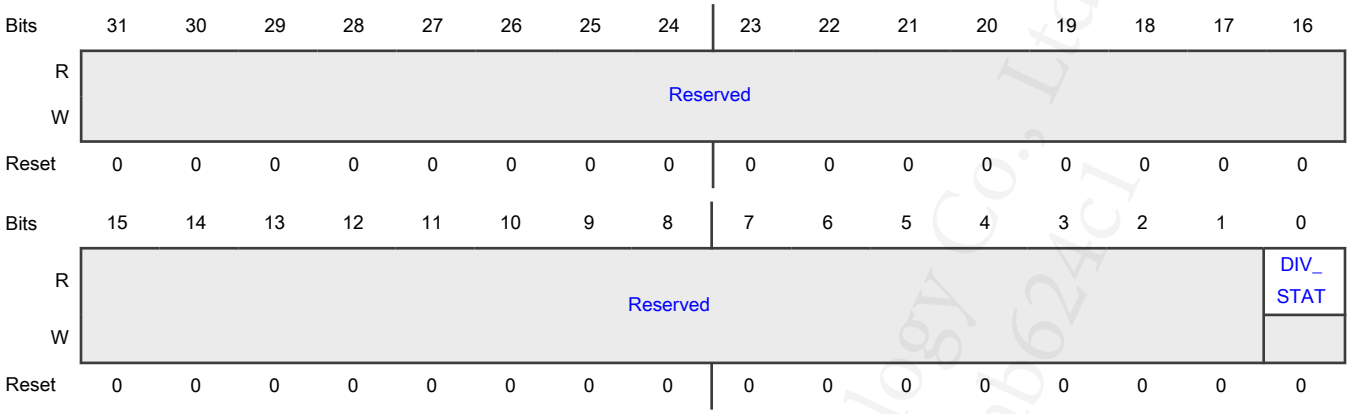
Register	Offset
MUX_14_DIV_UPD_STAT	6BCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 14. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other

clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 14 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.57 Clock Mux 15 Select Control Register (MUX_15_CSC)

Offset

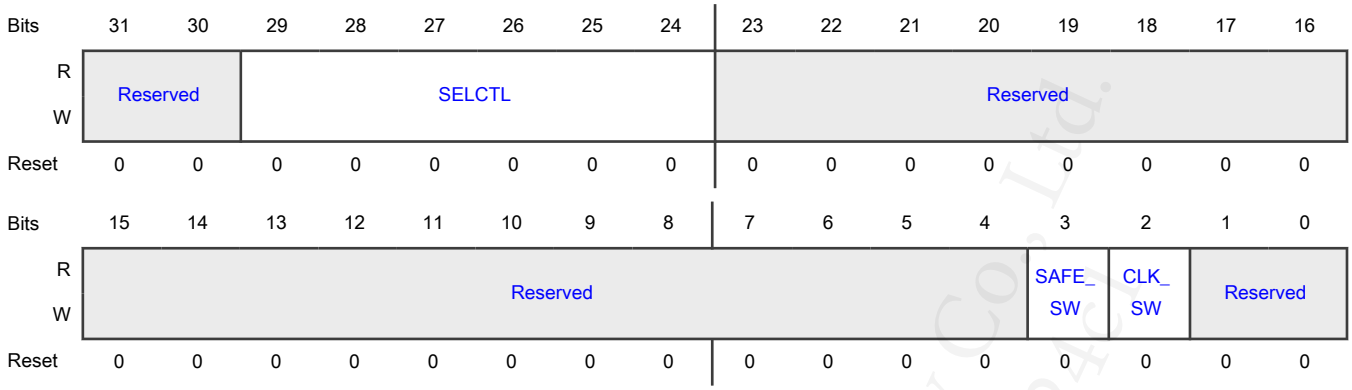
Register	Offset
MUX_15_CSC	6C0h

Function

This register provides the clock source selection control for clock mux 15. Clock mux 15 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 15. The reserved values are not displayed. 000000b - clk_src_0 100111b - clk_src_39
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 15. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.5.58 Clock Mux 15 Select Status Register (MUX_15_CSS)

Offset

Register	Offset
MUX_15_CSS	6C4h

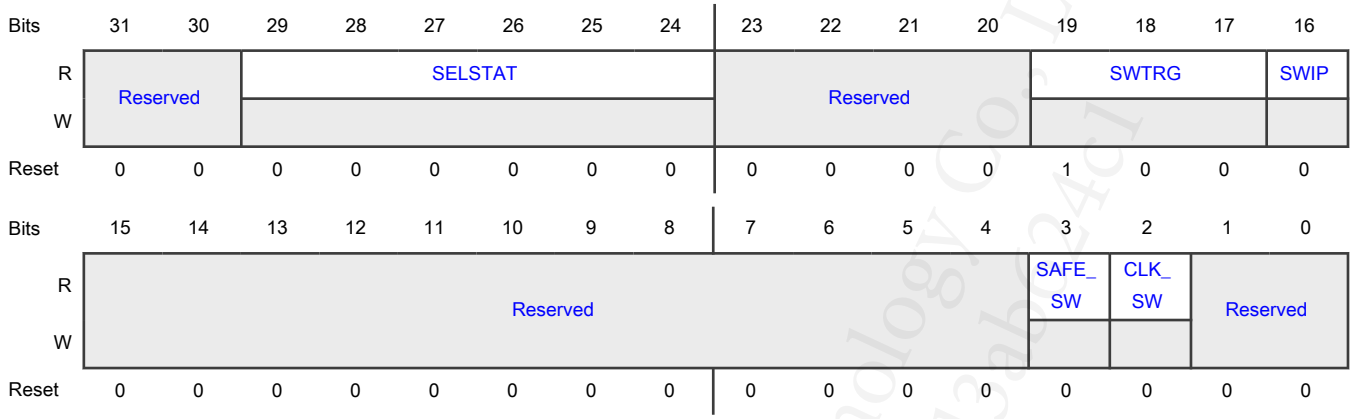
Provided under NDA only
COMPANY PROPRIETARY
Beijing C10 Automotive Technology Co., Ltd.
c8c4625_492_4a8-b251-390d3ab624c

Function

This register provides the current clock source selection status for clock mux 15.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 15. The reserved values are not displayed. 000000b - clk_src_0 100111b - clk_src_39
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 15. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 15. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.5.59 Clock Mux 15 Divider 0 Control Register (MUX_15_DC_0)

Offset

Register	Offset
MUX_15_DC_0	6C8h

Function

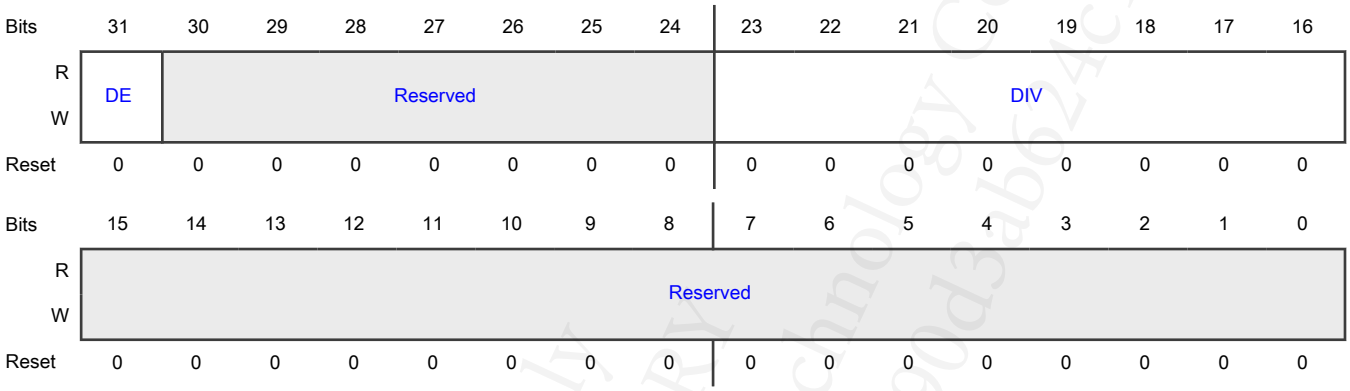
This register controls the clock divider 0 for clock mux 15.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.5.60 Clock Mux 15 Divider Update Status Register (MUX_15_DIV_UPD_STAT)

Offset

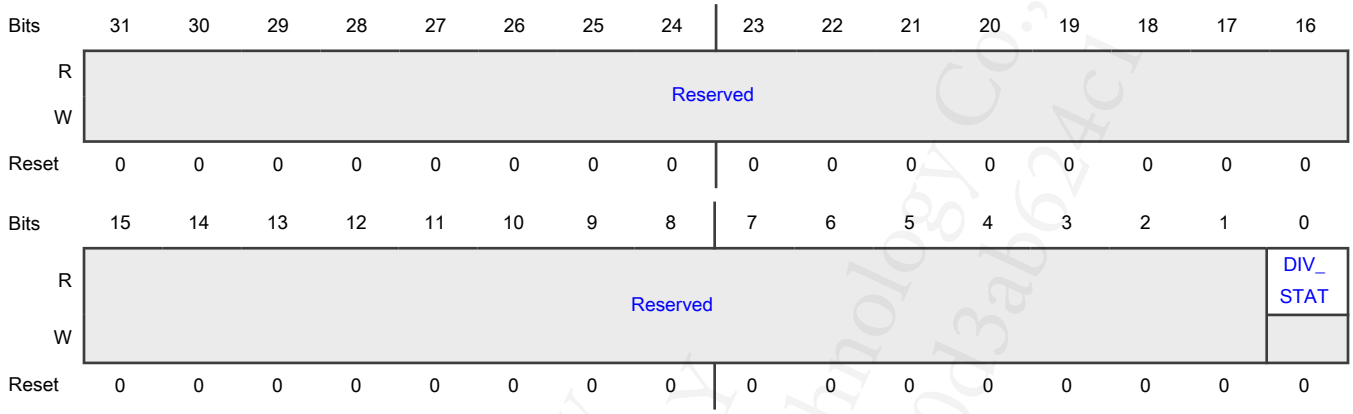
Register	Offset
MUX_15_DIV_UPD_STAT	6FCh

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46255-4a2-48a8-b251-390d3ab624c1

Function

This register provides the update status of the clock dividers corresponding to clock mux 15. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 15 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.5.61 Clock Mux 16 Select Control Register (MUX_16_CSC)

Offset

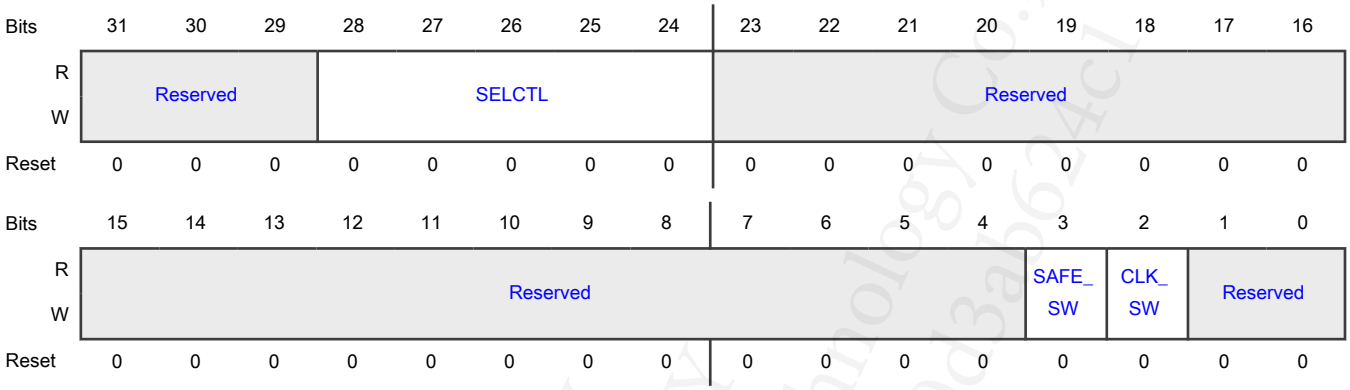
Register	Offset
MUX_16_CSC	700h

Function

This register provides the clock source selection control for clock mux 16. Clock mux 16 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELCTL	Clock source selection control Selects the source clock for clock mux 16. The reserved values are not displayed. 00000b - clk_src_0 11001b - clk_src_25
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 16. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

Printed under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca26c5c4-821-48a8-b251-390a3ab624c1

24.5.62 Clock Mux 16 Select Status Register (MUX_16_CSS)

Offset

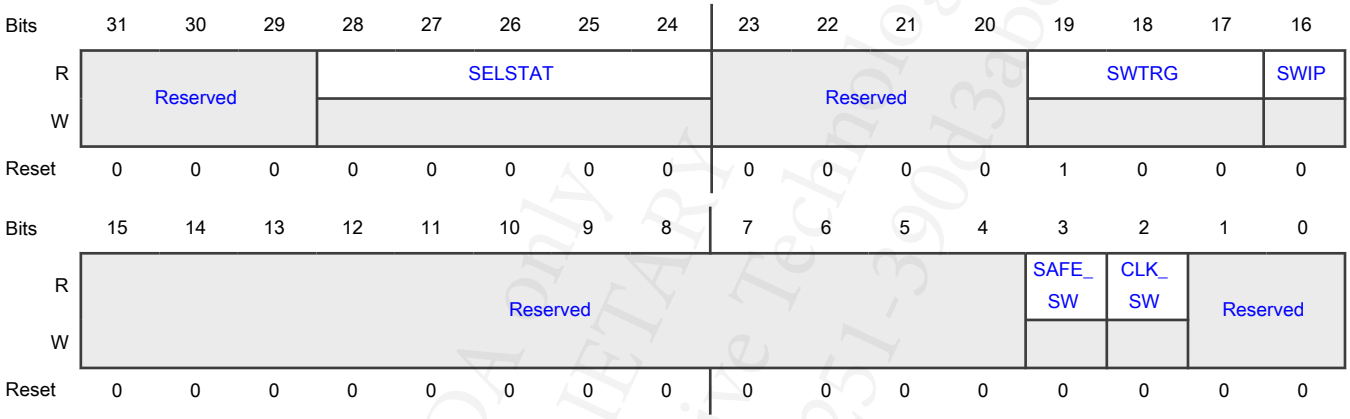
Register	Offset
MUX_16_CSS	704h

Function

This register provides the current clock source selection status for clock mux 16.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-29 —	This field is reserved and reads return zeros.
28-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 16. The reserved values are not displayed. 00000b - clk_src_0 11001b - clk_src_25
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 16. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 16. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.6 MC_CGM_1 register descriptions

MC_CGM implements a set of clock multiplexers that share PCFS configuration registers. MC_CGM registers have the following properties:

- All registers are 32-bit wide.
- Only 32-bit read and write accesses are supported.
- Read/write accesses of less than 32 bits do not terminate with an error, and hardware behavior is unpredictable.
- Writes to read-only register fields in writable registers are ignored and do not provide an error response.
- Writes to read-only registers are aborted with an error response.

24.6.1 MC_CGM_1 memory map

MC_CGM_1 base address: 4003_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PCFS Step Duration (PCFS_SDUR)	32	RW	0000_0000h
28h	PCFS Divider Change 4 Register (PCFS_DIVC4)	32	RW	0000_0000h
2Ch	PCFS Divider End 4 Register (PCFS_DIVE4)	32	RW	0000_03E7h
30h	PCFS Divider Start 4 Register (PCFS_DIVS4)	32	RW	0000_03E7h
300h	Clock Mux 0 Select Control Register (MUX_0_CSC)	32	RW	0000_0000h
304h	Clock Mux 0 Select Status Register (MUX_0_CSS)	32	RO	0008_0000h

24.6.2 PCFS Step Duration (PCFS_SDUR)

Offset

Register	Offset
PCFS_SDUR	0h

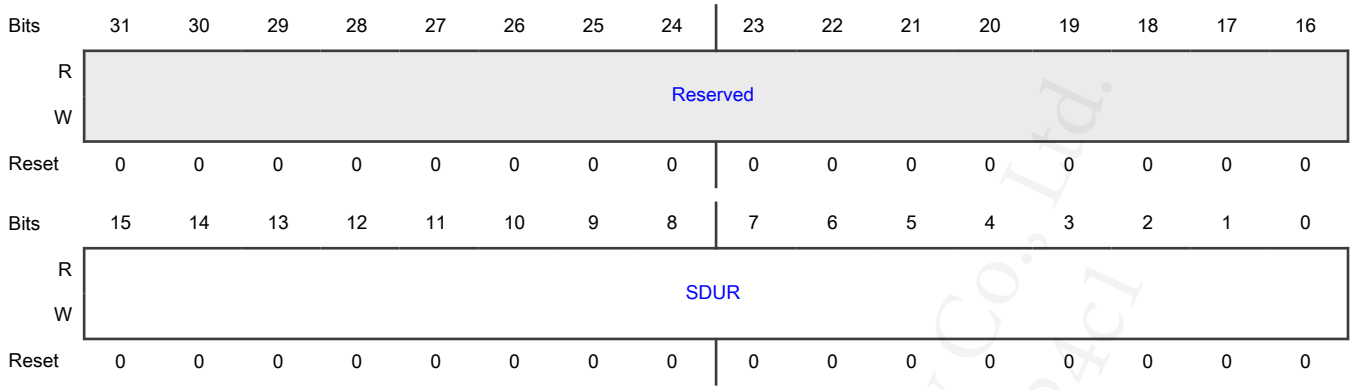
Function

This register specifies the step duration of each PCFS step. The value provided in this register specifies the PCFS step duration in terms of the number of cycles of clk_src_0.

This register is reset only by a destructive reset. For details, see PCFS.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHU Automotive Technology Co., Ltd.
c8c462e54621-48a8-b251-29063a624c1

Diagram



Fields

Field	Function
31-16 —	This field is reserved and reads return zeros.
15-0 SDUR	Step duration Count value of the step duration

24.6.3 PCFS Divider Change 4 Register (PCFS_DIVC4)

Offset

Register	Offset
PCFS_DIVC4	28h

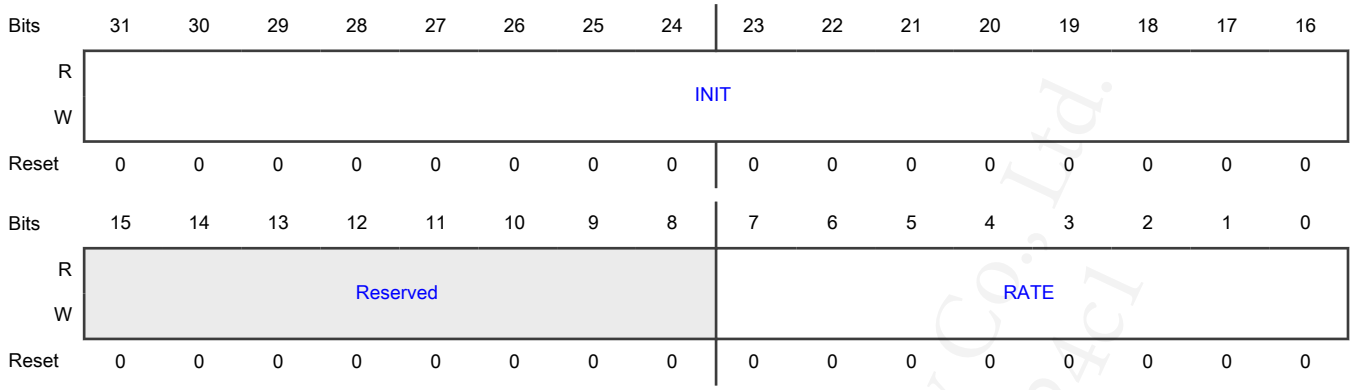
Function

This register defines the rate of frequency change and initial change value on frequency ramp-up for the Progressive Clock Frequency switching of clk_src_4.

This register is reset only on destructive reset.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-16 INIT	Divider change initial value This field provides the initial change value of the clock divider for the clock ramp-up phase of clk_src_4.
15-8 —	This field is reserved and reads return zeros.
7-0 RATE	Divider change rate This value controls the change value of the clock divider for the clock ramp-up and ramp-down phase of clk_src_4.

24.6.4 PCFS Divider End 4 Register (PCFS_DIVE4)

Offset

Register	Offset
PCFS_DIVE4	2Ch

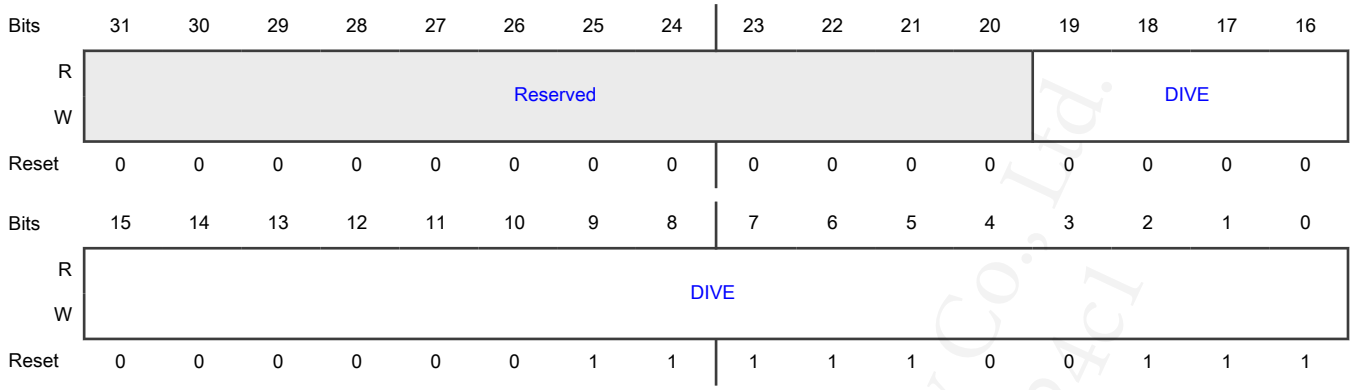
Function

This register defines the final division value on frequency ramp-down for the progressive system clock switching of clk_src_4.

This registers is reset only on destructive reset.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-1821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-20 —	This field is reserved and reads return zeros.
19-0 DIVE	Divider end value This field provides the end value of the clock divider for the clk_src_4 ramp-down phase.

24.6.5 PCFS Divider Start 4 Register (PCFS_DIVS4)

Offset

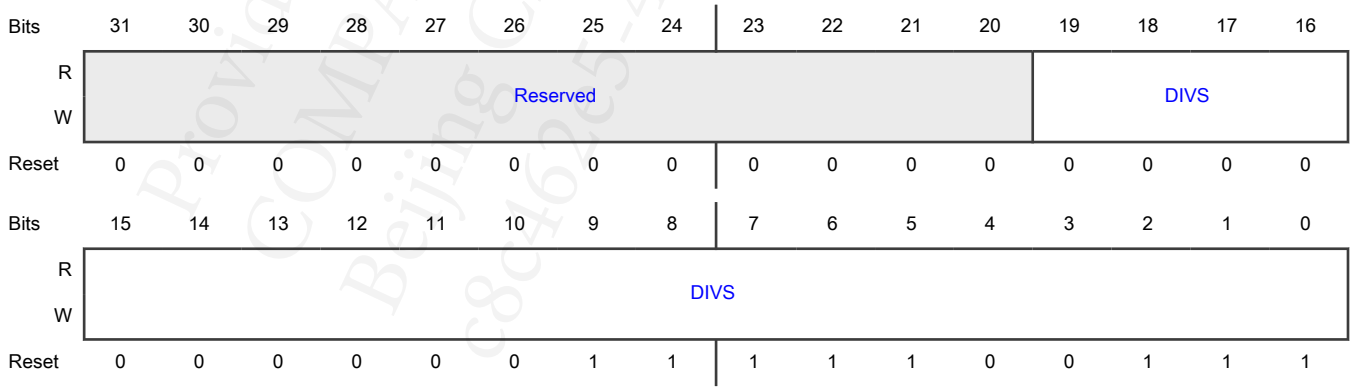
Register	Offset
PCFS_DIVS4	30h

Function

This register defines the initial division value on frequency ramp-up for the progressive system clock switching of clk_src_4.

This register is reset only on destructive reset.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8d462e5-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-20 —	This field is reserved and reads return zeros.
19-0 DIVS	Divider start value This field provides the start value of the clock divider for the clk_src_4 ramp-up phase

24.6.6 Clock Mux 0 Select Control Register (MUX_0_CSC)

Offset

Register	Offset
MUX_0_CSC	300h

Function

This register provides the clock source selection control for clock mux 0. Clock mux 0 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

An update to all the PCFS-related fields of this register must be an atomic write, which means a single write must update the CLK_SW, RAMPDOWN, and RAMPUP fields. It is necessary to set both RAMPUP and RAMPDOWN bits together even if you want to trigger either RAMPUP or RAMPDOWN process otherwise the desired PCFS sequence will not be executed.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				SELCTL				Reserved							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												SAFE_SW	CLK_SW	RAMPDOWN	RAMPUP
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-24 SELCTL	Clock source selection control Selects the source clock for clock mux 0. The reserved values are not displayed. 0000b - clk_src_0 0100b - clk_src_4 1101b - clk_src_13
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 0. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1 RAMPDOWN	PCFS ramp-down Writing 1 to this bit makes a PCFS ramp-down request to clock mux 0. After a PCFS ramp-down operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
0 RAMPUP	PCFS ramp-up Writing 1 to this bit makes a PCFS ramp-up request to clock mux 0. After a PCFS ramp-up operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.

24.6.7 Clock Mux 0 Select Status Register (MUX_0_CSS)

Offset

Register	Offset
MUX_0_CSS	304h

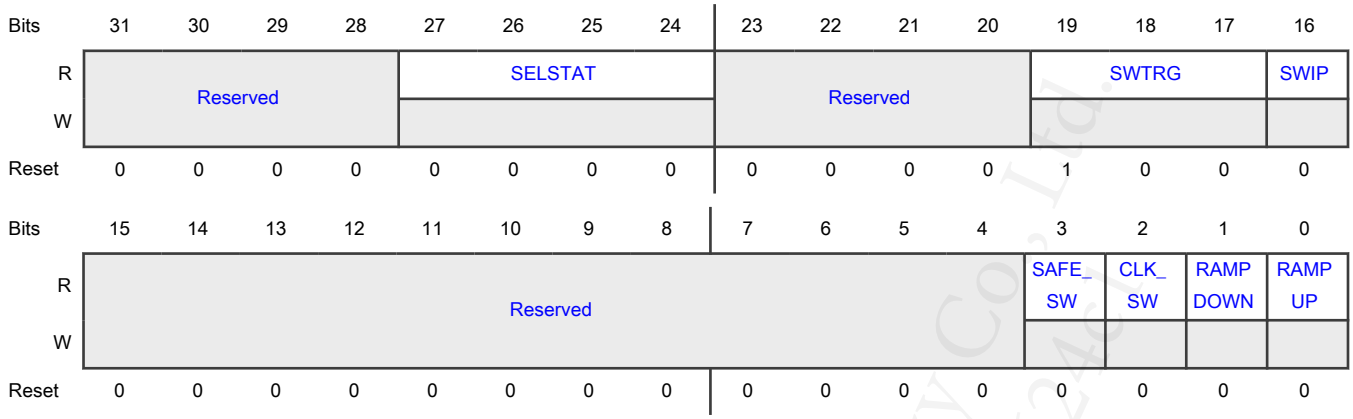
Function

This register provides the current clock source selection status for clock mux 0.

This register is reset on destructive reset only.

Provided by IHS Markit
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-4a8b-23_3-0a5b624c1

Diagram



Fields

Field	Function
31-28 —	This field is reserved and reads return zeros.
27-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 0. The reserved values are not displayed. 0000b - clk_src_0 0100b - clk_src_4 1101b - clk_src_13
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 0. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 0. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1 RAMPDOWN	PCFS ramp-down This field provides an indication of whether a PCFS ramp-down operation was requested during the previous/ongoing request on clock mux 0. **NOTE** In case of safe clock switching, the ramp-down operation runs internally, but the value of the corresponding status field is not set to 1. 0b - No ramp-down operation was requested. 1b - Ramp-down operation was requested.
0 RAMPUP	PCFS ramp-up This field provides an indication of whether a PCFS ramp-up operation was requested during the previous/ongoing request on clock mux 0. 0b - No ramp-up operation was requested. 1b - Ramp-up operation was requested.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CAL Automotive Technology Co., Ltd.
CS-626542 1-4848-b251-390d3ab6240d

24.7 MC_CGM_2 register descriptions

MC_CGM implements a set of clock multiplexers that share PCFS configuration registers. MC_CGM registers have the following properties:

- All registers are 32-bit wide.
- Only 32-bit read and write accesses are supported.
- Read/write accesses of less than 32 bits do not terminate with an error, and hardware behavior is unpredictable.
- Writes to read-only register fields in writable registers are ignored and do not provide an error response.
- Writes to read-only registers are aborted with an error response.

24.7.1 MC_CGM_2 memory map

MC_CGM_2 base address: 4401_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PCFS Step Duration (PCFS_SDUR)	32	RW	0000_0000h
184h	PCFS Divider Change 33 Register (PCFS_DIVC33)	32	RW	0000_0000h
188h	PCFS Divider End 33 Register (PCFS_DIVE33)	32	RW	0000_03E7h
18Ch	PCFS Divider Start 33 Register (PCFS_DIVS33)	32	RW	0000_03E7h
300h	Clock Mux 0 Select Control Register (MUX_0_CSC)	32	RW	0000_0000h
304h	Clock Mux 0 Select Status Register (MUX_0_CSS)	32	RO	0008_0000h
308h	Clock Mux 0 Divider 0 Control Register (MUX_0_DC_0)	32	RW	8001_0000h
334h	Clock Mux 0 Divider Trigger Control Register (MUX_0_DIV_TRIG_CTRL)	32	RW	0000_0000h
338h	Clock Mux 0 Divider Trigger Register (MUX_0_DIV_TRIG)	32	WORZ	0000_0000h
33Ch	Clock Mux 0 Divider Update Status Register (MUX_0_DIV_UPD_STAT)	32	RO	0000_0000h
340h	Clock Mux 1 Select Control Register (MUX_1_CSC)	32	RW	0000_0000h
344h	Clock Mux 1 Select Status Register (MUX_1_CSS)	32	RO	0008_0000h
348h	Clock Mux 1 Divider 0 Control Register (MUX_1_DC_0)	32	RW	0000_0000h
34Ch	Clock Mux 1 Divider 1 Control Register (MUX_1_DC_1)	32	RW	0003_0000h
374h	Clock Mux 1 Divider Trigger Control Register (MUX_1_DIV_TRIG_CTRL)	32	RW	0000_0000h
378h	Clock Mux 1 Divider Trigger Register (MUX_1_DIV_TRIG)	32	WORZ	0000_0000h
37Ch	Clock Mux 1 Divider Update Status Register (MUX_1_DIV_UPD_STAT)	32	RO	0000_0000h
380h	Clock Mux 2 Select Control Register (MUX_2_CSC)	32	RW	0000_0000h
384h	Clock Mux 2 Select Status Register (MUX_2_CSS)	32	RO	0008_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
388h	Clock Mux 2 Divider 0 Control Register (MUX_2_DC_0)	32	RW	0000_0000h
38Ch	Clock Mux 2 Divider 1 Control Register (MUX_2_DC_1)	32	RW	0003_0000h
3B4h	Clock Mux 2 Divider Trigger Control Register (MUX_2_DIV_TRIG_CTRL)	32	RW	0000_0000h
3B8h	Clock Mux 2 Divider Trigger Register (MUX_2_DIV_TRIG)	32	WORZ	0000_0000h
3BCh	Clock Mux 2 Divider Update Status Register (MUX_2_DIV_UPD_STAT)	32	RO	0000_0000h
3C0h	Clock Mux 3 Select Control Register (MUX_3_CSC)	32	RW	0000_0000h
3C4h	Clock Mux 3 Select Status Register (MUX_3_CSS)	32	RO	0008_0000h
3C8h	Clock Mux 3 Divider 0 Control Register (MUX_3_DC_0)	32	RW	0000_0000h
3CCh	Clock Mux 3 Divider 1 Control Register (MUX_3_DC_1)	32	RW	0003_0000h
3F4h	Clock Mux 3 Divider Trigger Control Register (MUX_3_DIV_TRIG_CTRL)	32	RW	0000_0000h
3F8h	Clock Mux 3 Divider Trigger Register (MUX_3_DIV_TRIG)	32	WORZ	0000_0000h
3FCh	Clock Mux 3 Divider Update Status Register (MUX_3_DIV_UPD_STAT)	32	RO	0000_0000h
400h	Clock Mux 4 Select Control Register (MUX_4_CSC)	32	RW	0000_0000h
404h	Clock Mux 4 Select Status Register (MUX_4_CSS)	32	RO	0008_0000h
440h	Clock Mux 5 Select Control Register (MUX_5_CSC)	32	RW	0000_0000h
444h	Clock Mux 5 Select Status Register (MUX_5_CSS)	32	RO	0008_0000h
480h	Clock Mux 6 Select Control Register (MUX_6_CSC)	32	RW	0000_0000h
484h	Clock Mux 6 Select Status Register (MUX_6_CSS)	32	RO	0008_0000h
4C0h	Clock Mux 7 Select Control Register (MUX_7_CSC)	32	RW	0000_0000h
4C4h	Clock Mux 7 Select Status Register (MUX_7_CSS)	32	RO	0008_0000h
4C8h	Clock Mux 7 Divider 0 Control Register (MUX_7_DC_0)	32	RW	0000_0000h
4F4h	Clock Mux 7 Divider Trigger Control Register (MUX_7_DIV_TRIG_CTRL)	32	RW	0000_0000h
4F8h	Clock Mux 7 Divider Trigger Register (MUX_7_DIV_TRIG)	32	WORZ	0000_0000h
4FCh	Clock Mux 7 Divider Update Status Register (MUX_7_DIV_UPD_STAT)	32	RO	0000_0000h
500h	Clock Mux 8 Select Control Register (MUX_8_CSC)	32	RW	0000_0000h
504h	Clock Mux 8 Select Status Register (MUX_8_CSS)	32	RO	0008_0000h
508h	Clock Mux 8 Divider 0 Control Register (MUX_8_DC_0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
534h	Clock Mux 8 Divider Trigger Control Register (MUX_8_DIV_TRIG_CTRL)	32	RW	0000_0000h
538h	Clock Mux 8 Divider Trigger Register (MUX_8_DIV_TRIG)	32	WORZ	0000_0000h
53Ch	Clock Mux 8 Divider Update Status Register (MUX_8_DIV_UPD_STAT)	32	RO	0000_0000h
540h	Clock Mux 9 Select Control Register (MUX_9_CSC)	32	RW	0000_0000h
544h	Clock Mux 9 Select Status Register (MUX_9_CSS)	32	RO	0008_0000h
548h	Clock Mux 9 Divider 0 Control Register (MUX_9_DC_0)	32	RW	0000_0000h
574h	Clock Mux 9 Divider Trigger Control Register (MUX_9_DIV_TRIG_CTRL)	32	RW	0000_0000h
578h	Clock Mux 9 Divider Trigger Register (MUX_9_DIV_TRIG)	32	WORZ	0000_0000h
57Ch	Clock Mux 9 Divider Update Status Register (MUX_9_DIV_UPD_STAT)	32	RO	0000_0000h

24.7.2 PCFS Step Duration (PCFS_SDUR)

Offset

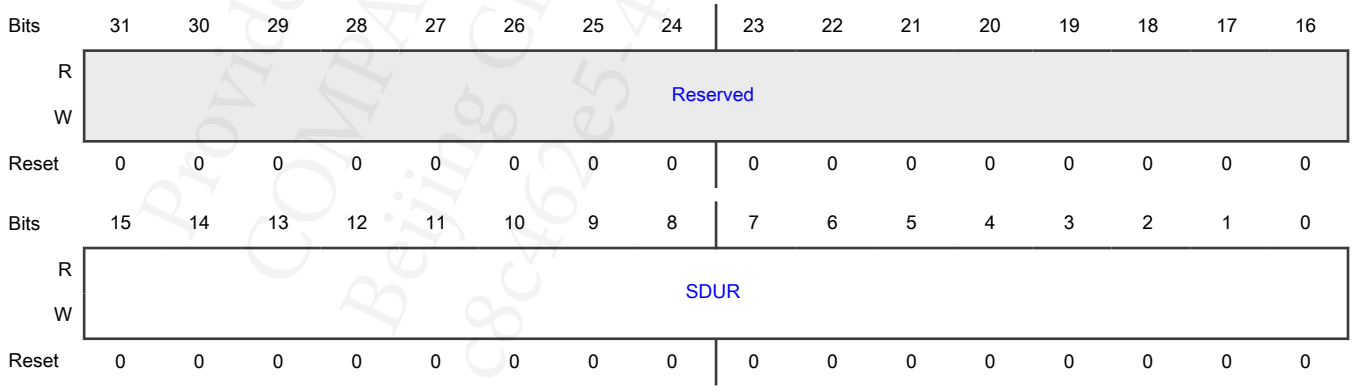
Register	Offset
PCFS_SDUR	0h

Function

This register specifies the step duration of each PCFS step. The value provided in this register specifies the PCFS step duration in terms of the number of cycles of clk_src_0.

This register is reset only by a destructive reset. For details, see PCFS.

Diagram



Fields

Field	Function
31-16 —	This field is reserved and reads return zeros.
15-0 SDUR	Step duration Count value of the step duration

24.7.3 PCFS Divider Change 33 Register (PCFS_DIVC33)

Offset

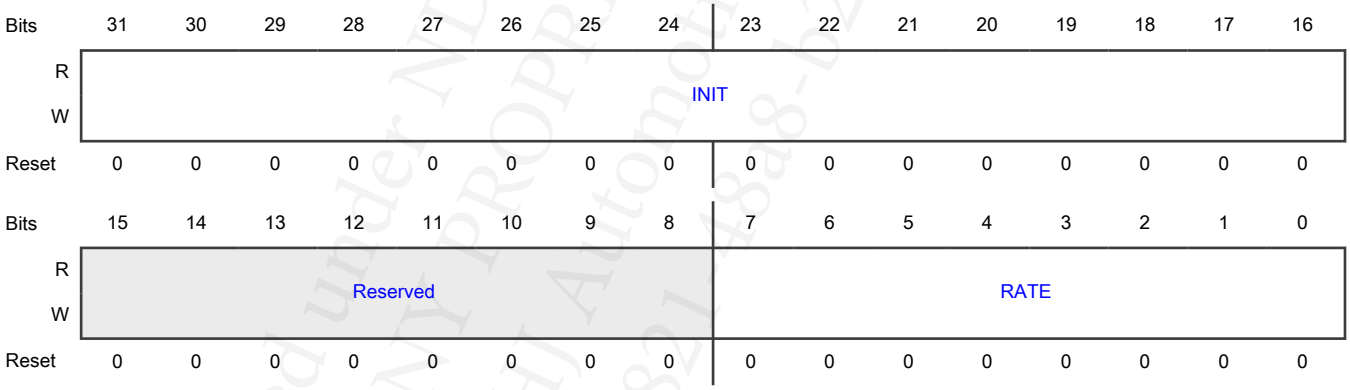
Register	Offset
PCFS_DIVC33	184h

Function

This register defines the rate of frequency change and initial change value on frequency ramp-up for the Progressive Clock Frequency switching of clk_src_33.

This register is reset only on destructive reset.

Diagram



Fields

Field	Function
31-16 INIT	Divider change initial value This field provides the initial change value of the clock divider for the clock ramp-up phase of clk_src_33.
15-8 —	This field is reserved and reads return zeros.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
7-0 RATE	Divider change rate This value controls the change value of the clock divider for the clock ramp-up and ramp-down phase of clk_src_33.

24.7.4 PCFS Divider End 33 Register (PCFS_DIVE33)

Offset

Register	Offset
PCFS_DIVE33	188h

Function

This register defines the final division value on frequency ramp-down for the progressive system clock switching of clk_src_33.

This registers is reset only on destructive reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved												DIVE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DIVE															
Reset	0	0	0	0	0	0	1	1	1	1	1	0	0	1	1	1

Fields

Field	Function
31-20 —	This field is reserved and reads return zeros.
19-0 DIVE	Divider end value This field provides the end value of the clock divider for the clk_src_33 ramp-down phase.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462c54821-48a8-b251-3901-3ab624c1

24.7.5 PCFS Divider Start 33 Register (PCFS_DIVS33)

Offset

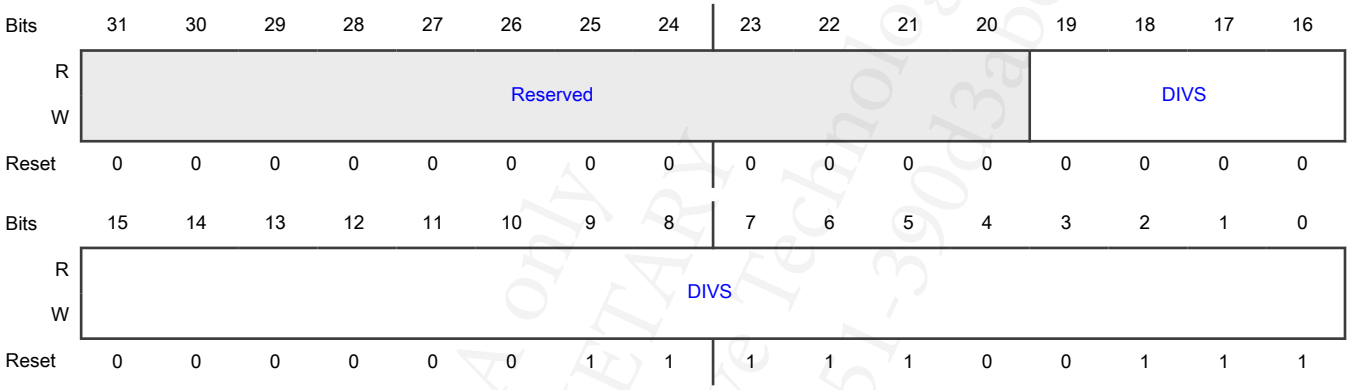
Register	Offset
PCFS_DIVS33	18Ch

Function

This register defines the initial division value on frequency ramp-up for the progressive system clock switching of clk_src_33.

This register is reset only on destructive reset.

Diagram



Fields

Field	Function
31-20 —	This field is reserved and reads return zeros.
19-0 DIVS	Divider start value This field provides the start value of the clock divider for the clk_src_33 ramp-up phase

24.7.6 Clock Mux 0 Select Control Register (MUX_0_CSC)

Offset

Register	Offset
MUX_0_CSC	300h

Function

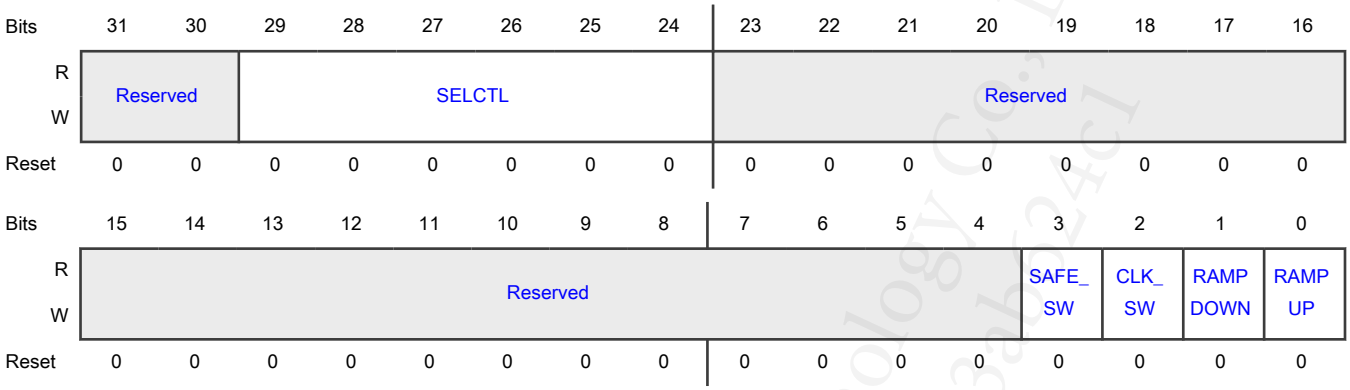
This register provides the clock source selection control for clock mux 0. Clock mux 0 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CTI Automotive Technology Co., Ltd.
8c462e5-4821-48a8-b251-39043a62401

This register is reset on destructive reset only.

An update to all the PCFS-related fields of this register must be an atomic write, which means a single write must update the CLK_SW, RAMPDOWN, and RAMPUP fields. It is necessary to set both RAMPUP and RAMPDOWN bits together even if you want to trigger either RAMPUP or RAMPDOWN process otherwise the desired PCFS sequence will not be executed.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 0. The reserved values are not displayed. 000000b - clk_src_0 100001b - clk_src_33
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 0. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1 RAMPDOWN	PCFS ramp-down Writing 1 to this bit makes a PCFS ramp-down request to clock mux 0. After a PCFS ramp-down operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
0 RAMPUP	PCFS ramp-up Writing 1 to this bit makes a PCFS ramp-up request to clock mux 0. After a PCFS ramp-up operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.

24.7.7 Clock Mux 0 Select Status Register (MUX_0_CSS)

Offset

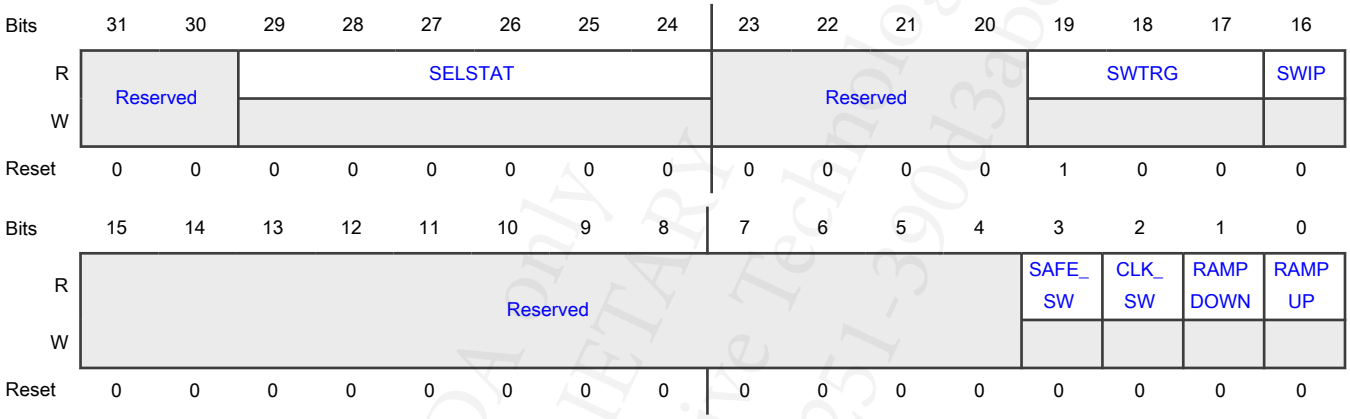
Register	Offset
MUX_0_CSS	304h

Function

This register provides the current clock source selection status for clock mux 0.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 0. The reserved values are not displayed. 000000b - clk_src_0 100001b - clk_src_33
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
ca162e5-4821-48a8-b251-390d3a624c1

Table continued from the previous page...

Field	Function
	NOTE If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 0. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ongoing request on clock mux 0. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1 RAMPDOWN	PCFS ramp-down

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field provides an indication of whether a PCFS ramp-down operation was requested during the previous/ongoing request on clock mux 0. **NOTE** In case of safe clock switching, the ramp-down operation runs internally, but the value of the corresponding status field is not set to 1. 0b - No ramp-down operation was requested. 1b - Ramp-down operation was requested.
0 RAMPUP	PCFS ramp-up This field provides an indication of whether a PCFS ramp-up operation was requested during the previous/ ongoing request on clock mux 0. 0b - No ramp-up operation was requested. 1b - Ramp-up operation was requested.

24.7.8 Clock Mux 0 Divider 0 Control Register (MUX_0_DC_0)

Offset

Register	Offset
MUX_0_DC_0	308h

Function

This register controls the clock divider 0 for clock mux 0.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DE	Reserved											DIV			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-20 —	This field is reserved and reads return zeros.
19-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.7.9 Clock Mux 0 Divider Trigger Control Register (MUX_0_DIV_TRIG_CTRL)

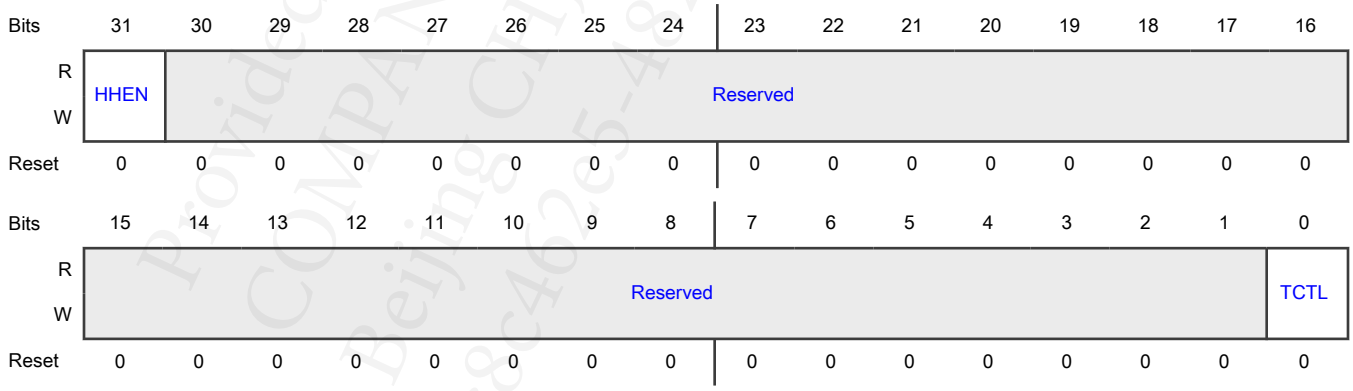
Offset

Register	Offset
MUX_0_DIV_TRIG_CTRL	334h

Function

This register selects whether the dividers associated with clock mux 0 are updated immediately on writing to the corresponding divider configuration register (referred to as immediate divider update) or only on writing to the MC_CGM_MUX_0_DIV_TRIG register (referred to as common trigger update). When common trigger update is configured, this register also controls initiation of the halt handshake protocol with the on-chip AXBS. Software is required to configure HHEN field for handshaking with on-chip AXBS when the ratio of division value among the clock dividers need to be changed.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a48-b251-390d3ab604c1

Fields

Field	Function
31 HHEN	Halt handshake enable This field controls the initiation of the halt handshake protocol with AXBS when a common trigger divider update is initiated. 0b - No halt handshake protocol is initiated. 1b - Halt handshake protocol is initiated.
30-1 —	This field is reserved and reads return zeros.
0 TCTL	Trigger control This field controls the divider update configuration between immediate and common update. 0b - Immediate divider update 1b - Common trigger divider update

24.7.10 Clock Mux 0 Divider Trigger Register (MUX_0_DIV_TRIG)

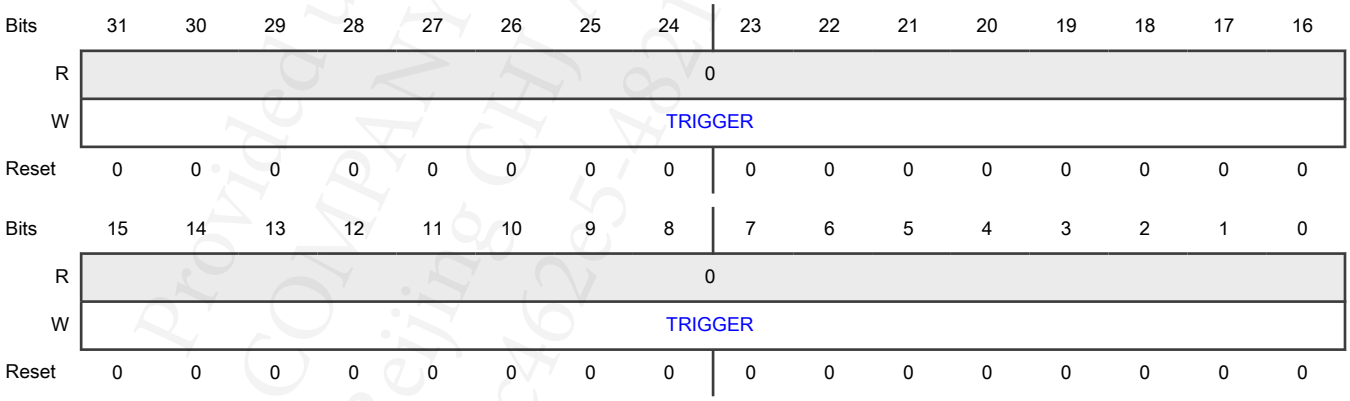
Offset

Register	Offset
MUX_0_DIV_TRIG	338h

Function

This register provides a common trigger for the clock dividers (only 50% duty cycle dividers) of clock mux 0. Writing any value to this register provides a trigger to the dividers. This register should only be written after appropriately configuring the MC_CGM_MUX_0_DIV_TRIG_CTRL register.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c16255-4821-4808-b251-390d3ab624c1

Fields

Field	Function
31-0 TRIGGER	Trigger for divider update

24.7.11 Clock Mux 0 Divider Update Status Register (MUX_0_DIV_UPD_STAT)

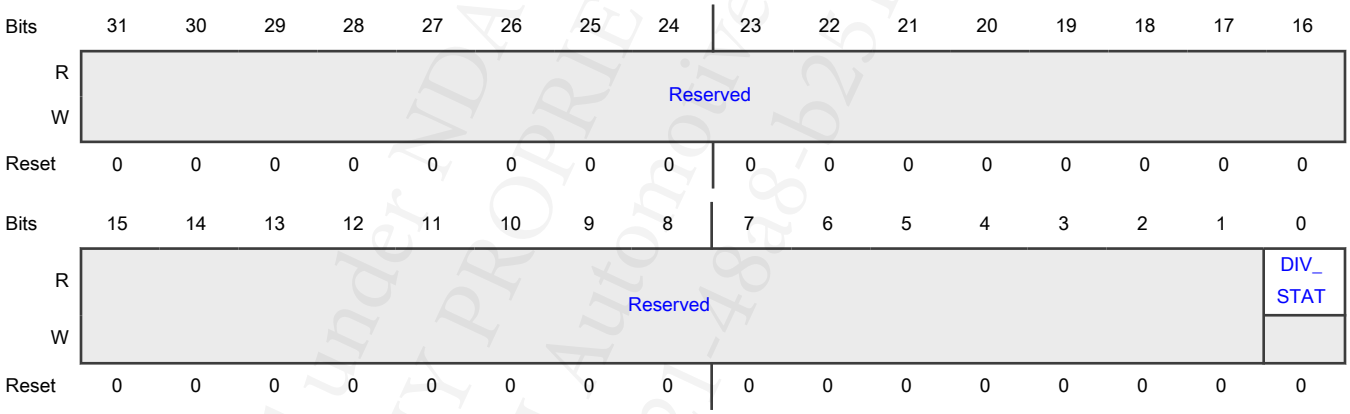
Offset

Register	Offset
MUX_0_DIV_UPD_STAT	33Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 0. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 0 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.7.12 Clock Mux 1 Select Control Register (MUX_1_CSC)

Offset

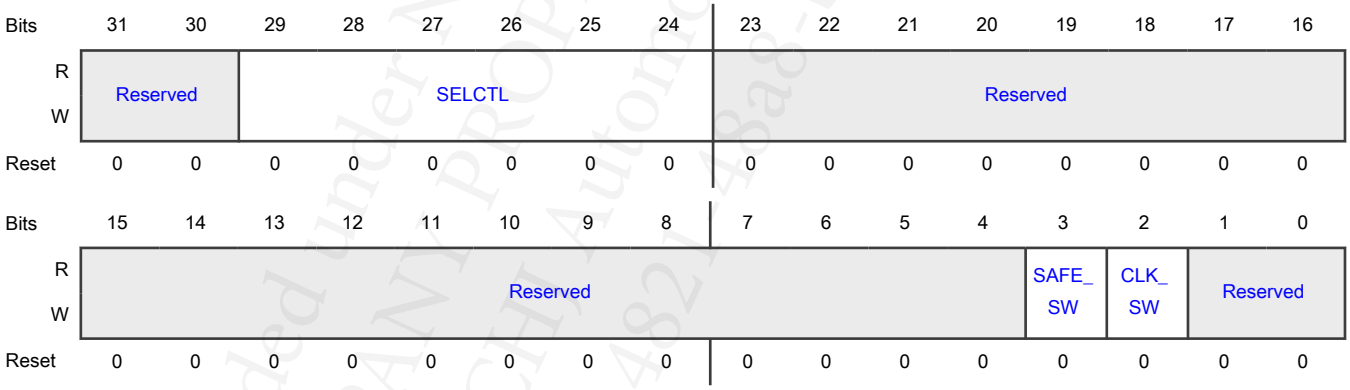
Register	Offset
MUX_1_CSC	340h

Function

This register provides the clock source selection control for clock mux 1. Clock mux 1 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24	Clock source selection control

Table continues on the next page...

Table continued from the previous page...

Field	Function
SELCTL	Selects the source clock for clock mux 1. The reserved values are not displayed. 000000b - clk_src_0 010111b - clk_src_23 110000b - clk_src_48 110010b - clk_src_50 111001b - clk_src_57
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 1. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.7.13 Clock Mux 1 Select Status Register (MUX_1_CSS)

Offset

Register	Offset
MUX_1_CSS	344h

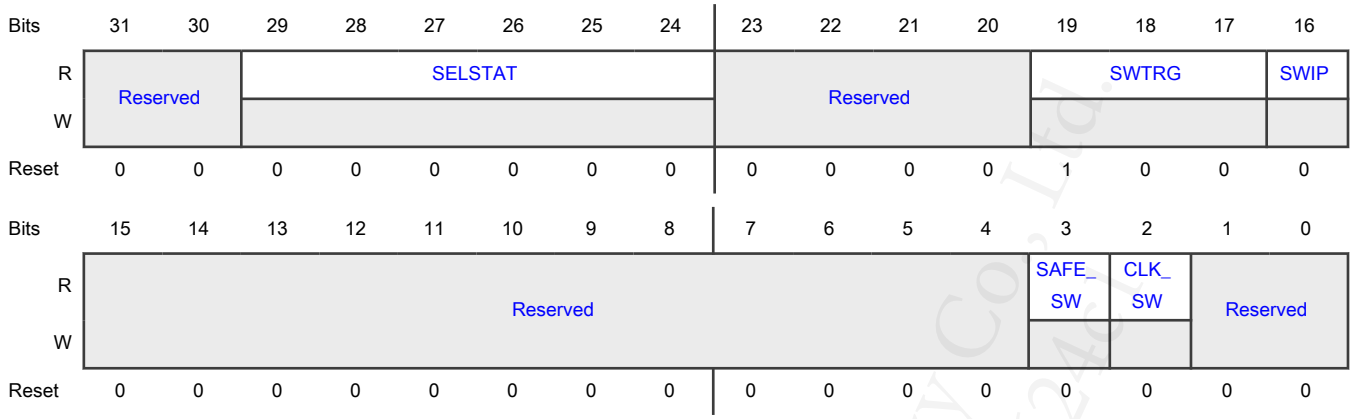
Function

This register provides the current clock source selection status for clock mux 1.

This register is reset on destructive reset only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-90cab624c1

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 1. The reserved values are not displayed. 000000b - clk_src_0 010111b - clk_src_23 110000b - clk_src_48 110010b - clk_src_50 111001b - clk_src_57
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 1. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 1. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.7.14 Clock Mux 1 Divider 0 Control Register (MUX_1_DC_0)

Offset

Register	Offset
MUX_1_DC_0	348h

Function

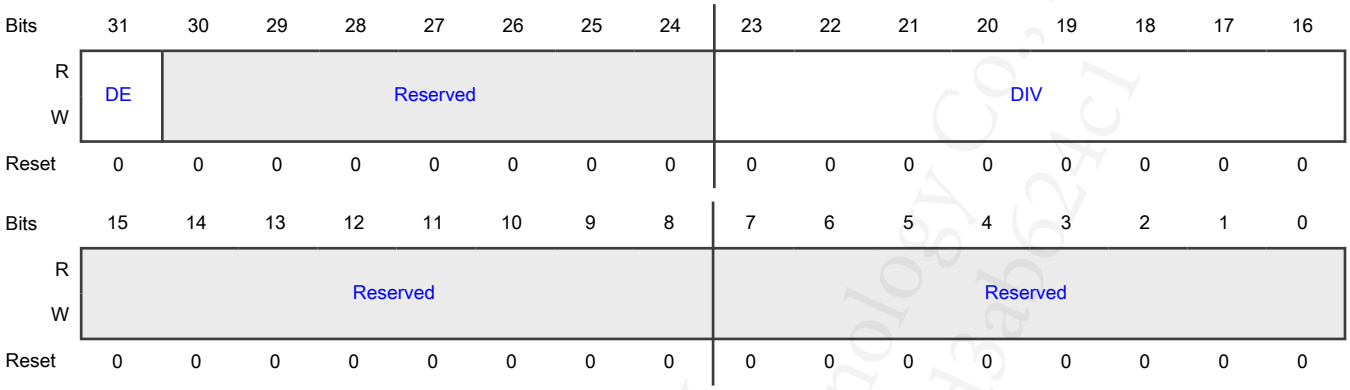
This register controls the clock divider 0 for clock mux 1.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-8 —	This field is reserved and reads return zeros.
7-0 —	This field is reserved and reads return zeros.

24.7.15 Clock Mux 1 Divider 1 Control Register (MUX_1_DC_1)

Offset

Register	Offset
MUX_1_DC_1	34Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJU Automotive Technology Co., Ltd.
c8c46255-4821-8a8-b251-390d3ab624c1

Function

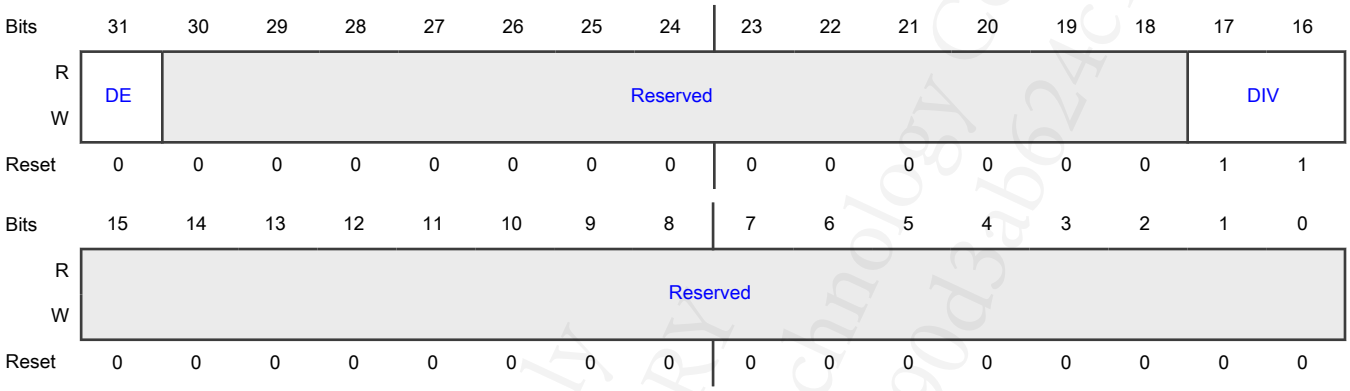
This register controls the clock divider 1 for clock mux 1.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-18 —	This field is reserved and reads return zeros.
17-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.7.16 Clock Mux 1 Divider Trigger Control Register (MUX_1_DIV_TRIG_CTRL)

Offset

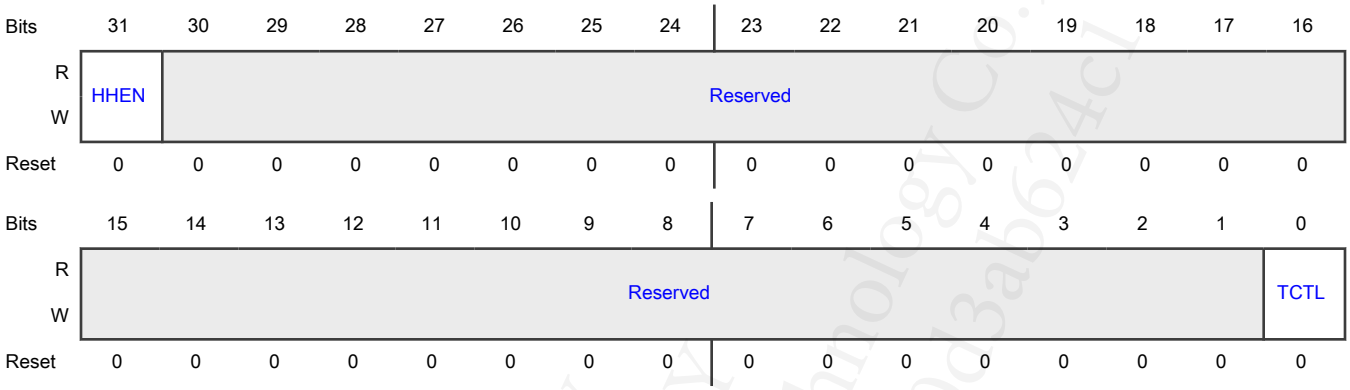
Register	Offset
MUX_1_DIV_TRIG_CTRL	374h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46255-462a-48a8-b251-390d3ab624c1

Function

This register selects whether the dividers associated with clock mux 1 are updated immediately on writing to the corresponding divider configuration register (referred to as immediate divider update) or only on writing to the MC_CGM_MUX_1_DIV_TRIG register (referred to as common trigger update). When common trigger update is configured, this register also controls initiation of the halt handshake protocol with the on-chip AXBS. Software is required to configure HHEN field for handshaking with on-chip AXBS when the ratio of division value among the clock dividers need to be changed.

Diagram



Fields

Field	Function
31 HHEN	Halt handshake enable This field controls the initiation of the halt handshake protocol with AXBS when a common trigger divider update is initiated. 0b - No halt handshake protocol is initiated. 1b - Halt handshake protocol is initiated.
30-1 —	This field is reserved and reads return zeros.
0 TCTL	Trigger control This field controls the divider update configuration between immediate and common update. 0b - Immediate divider update 1b - Common trigger divider update

24.7.17 Clock Mux 1 Divider Trigger Register (MUX_1_DIV_TRIG)

Offset

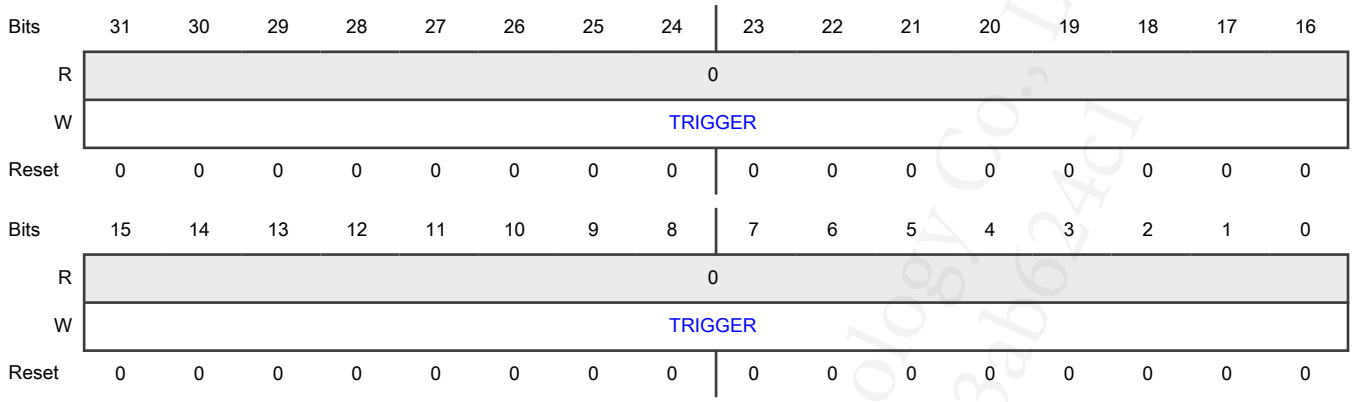
Register	Offset
MUX_1_DIV_TRIG	378h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46255-8181-48a8-b251-3903ab614c1

Function

This register provides a common trigger for the clock dividers (only 50% duty cycle dividers) of clock mux 1. Writing any value to this register provides a trigger to the dividers. This register should only be written after appropriately configuring the MC_CGM_MUX_1_DIV_TRIG_CTRL register.

Diagram



Fields

Field	Function
31-0 TRIGGER	Trigger for divider update

24.7.18 Clock Mux 1 Divider Update Status Register (MUX_1_DIV_UPD_STAT)

Offset

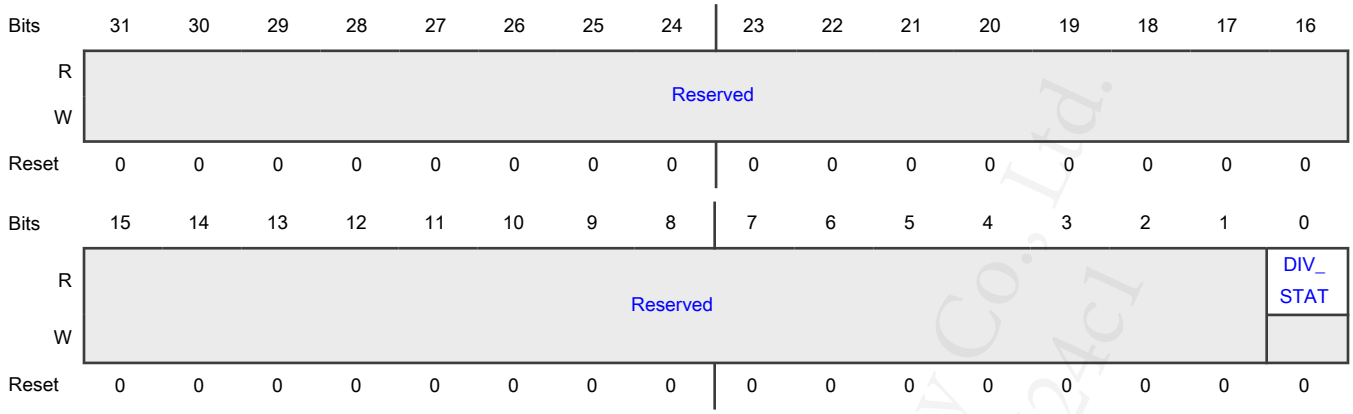
Register	Offset
MUX_1_DIV_UPD_STAT	37Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 1. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Provided under NDA only
COMPANY PROPRIETARY
Beijing Chip Automotive Technology Co., Ltd.
c8c462e5-0241-48a8-b251-390d3a8b624c1

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 1 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.7.19 Clock Mux 2 Select Control Register (MUX_2_CSC)

Offset

Register	Offset
MUX_2_CSC	380h

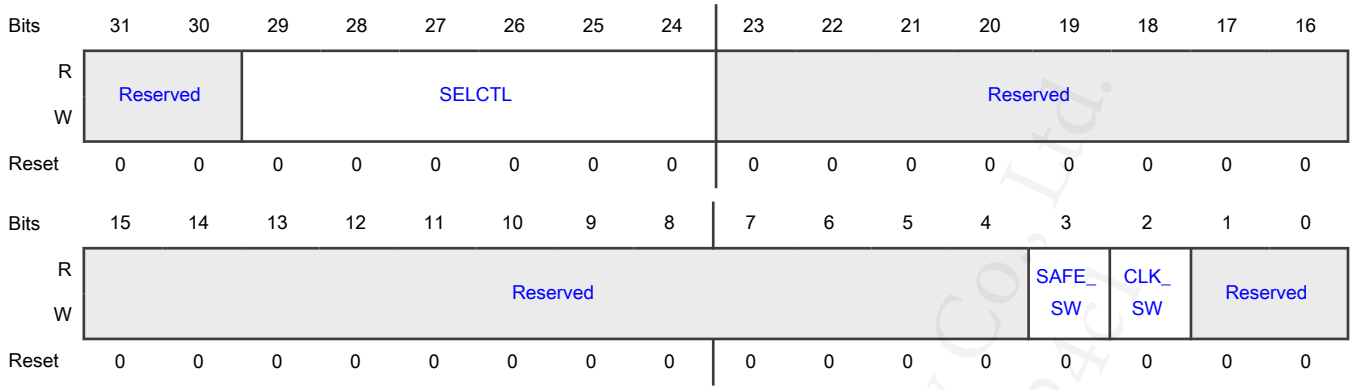
Function

This register provides the clock source selection control for clock mux 2. Clock mux 2 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e5-4821-48a8-b253-390d3ab624c1

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 2. The reserved values are not displayed. 000000b - clk_src_0 010111b - clk_src_23 110011b - clk_src_51 110101b - clk_src_53 111110b - clk_src_62
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 2. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46253-4821-48a8-b251-390d3ab624c

24.7.20 Clock Mux 2 Select Status Register (MUX_2_CSS)

Offset

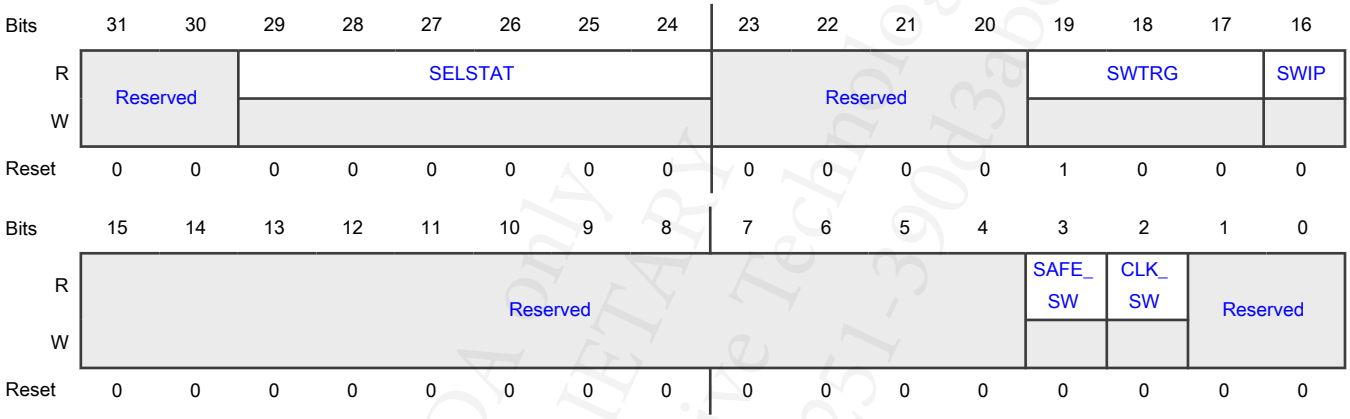
Register	Offset
MUX_2_CSS	384h

Function

This register provides the current clock source selection status for clock mux 2.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 2. The reserved values are not displayed. 000000b - clk_src_0 010111b - clk_src_23 110011b - clk_src_51 110101b - clk_src_53 111110b - clk_src_62
23-20 —	This field is reserved and reads return zeros.
19-17	Switch trigger cause

Table continues on the next page...

Table continued from the previous page...

Field	Function
SWTRG	This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 2. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 2. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

24.7.21 Clock Mux 2 Divider 0 Control Register (MUX_2_DC_0)

Offset

Register	Offset
MUX_2_DC_0	388h

Function

This register controls the clock divider 0 for clock mux 2.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DE	Reserved							DIV							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-8 —	This field is reserved and reads return zeros.
7-0 —	This field is reserved and reads return zeros.

24.7.22 Clock Mux 2 Divider 1 Control Register (MUX_2_DC_1)

Offset

Register	Offset
MUX_2_DC_1	38Ch

Function

This register controls the clock divider 1 for clock mux 2.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DE	Reserved													DIV	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4a21-48a8-b251-390d3ab624c1

Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-18 —	This field is reserved and reads return zeros.
17-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.7.23 Clock Mux 2 Divider Trigger Control Register (MUX_2_DIV_TRIG_CTRL)

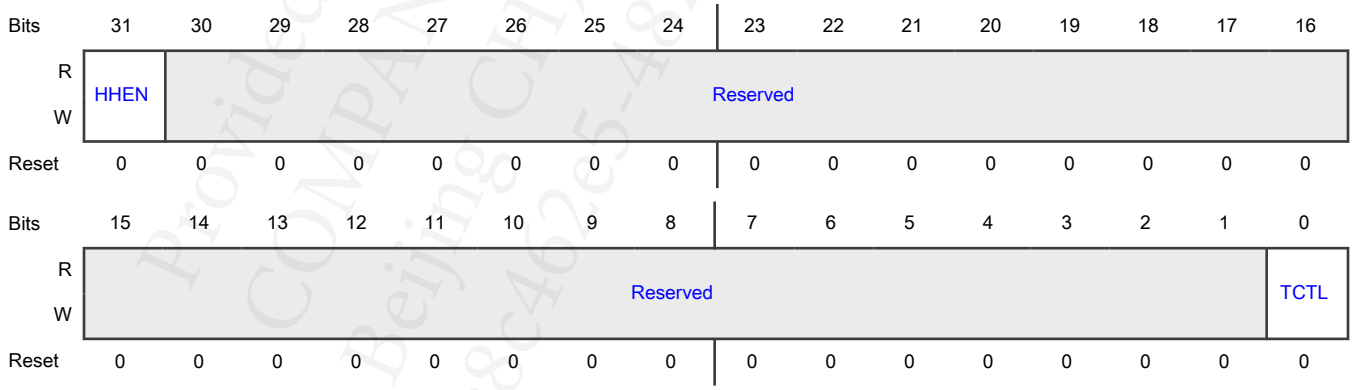
Offset

Register	Offset
MUX_2_DIV_TRIG_CTRL	3B4h

Function

This register selects whether the dividers associated with clock mux 2 are updated immediately on writing to the corresponding divider configuration register (referred to as immediate divider update) or only on writing to the MC_CGM_MUX_2_DIV_TRIG register (referred to as common trigger update). When common trigger update is configured, this register also controls initiation of the halt handshake protocol with the on-chip AXBS. Software is required to configure HHEN field for handshaking with on-chip AXBS when the ratio of division value among the clock dividers need to be changed.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a48-b251-390a3ab604c1

Fields

Field	Function
31 HHEN	Halt handshake enable This field controls the initiation of the halt handshake protocol with AXBS when a common trigger divider update is initiated. 0b - No halt handshake protocol is initiated. 1b - Halt handshake protocol is initiated.
30-1 —	This field is reserved and reads return zeros.
0 TCTL	Trigger control This field controls the divider update configuration between immediate and common update. 0b - Immediate divider update 1b - Common trigger divider update

24.7.24 Clock Mux 2 Divider Trigger Register (MUX_2_DIV_TRIG)

Offset

Register	Offset
MUX_2_DIV_TRIG	3B8h

Function

This register provides a common trigger for the clock dividers (only 50% duty cycle dividers) of clock mux 2. Writing any value to this register provides a trigger to the dividers. This register should only be written after appropriately configuring the MC_CGM_MUX_2_DIV_TRIG_CTRL register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	TRIGGER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	TRIGGER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c16255-4821-4828-b251-390d3ab624c1

Fields

Field	Function
31-0 TRIGGER	Trigger for divider update

24.7.25 Clock Mux 2 Divider Update Status Register (MUX_2_DIV_UPD_STAT)

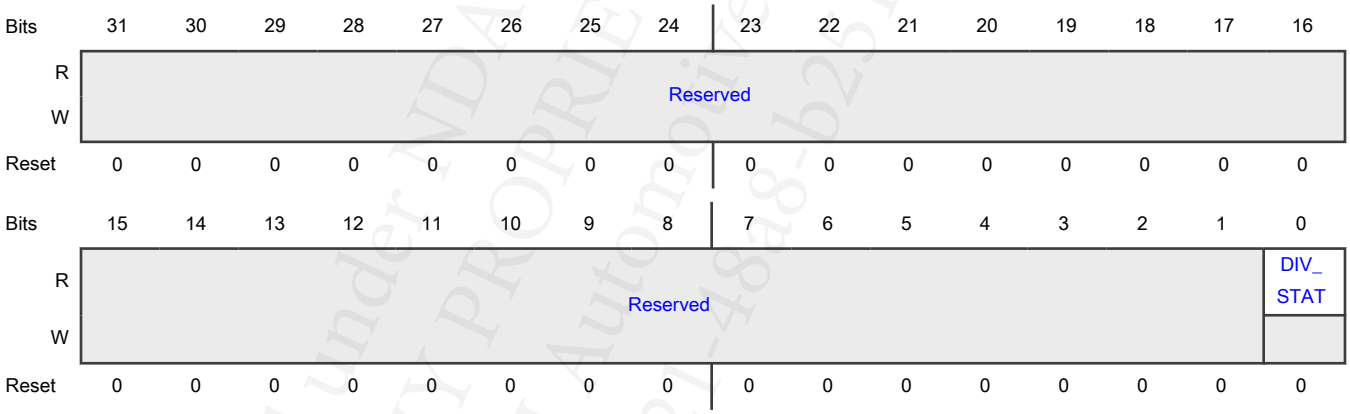
Offset

Register	Offset
MUX_2_DIV_UPD_STAT	3BCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 2. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 2 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.7.26 Clock Mux 3 Select Control Register (MUX_3_CSC)

Offset

Register	Offset
MUX_3_CSC	3C0h

Function

This register provides the clock source selection control for clock mux 3. Clock mux 3 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		SELCTL						Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												SAFE_SW	CLK_SW	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24	Clock source selection control

Table continues on the next page...

Table continued from the previous page...

Field	Function
SELCTL	Selects the source clock for clock mux 3. The reserved values are not displayed. 000000b - clk_src_0 010111b - clk_src_23 101110b - clk_src_46 110110b - clk_src_54 111000b - clk_src_56
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 3. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.7.27 Clock Mux 3 Select Status Register (MUX_3_CSS)

Offset

Register	Offset
MUX_3_CSS	3C4h

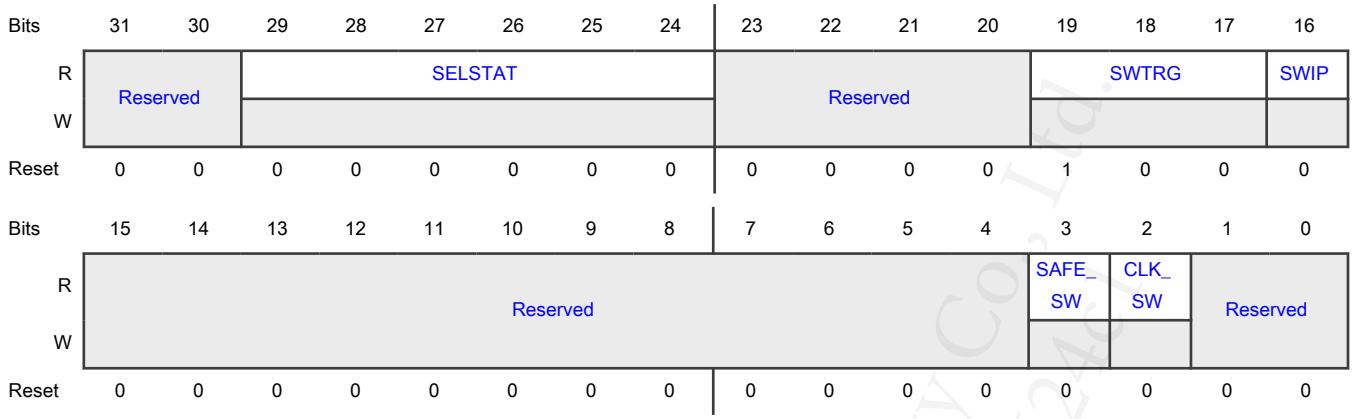
Function

This register provides the current clock source selection status for clock mux 3.

This register is reset on destructive reset only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-900ab624c1

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 3. The reserved values are not displayed. 000000b - clk_src_0 010111b - clk_src_23 101110b - clk_src_46 110110b - clk_src_54 111000b - clk_src_56
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 3. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 3. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.7.28 Clock Mux 3 Divider 0 Control Register (MUX_3_DC_0)

Offset

Register	Offset
MUX_3_DC_0	3C8h

Function

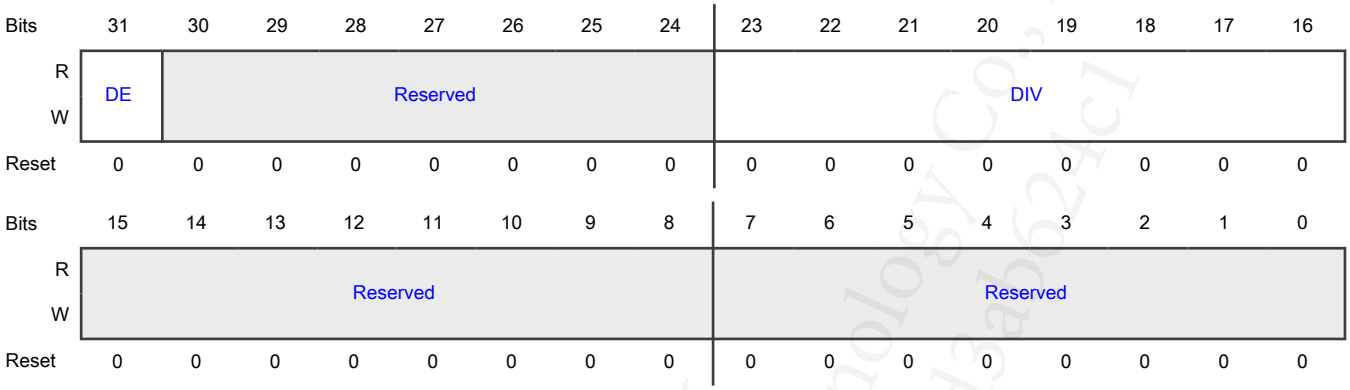
This register controls the clock divider 0 for clock mux 3.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-24 —	This field is reserved and reads return zeros.
23-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-8 —	This field is reserved and reads return zeros.
7-0 —	This field is reserved and reads return zeros.

24.7.29 Clock Mux 3 Divider 1 Control Register (MUX_3_DC_1)

Offset

Register	Offset
MUX_3_DC_1	3CCh

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co., Ltd.
c8c46255-4821-8a8-b251-390d3ab624c1

Function

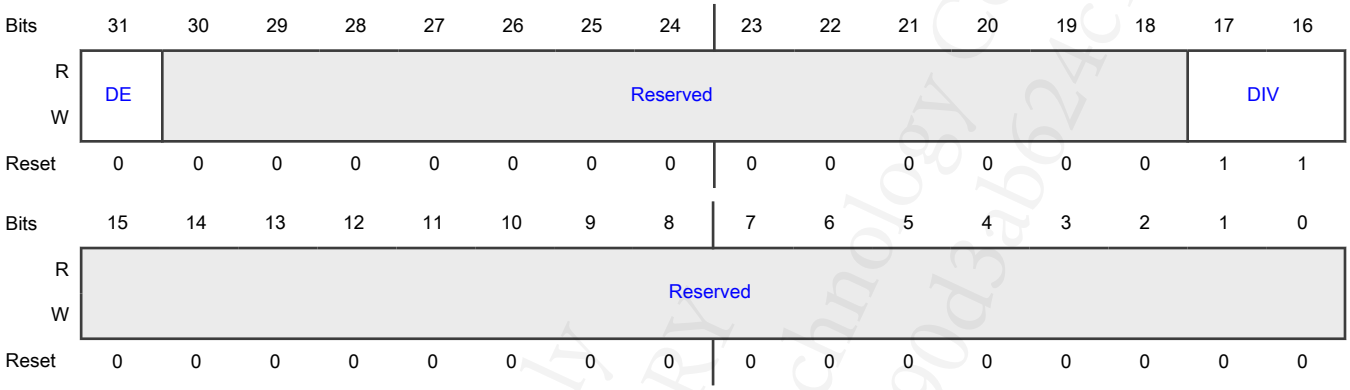
This register controls the clock divider 1 for clock mux 3.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-18 —	This field is reserved and reads return zeros.
17-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.7.30 Clock Mux 3 Divider Trigger Control Register (MUX_3_DIV_TRIG_CTRL)

Offset

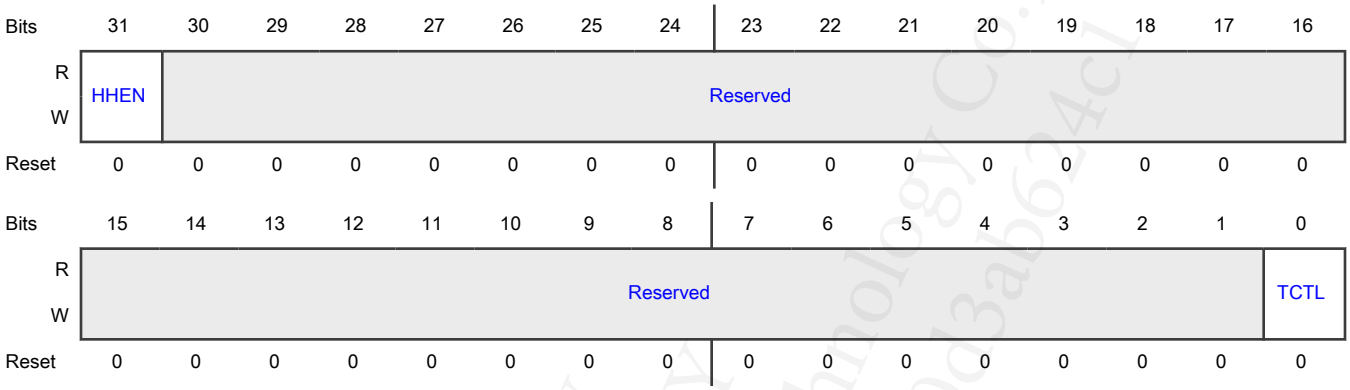
Register	Offset
MUX_3_DIV_TRIG_CTRL	3F4h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c46255-462a-48a8-b251-390d3ab624c1

Function

This register selects whether the dividers associated with clock mux 3 are updated immediately on writing to the corresponding divider configuration register (referred to as immediate divider update) or only on writing to the MC_CGM_MUX_3_DIV_TRIG register (referred to as common trigger update). When common trigger update is configured, this register also controls initiation of the halt handshake protocol with the on-chip AXBS. Software is required to configure HHEN field for handshaking with on-chip AXBS when the ratio of division value among the clock dividers need to be changed.

Diagram



Fields

Field	Function
31 HHEN	Halt handshake enable This field controls the initiation of the halt handshake protocol with AXBS when a common trigger divider update is initiated. 0b - No halt handshake protocol is initiated. 1b - Halt handshake protocol is initiated.
30-1 —	This field is reserved and reads return zeros.
0 TCTL	Trigger control This field controls the divider update configuration between immediate and common update. 0b - Immediate divider update 1b - Common trigger divider update

24.7.31 Clock Mux 3 Divider Trigger Register (MUX_3_DIV_TRIG)

Offset

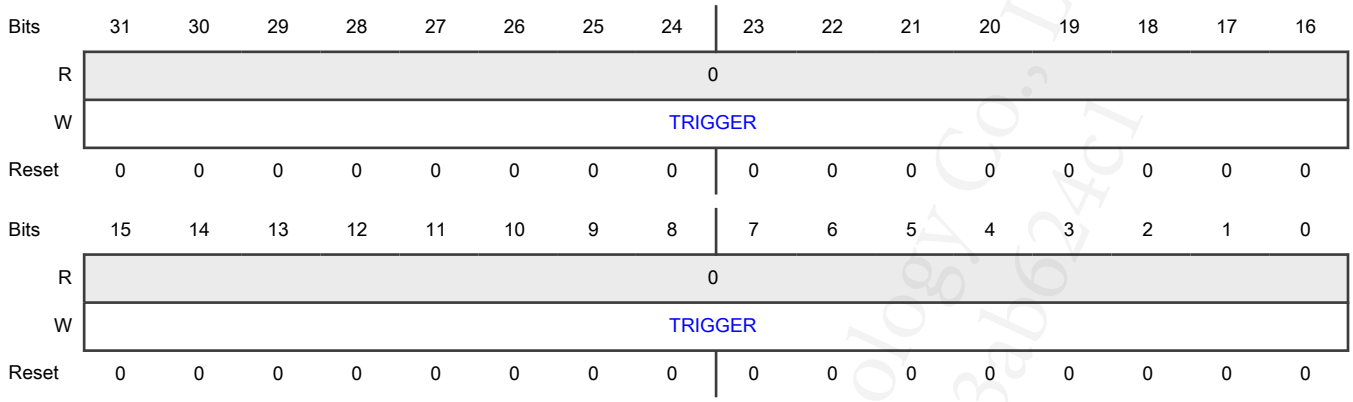
Register	Offset
MUX_3_DIV_TRIG	3F8h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46255-8181-48a8-b251-3903ab614c1

Function

This register provides a common trigger for the clock dividers (only 50% duty cycle dividers) of clock mux 3. Writing any value to this register provides a trigger to the dividers. This register should only be written after appropriately configuring the MC_CGM_MUX_3_DIV_TRIG_CTRL register.

Diagram



Fields

Field	Function
31-0 TRIGGER	Trigger for divider update

24.7.32 Clock Mux 3 Divider Update Status Register (MUX_3_DIV_UPD_STAT)

Offset

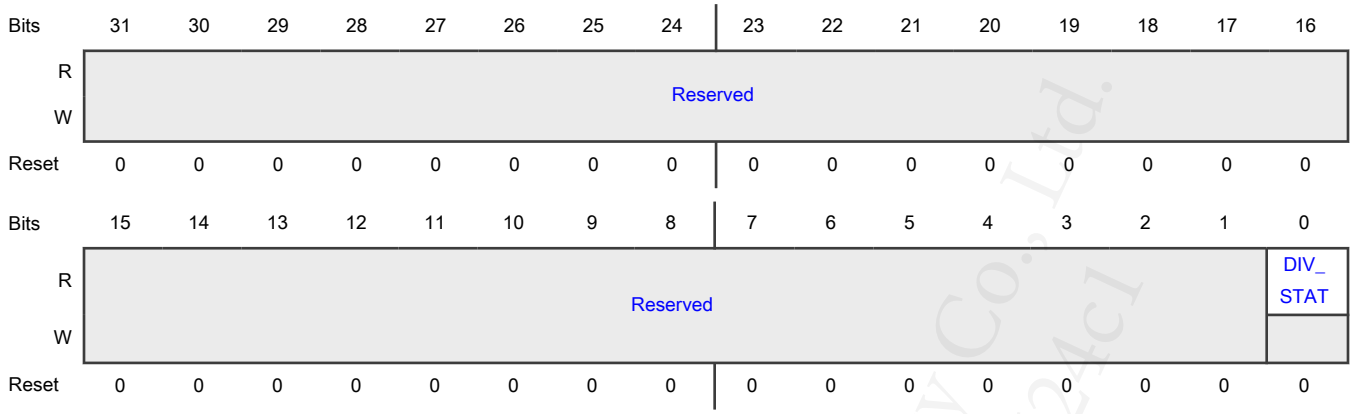
Register	Offset
MUX_3_DIV_UPD_STAT	3FCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 3. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Provided under NDA only
COMPANY PROPRIETARY
Beijing Chipsea Automotive Technology Co., Ltd.
c8c462e5-0241-48a8-b251-390d3a8b624c1

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 3 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source: 1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.7.33 Clock Mux 4 Select Control Register (MUX_4_CSC)

Offset

Register	Offset
MUX_4_CSC	400h

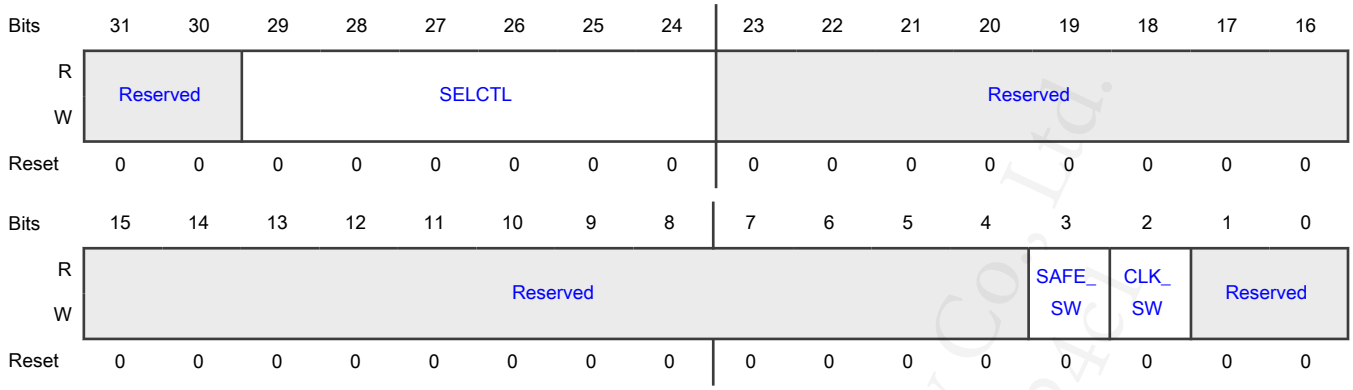
Function

This register provides the clock source selection control for clock mux 4. Clock mux 4 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e5-4821-48a8-b253-390d3ab624c1

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 4. The reserved values are not displayed. 000000b - clk_src_0 110001b - clk_src_49 111010b - clk_src_58 111011b - clk_src_59
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 4. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEIL Automotive Technology Co., Ltd.
c8c462e5-8214-48a8-b251-390d3ab624c

24.7.34 Clock Mux 4 Select Status Register (MUX_4_CSS)

Offset

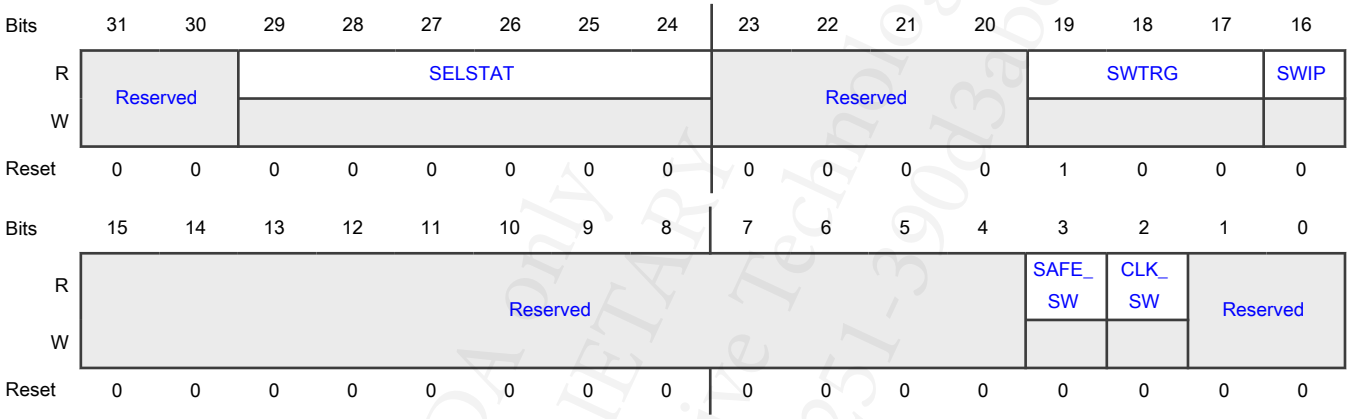
Register	Offset
MUX_4_CSS	404h

Function

This register provides the current clock source selection status for clock mux 4.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 4. The reserved values are not displayed. 000000b - clk_src_0 110001b - clk_src_49 111010b - clk_src_58 111011b - clk_src_59
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c460e5-4821-48a8-b251-390d3a624c1

Table continued from the previous page...

Field	Function
	This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 4. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 4. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

24.7.35 Clock Mux 5 Select Control Register (MUX_5_CSC)

Offset

Register	Offset
MUX_5_CSC	440h

Function

This register provides the clock source selection control for clock mux 5. Clock mux 5 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		SELCTL						Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												SAFE_SW	CLK_SW	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 5. The reserved values are not displayed. 000000b - clk_src_0 110100b - clk_src_52

Table continues on the next page...

Table continued from the previous page...

Field	Function
	111100b - clk_src_60 111111b - clk_src_63
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 5. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.7.36 Clock Mux 5 Select Status Register (MUX_5_CSS)

Offset

Register	Offset
MUX_5_CSS	444h

Function

This register provides the current clock source selection status for clock mux 5.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		SELSTAT						Reserved				SWTRG			SWIP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												SAFE_SW	CLK_SW	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CTJ Automotive Technology Co., Ltd.
c8c46e65-4821-48a8-b251-390a0a6b64c1

Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 5. The reserved values are not displayed. 000000b - clk_src_0 110100b - clk_src_52 111100b - clk_src_60 111111b - clk_src_63
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 5. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 5. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.7.37 Clock Mux 6 Select Control Register (MUX_6_CSC)

Offset

Register	Offset
MUX_6_CSC	480h

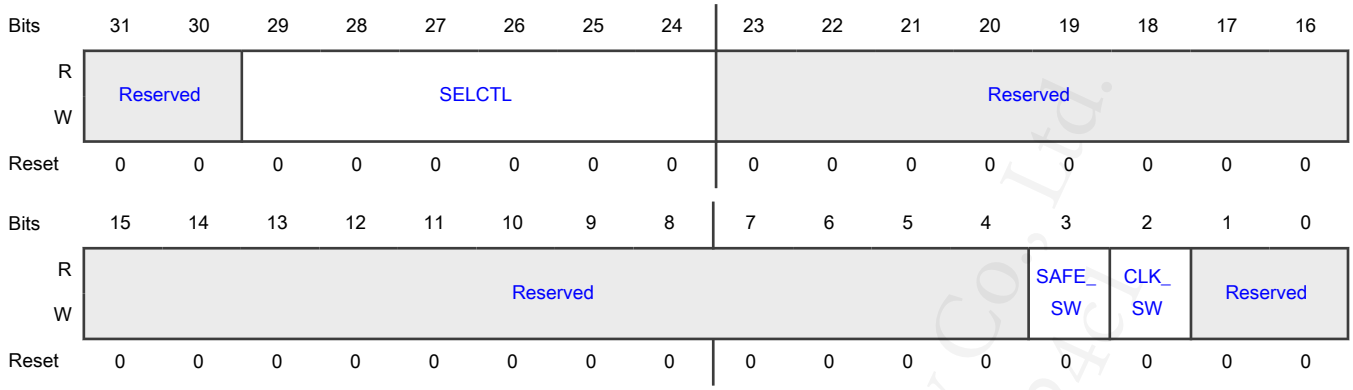
Function

This register provides the clock source selection control for clock mux 6. Clock mux 6 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 6. The reserved values are not displayed. 000000b - clk_src_0 101111b - clk_src_47 110111b - clk_src_55 111101b - clk_src_61
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 6. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEIU Automotive Technology Co., Ltd.
c8c462e5-8214-48a8-b251-390d3ab624c

24.7.38 Clock Mux 6 Select Status Register (MUX_6_CSS)

Offset

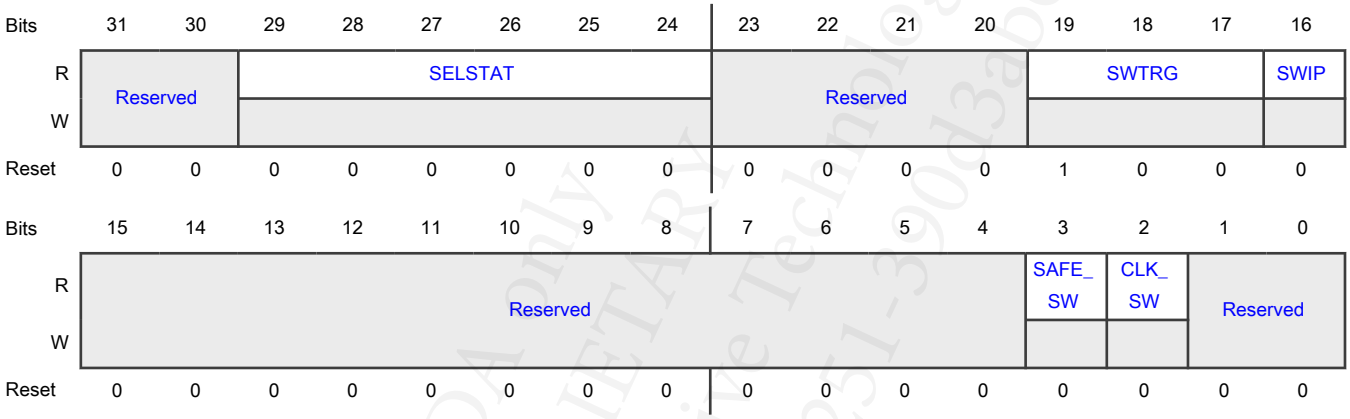
Register	Offset
MUX_6_CSS	484h

Function

This register provides the current clock source selection status for clock mux 6.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 6. The reserved values are not displayed. 000000b - clk_src_0 101111b - clk_src_47 110111b - clk_src_55 111101b - clk_src_61
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 6. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 6. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0	This field is reserved and reads return zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

24.7.39 Clock Mux 7 Select Control Register (MUX_7_CSC)

Offset

Register	Offset
MUX_7_CSC	4C0h

Function

This register provides the clock source selection control for clock mux 7. Clock mux 7 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		SELCTL						Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												SAFE_SW	CLK_SW	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 7. The reserved values are not displayed. 000000b - clk_src_0 110010b - clk_src_50

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 7. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.7.40 Clock Mux 7 Select Status Register (MUX_7_CSS)

Offset

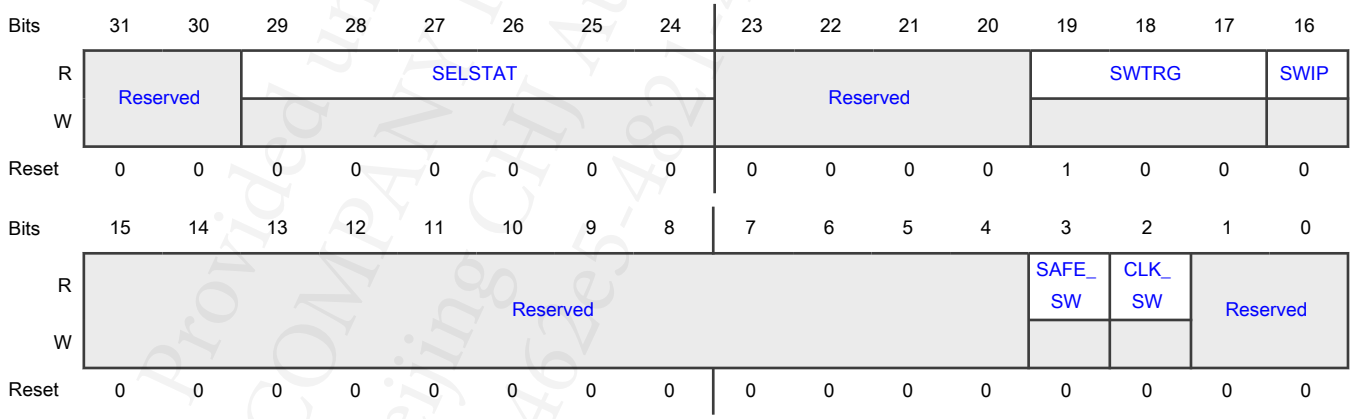
Register	Offset
MUX_7_CSS	4C4h

Function

This register provides the current clock source selection status for clock mux 7.

This register is reset on destructive reset only.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48ad-b251-390d3ab6240f

Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 7. The reserved values are not displayed. 000000b - clk_src_0 110010b - clk_src_50
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
SAFE_SW	This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 7. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 7. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.7.41 Clock Mux 7 Divider 0 Control Register (MUX_7_DC_0)

Offset

Register	Offset
MUX_7_DC_0	4C8h

Function

This register controls the clock divider 0 for clock mux 7.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DE	Reserved									DIV					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.7.42 Clock Mux 7 Divider Trigger Control Register (MUX_7_DIV_TRIG_CTRL)

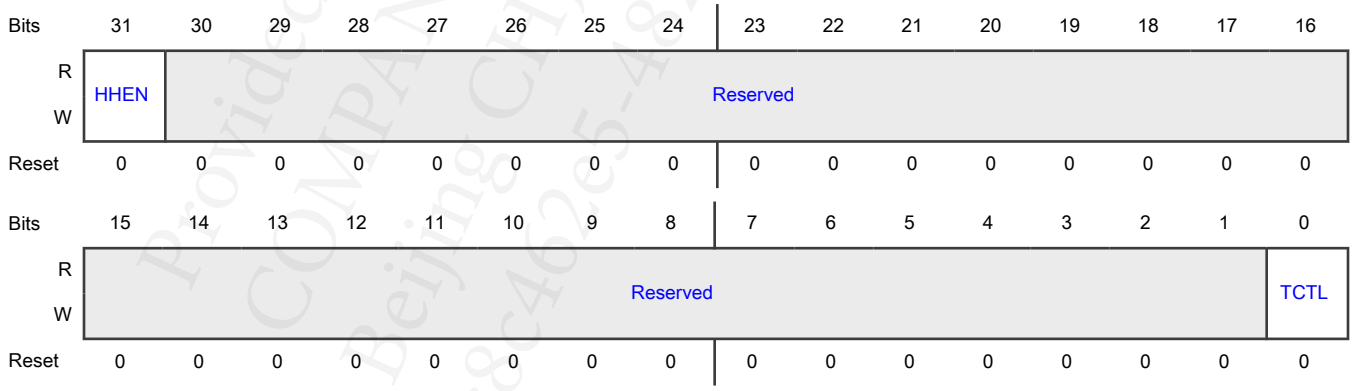
Offset

Register	Offset
MUX_7_DIV_TRIG_CTRL	4F4h

Function

This register selects whether the dividers associated with clock mux 7 are updated immediately on writing to the corresponding divider configuration register (referred to as immediate divider update) or only on writing to the MC_CGM_MUX_7_DIV_TRIG register (referred to as common trigger update). When common trigger update is configured, this register also controls initiation of the halt handshake protocol with the on-chip AXBS. Software is required to configure HHEN field for handshaking with on-chip AXBS when the ratio of division value among the clock dividers need to be changed.

Diagram



Fields

Field	Function
31 HHEN	Halt handshake enable This field controls the initiation of the halt handshake protocol with AXBS when a common trigger divider update is initiated. 0b - No halt handshake protocol is initiated. 1b - Halt handshake protocol is initiated.
30-1 —	This field is reserved and reads return zeros.
0 TCTL	Trigger control This field controls the divider update configuration between immediate and common update. 0b - Immediate divider update 1b - Common trigger divider update

24.7.43 Clock Mux 7 Divider Trigger Register (MUX_7_DIV_TRIG)

Offset

Register	Offset
MUX_7_DIV_TRIG	4F8h

Function

This register provides a common trigger for the clock dividers (only 50% duty cycle dividers) of clock mux 7. Writing any value to this register provides a trigger to the dividers. This register should only be written after appropriately configuring the MC_CGM_MUX_7_DIV_TRIG_CTRL register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	TRIGGER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	TRIGGER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c16255-4821-4828-b251-390d3ab624c1

Fields

Field	Function
31-0 TRIGGER	Trigger for divider update

24.7.44 Clock Mux 7 Divider Update Status Register (MUX_7_DIV_UPD_STAT)

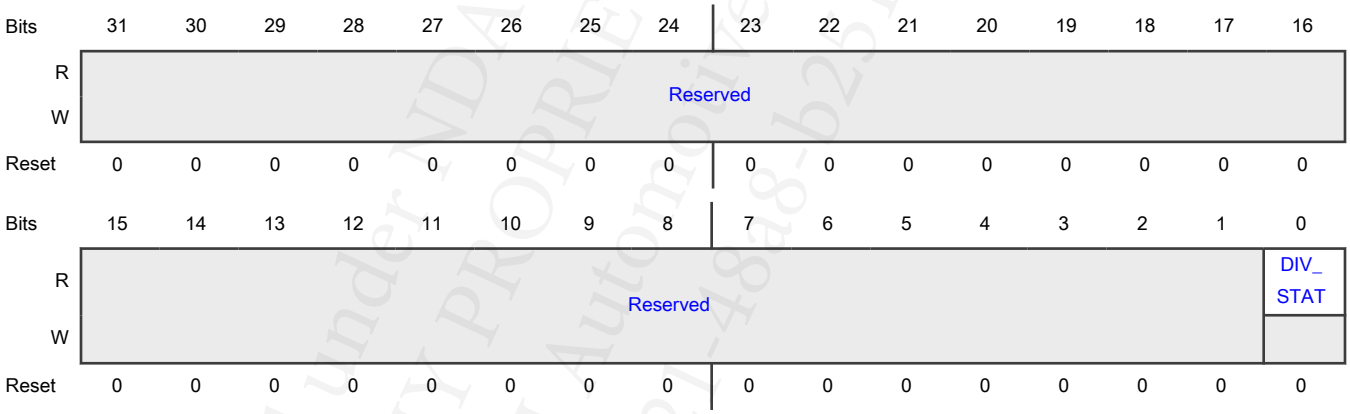
Offset

Register	Offset
MUX_7_DIV_UPD_STAT	4FCh

Function

This register provides the update status of the clock dividers corresponding to clock mux 7. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 7 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.7.45 Clock Mux 8 Select Control Register (MUX_8_CSC)

Offset

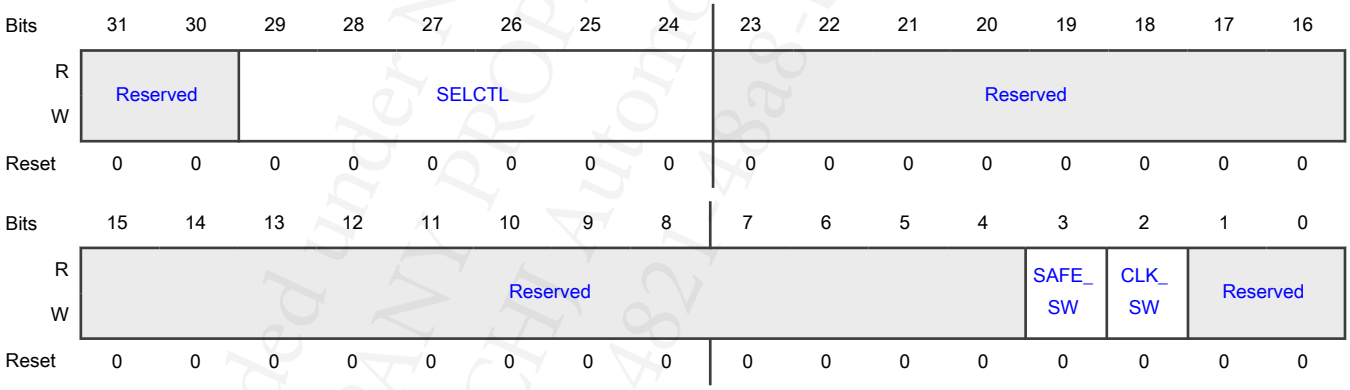
Register	Offset
MUX_8_CSC	500h

Function

This register provides the clock source selection control for clock mux 8. Clock mux 8 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24	Clock source selection control

Table continues on the next page...

Table continued from the previous page...

Field	Function
SELCTL	Selects the source clock for clock mux 8. The reserved values are not displayed. 000000b - clk_src_0 110101b - clk_src_53
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 8. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.7.46 Clock Mux 8 Select Status Register (MUX_8_CSS)

Offset

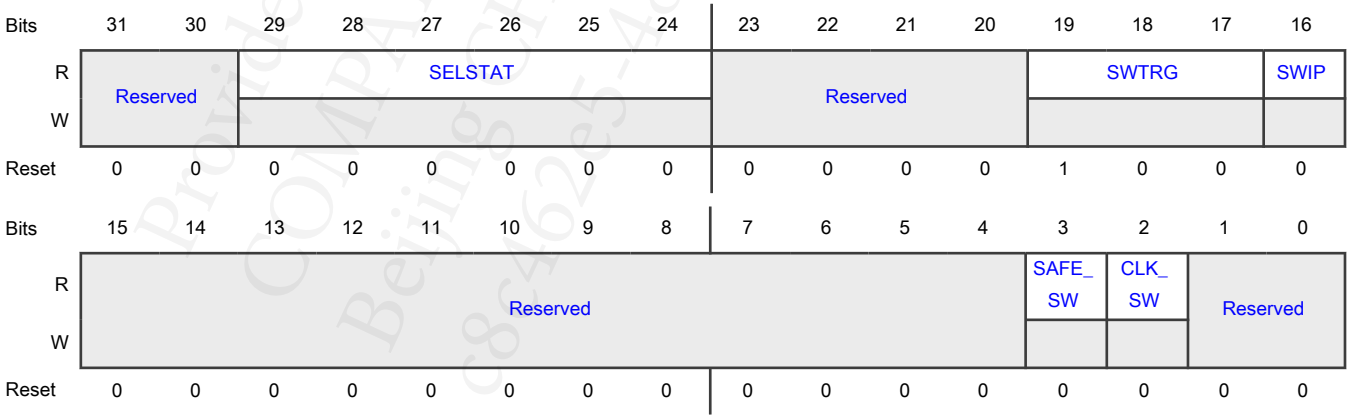
Register	Offset
MUX_8_CSS	504h

Function

This register provides the current clock source selection status for clock mux 8.

This register is reset on destructive reset only.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing ESWIN Automotive Technology Co., Ltd.
c8462e25-4821-48a8-b251-3903cab24c1

Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 8. The reserved values are not displayed. 000000b - clk_src_0 110101b - clk_src_53
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
SAFE_SW	This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 8. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 8. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.7.47 Clock Mux 8 Divider 0 Control Register (MUX_8_DC_0)

Offset

Register	Offset
MUX_8_DC_0	508h

Function

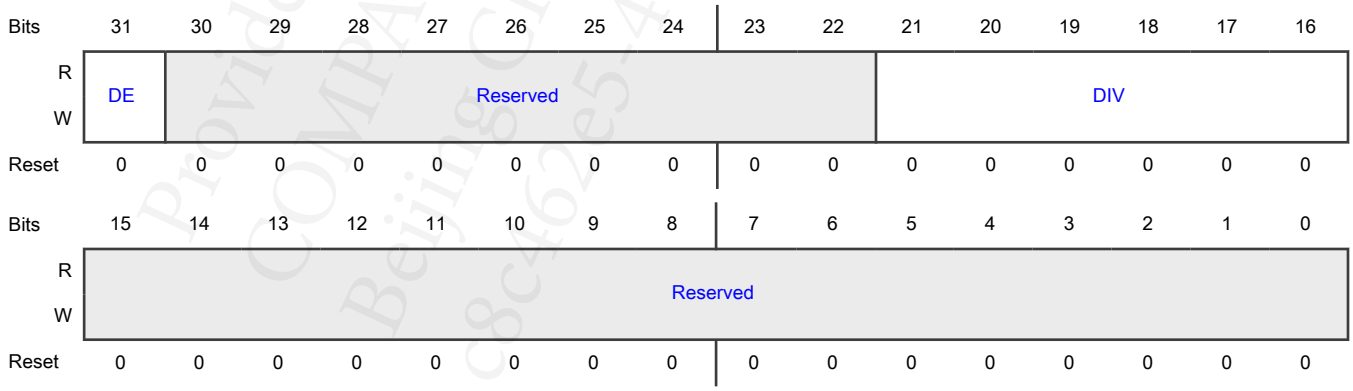
This register controls the clock divider 0 for clock mux 8.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram



Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.7.48 Clock Mux 8 Divider Trigger Control Register (MUX_8_DIV_TRIG_CTRL)

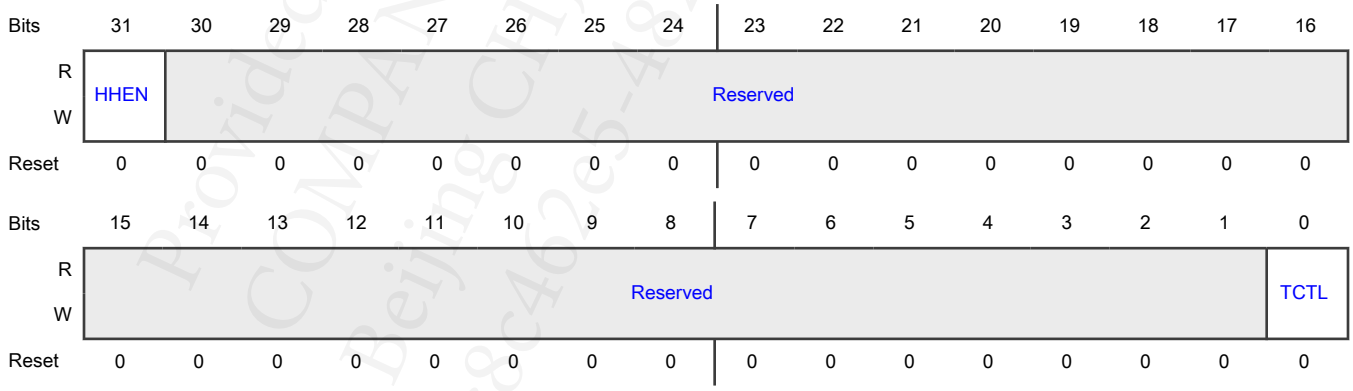
Offset

Register	Offset
MUX_8_DIV_TRIG_CTRL	534h

Function

This register selects whether the dividers associated with clock mux 8 are updated immediately on writing to the corresponding divider configuration register (referred to as immediate divider update) or only on writing to the MC_CGM_MUX_8_DIV_TRIG register (referred to as common trigger update). When common trigger update is configured, this register also controls initiation of the halt handshake protocol with the on-chip AXBS. Software is required to configure HHEN field for handshaking with on-chip AXBS when the ratio of division value among the clock dividers need to be changed.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-4248-b251-390a3ab604c1

Fields

Field	Function
31 HHEN	Halt handshake enable This field controls the initiation of the halt handshake protocol with AXBS when a common trigger divider update is initiated. 0b - No halt handshake protocol is initiated. 1b - Halt handshake protocol is initiated.
30-1 —	This field is reserved and reads return zeros.
0 TCTL	Trigger control This field controls the divider update configuration between immediate and common update. 0b - Immediate divider update 1b - Common trigger divider update

24.7.49 Clock Mux 8 Divider Trigger Register (MUX_8_DIV_TRIG)

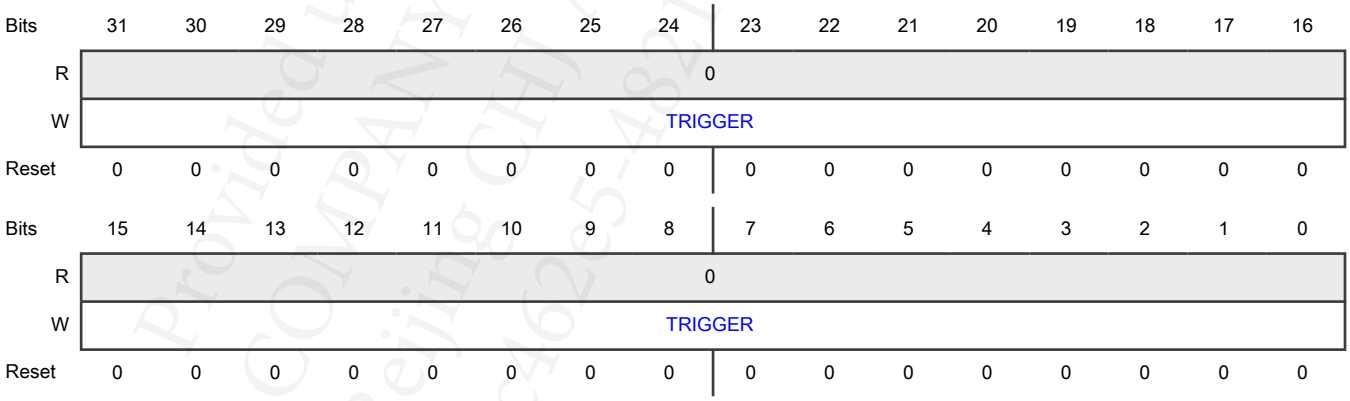
Offset

Register	Offset
MUX_8_DIV_TRIG	538h

Function

This register provides a common trigger for the clock dividers (only 50% duty cycle dividers) of clock mux 8. Writing any value to this register provides a trigger to the dividers. This register should only be written after appropriately configuring the MC_CGM_MUX_8_DIV_TRIG_CTRL register.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c16255-4821-4828-b251-390d3ab624c1

Fields

Field	Function
31-0 TRIGGER	Trigger for divider update

24.7.50 Clock Mux 8 Divider Update Status Register (MUX_8_DIV_UPD_STAT)

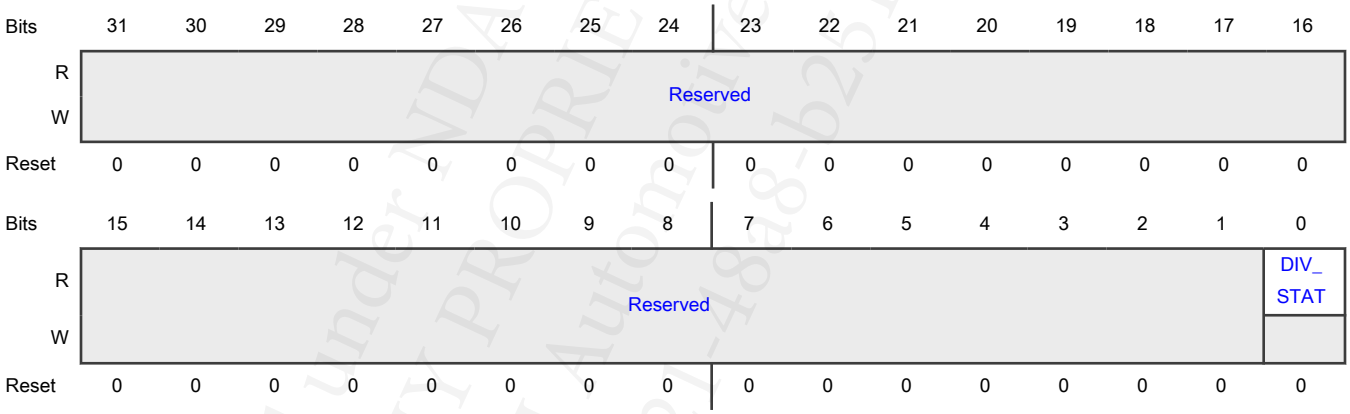
Offset

Register	Offset
MUX_8_DIV_UPD_STAT	53Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 8. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 8 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.7.51 Clock Mux 9 Select Control Register (MUX_9_CSC)

Offset

Register	Offset
MUX_9_CSC	540h

Function

This register provides the clock source selection control for clock mux 9. Clock mux 9 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		SELCTL						Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												SAFE_SW	CLK_SW	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24	Clock source selection control

Table continues on the next page...

Table continued from the previous page...

Field	Function
SELCTL	Selects the source clock for clock mux 9. The reserved values are not displayed. 000000b - clk_src_0 111000b - clk_src_56
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 9. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.7.52 Clock Mux 9 Select Status Register (MUX_9_CSS)

Offset

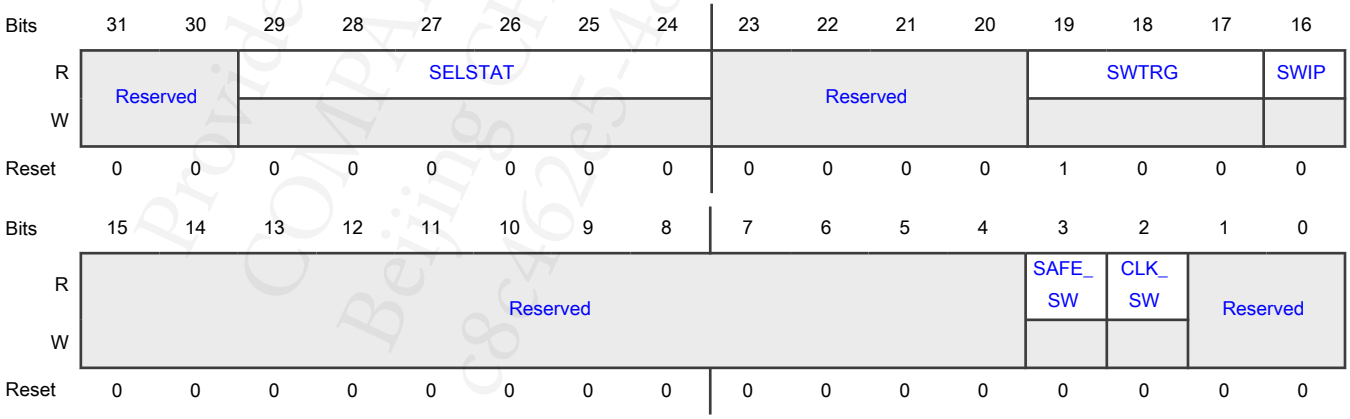
Register	Offset
MUX_9_CSS	544h

Function

This register provides the current clock source selection status for clock mux 9.

This register is reset on destructive reset only.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing ESWIN Automotive Technology Co., Ltd.
c8462e25-4821-48a8-b251-3902ab24c1

Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 9. The reserved values are not displayed. 000000b - clk_src_0 111000b - clk_src_56
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0. 100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3	Safe clock request

Table continues on the next page...

Table continued from the previous page...

Field	Function
SAFE_SW	This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 9. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 9. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.7.53 Clock Mux 9 Divider 0 Control Register (MUX_9_DC_0)

Offset

Register	Offset
MUX_9_DC_0	548h

Function

This register controls the clock divider 0 for clock mux 9.

This divider is a 50% duty cycle divider.

NOTE

The update to the fields of this register should be an atomic write, that is, one single write should update the complete register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DE	Reserved									DIV					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e54-8214-48a8-b251-390d3ab624c1

Fields

Field	Function
31 DE	Divider enable 0b - Divider is disabled. 1b - Divider is enabled.
30-22 —	This field is reserved and reads return zeros.
21-16 DIV	Division value This field provides the division value for the clock divider. The clock period of the clock after division is 'DIV+1' times the time period of the current input clock to the divider.
15-0 —	This field is reserved and reads return zeros.

24.7.54 Clock Mux 9 Divider Trigger Control Register (MUX_9_DIV_TRIG_CTRL)

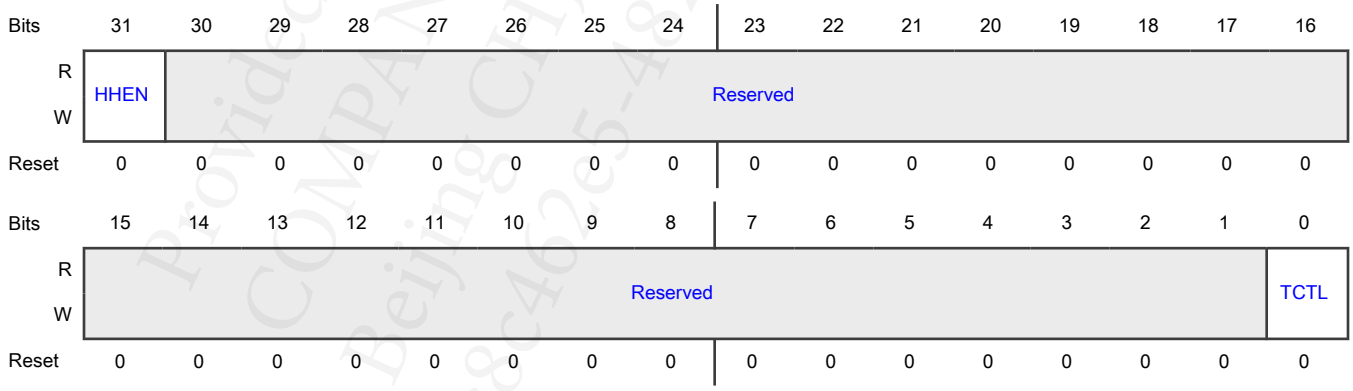
Offset

Register	Offset
MUX_9_DIV_TRIG_CTRL	574h

Function

This register selects whether the dividers associated with clock mux 9 are updated immediately on writing to the corresponding divider configuration register (referred to as immediate divider update) or only on writing to the MC_CGM_MUX_9_DIV_TRIG register (referred to as common trigger update). When common trigger update is configured, this register also controls initiation of the halt handshake protocol with the on-chip AXBS. Software is required to configure HHEN field for handshaking with on-chip AXBS when the ratio of division value among the clock dividers need to be changed.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-4e48-b251-390a3ab604c1

Fields

Field	Function
31 HHEN	Halt handshake enable This field controls the initiation of the halt handshake protocol with AXBS when a common trigger divider update is initiated. 0b - No halt handshake protocol is initiated. 1b - Halt handshake protocol is initiated.
30-1 —	This field is reserved and reads return zeros.
0 TCTL	Trigger control This field controls the divider update configuration between immediate and common update. 0b - Immediate divider update 1b - Common trigger divider update

24.7.55 Clock Mux 9 Divider Trigger Register (MUX_9_DIV_TRIG)

Offset

Register	Offset
MUX_9_DIV_TRIG	578h

Function

This register provides a common trigger for the clock dividers (only 50% duty cycle dividers) of clock mux 9. Writing any value to this register provides a trigger to the dividers. This register should only be written after appropriately configuring the MC_CGM_MUX_9_DIV_TRIG_CTRL register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	TRIGGER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	TRIGGER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c16255-4821-4828-b251-390d3ab624c1

Fields

Field	Function
31-0 TRIGGER	Trigger for divider update

24.7.56 Clock Mux 9 Divider Update Status Register (MUX_9_DIV_UPD_STAT)

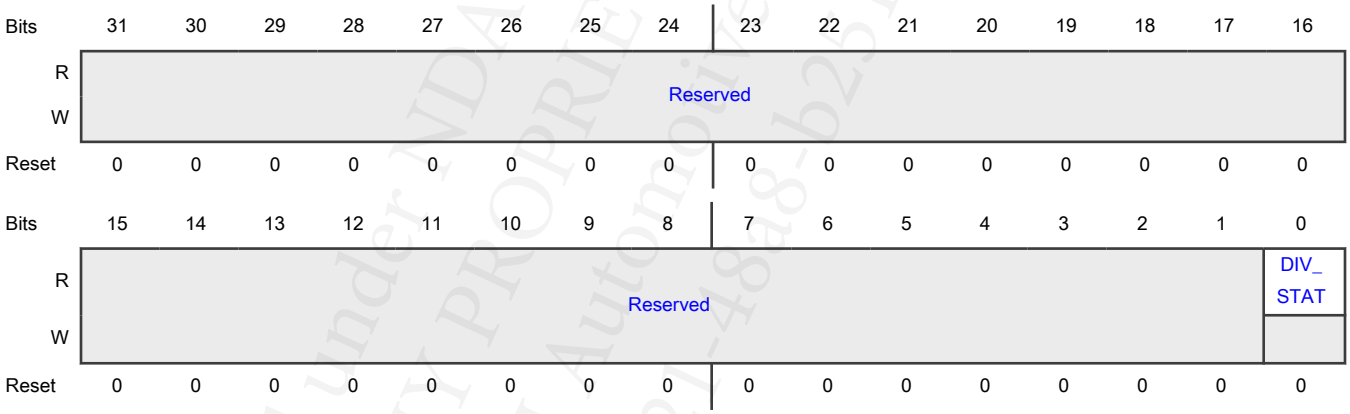
Offset

Register	Offset
MUX_9_DIV_UPD_STAT	57Ch

Function

This register provides the update status of the clock dividers corresponding to clock mux 9. When a write operation on any divider control register is performed, the divider status bit in this register is set to logic-1. The bit is set to logic-0 when the divider has sampled the new divider configuration. Performing multiple writes without tracking the status bit on same or other clock dividers inside the same clock mux leads to inconsistent reporting, that is, the divider status maybe be set to logic-0 when the corresponding divider update is pending.

Diagram



Fields

Field	Function
31-1 —	This field is reserved and reads return zeros.
0 DIV_STAT	Divider status for clock mux 9 On reading MUX_n_DIV_UPD_STAT after updating a divider control register, if the value of this field is fixed to 1 because of an error in the selected clock source, perform the following steps to switch the mux to a new clock source:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1. Switch the mux to a working clock source without polling this field. 2. Update MUX_n_DC_m and poll this field. 0b - No divider configuration update is pending. 1b - Divider configuration update on at least one divider associated with this multiplexer is pending.

24.8 MC_CGM_5 register descriptions

MC_CGM implements a set of clock multiplexers that share PCFS configuration registers. MC_CGM registers have the following properties:

- All registers are 32-bit wide.
- Only 32-bit read and write accesses are supported.
- Read/write accesses of less than 32 bits do not terminate with an error, and hardware behavior is unpredictable.
- Writes to read-only register fields in writable registers are ignored and do not provide an error response.
- Writes to read-only registers are aborted with an error response.

24.8.1 MC_CGM_5 memory map

MC_CGM_5 base address: 4006_8000h

Offset	Register	Width (In bits)	Access	Reset value
300h	Clock Mux 0 Select Control Register (MUX_0_CSC)	32	RW	0000_0000h
304h	Clock Mux 0 Select Status Register (MUX_0_CSS)	32	RO	0008_0000h

24.8.2 Clock Mux 0 Select Control Register (MUX_0_CSC)

Offset

Register	Offset
MUX_0_CSC	300h

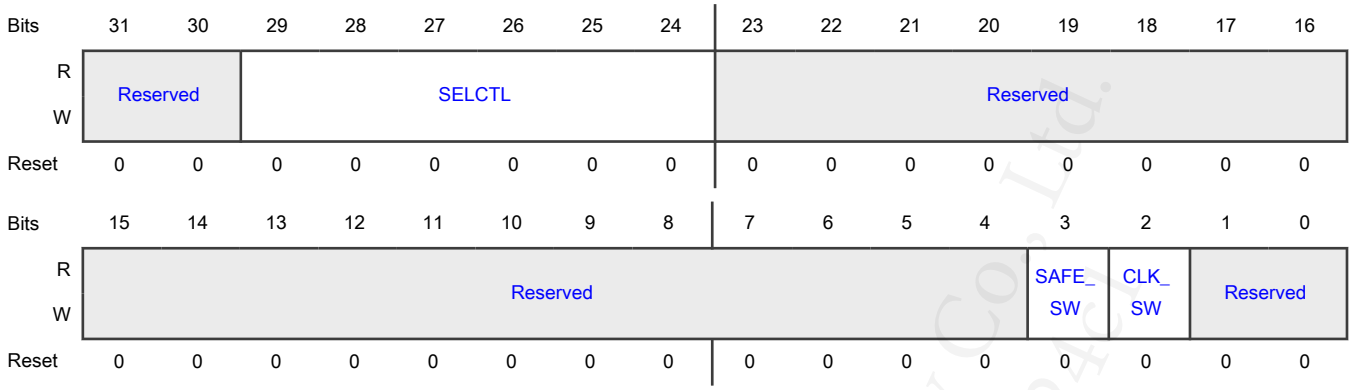
Function

This register provides the clock source selection control for clock mux 0. Clock mux 0 implements hardware control clock switching ensuring that the clock switch happens in a graceful manner (without glitches). See the "Hardware-controlled clock multiplexer" section for details.

This register is reset on destructive reset only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e5-4821-48a8-b251-390d3ab624d1

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELCTL	Clock source selection control Selects the source clock for clock mux 0. The reserved values are not displayed. 000000b - clk_src_0 100100b - clk_src_36
23-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request Writing 1 to this bit makes a safe clock switch request to clk_src_0. After a safe clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
2 CLK_SW	Clock switch Writing 1 to this bit makes a clock switch request to clock mux 0. After a clock switch operation is requested, this bit is auto cleared and a corresponding bit in the status register is set.
1-0 —	This field is reserved and reads return zeros.

24.8.3 Clock Mux 0 Select Status Register (MUX_0_CSS)

Offset

Register	Offset
MUX_0_CSS	304h

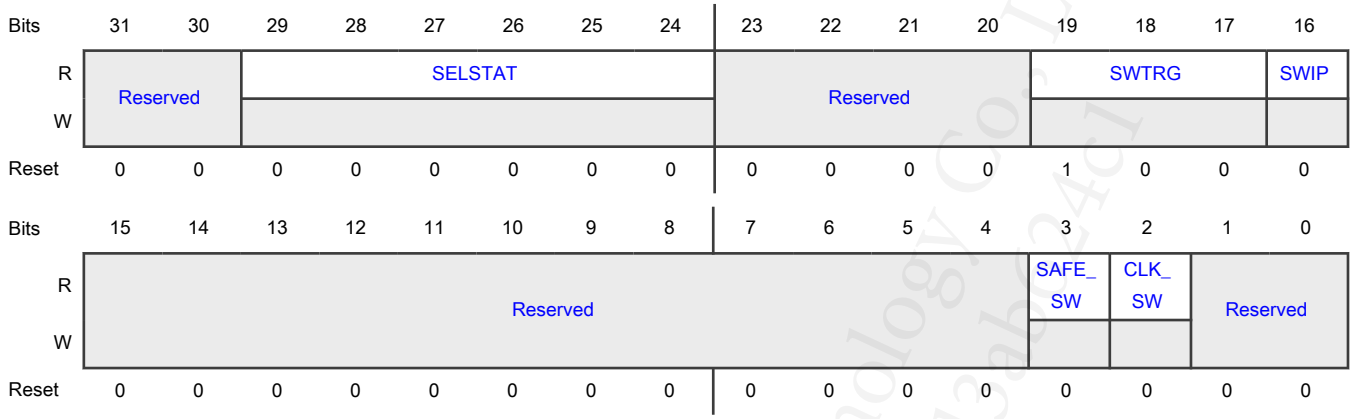
Preliminary NDA only
COMPANY PROPRIETARY
Beijing C10 Automotive Technology Co., Ltd.
c8c4625_492_4a8-b251-390d3ab624

Function

This register provides the current clock source selection status for clock mux 0.

This register is reset on destructive reset only.

Diagram



Fields

Field	Function
31-30 —	This field is reserved and reads return zeros.
29-24 SELSTAT	Clock source selection status This value indicates the current source selected for clock mux 0. The reserved values are not displayed. 000000b - clk_src_0 100100b - clk_src_36
23-20 —	This field is reserved and reads return zeros.
19-17 SWTRG	Switch trigger cause This value indicates the cause for the latest clock source switch. **NOTE** If the clock fails, followed by multiple safe clock switch requests for MC_CGM hardware clock mux, the value of the SWTRG field can be either 4 or 5. 000b - Reserved 001b - Switch after request succeeded. 010b - Switch after the request failed because of an inactive target clock and the current clock is clk_src_0. 011b - Switch after the request failed because of an inactive current clock and the current clock is clk_src_0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - Switch to clk_src_0 because of a safe clock request or reset succeeded. 101b - Switch to clk_src_0 because of a safe clock request or reset succeeded, but the previous current clock source was inactive. 110b - Reserved 111b - Reserved
16 SWIP	Switch in progress **NOTE** New clock switch request can only be given three clock cycles after the completion of the previous request. 0b - Clock source switching is complete. 1b - Clock source switching is in progress.
15-4 —	This field is reserved and reads return zeros.
3 SAFE_SW	Safe clock request This field provides an indication of whether a switch to safe clock operation was requested during the previous/ongoing request on clock mux 0. 0b - No safe clock switch operation was requested. 1b - Safe clock switch operation was requested.
2 CLK_SW	Clock switch This field provides an indication of whether a clock switch operation was requested during the previous/ ongoing request on clock mux 0. 0b - No clock switch operation was requested. 1b - Clock switch operation was requested.
1-0 —	This field is reserved and reads return zeros.

24.9 Functional description

24.9.1 Clock selection multiplexer

Each of the clock multiplexers inside the MC_CGM either implements a fully hardware-controlled clock multiplexer or a software-controlled clock multiplexer. The following sections describe the two variants of the clock multiplexer.

24.9.1.1 Hardware-controlled clock multiplexer

In hardware-based clock multiplexing, the underlying assumption is that under some conditions, error or reset states, software may not be active. Therefore, the clock switching is fully hardware based and is glitchless. To facilitate clock switching requests with software, the MUX_n_CSC and MUX_n_CSS registers implement request and status for the clock multiplexer, the rest is managed in hardware. Using these registers, the software can monitor the state of the hardware-based clock multiplexer

and also make clock switching requests. It is recommended that a new clock switch request is given only when there are no pending/ongoing clock switching requests. However, a switch to the safe clock, that is, clk_src_0, can be performed at any instance of time. A switch to the safe clock is always completed. Software should ensure that while making a software-based switch to safe clock, the register configuration clock should be available, at least for completing the write to the MUX_n_CSC register. This means that the clock should be running for the register write to complete. Hardware clock multiplexer also supports hardware-based switch to safe clock, which is requested externally to MC_CGM (for example, by MC_RGM). For a hardware-based switch to safe clock, it is not required to have a register configuration clock for MC_CGM.

NOTE

- "Switch to safe clock" from software is requested by programming the MUX_n_CSC[SAFE_SW] bit field only and not by combining the MUX_n_CSC[CLK_SW] and MUX_n_CSC[SELCTL] bit fields of the MUX_n_CSC register. Writes to other fields are ignored when requesting switch for safe clock.
- After the switch to the safe clock requested by the MC_RGM has completed, the MC_RGM also requests the clock dividers to switch to their default values. This hardware-triggered divider update can be monitored in the same manner as for software-configured updates, and software should use it to ensure that the configuration update has completed before reconfiguring the clocks.
- Write accesses to the MUX_n_CSC register with clock select pointing to the "reserved" input clock source are aborted with bus transfer errors.

See Figure 98 that shows the flowchart representation for the sequence of steps to be followed when making a clock switch request to hardware-controlled clock multiplexer. Switch to safe clock can be requested at any instance of time, and for clarity reasons, it is not shown in Figure 98.

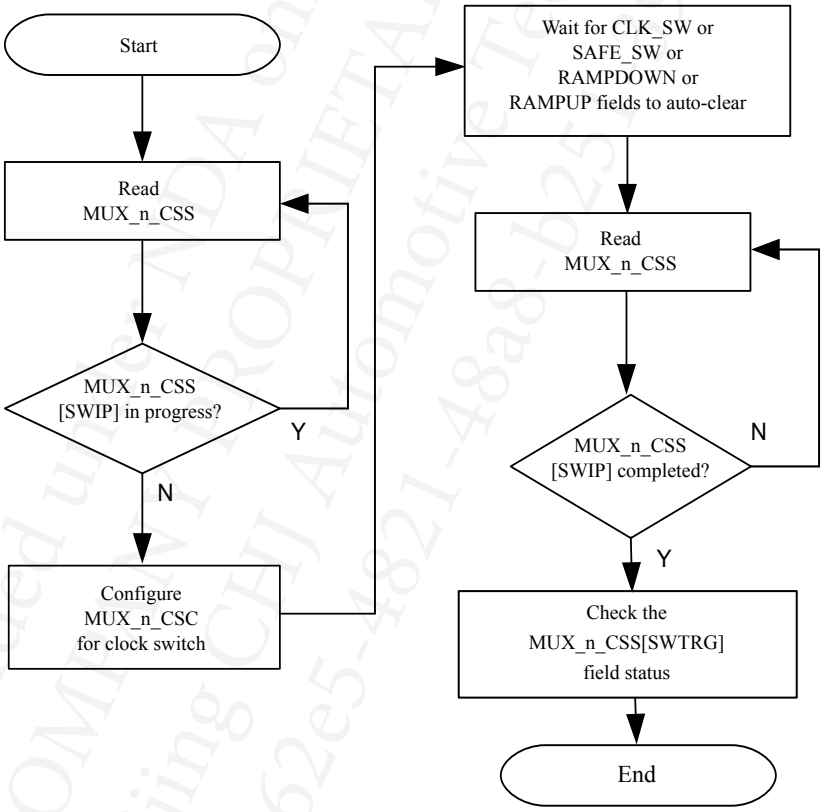


Figure 98. Flow for clock switch request on hardware-controlled clock multiplexer

NOTE

- A switch to the safe clock command always leads to a ramp-down from the currently selected clock and then a switch to "safe clock", except when there is an ongoing clock switch requested by the software without ramp-up and ramp-down. A safe clock switch request when there is an ongoing clock switch without ramp-up or ramp-down results in a switch to the safe clock without performing a ramp-down (either by MC_RGM or provided using the MUX_n_CSC register) does not perform a ramp-down before switching to "safe clock".
- The above flowchart steps can be preceded by points 1 and 3 below:
  1. Disable the divider.
  2. Switch clocks through hardware multiplexer.
  3. Enable and configure the dividers (atomic write instruction).

24.9.1.2 Software-controlled clock multiplexer

In software-based clock multiplexing, the underlying assumption is that the software is always available and there are no error or reset conditions in the chip. This implies that a glitchless switch between input clock sources of MC_CGM MUX can be achieved by following a sequence of steps in software. The software-based clock multiplexer implements a clock gate at the output of the clock mux. The software can gate the selected clock of MC_CGM MUX using a synchronous/graceful clock gate bit (that is, MUX_n_CSC[CG]) or a forced clock gate bit (that is, MUX_n_CSC[FCG]). The hardware does not guarantee that any glitches will escape when using forced clock gating. When a forced clock gate bit is written to 1, the internal clock gate forcefully gates the selected clock to logic-0, therefore, to avoid clock glitches, it should be ensured that the selected clock source is not running. See Figure 99 that shows the flowchart representation for the sequence of steps to be followed when making a clock switch request to software-controlled clock multiplexer. No switch to safe clock is supported in software-controlled clock multiplexer.

Provided under NDA of COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-4821-48a8-b251-3[illegible]624c1

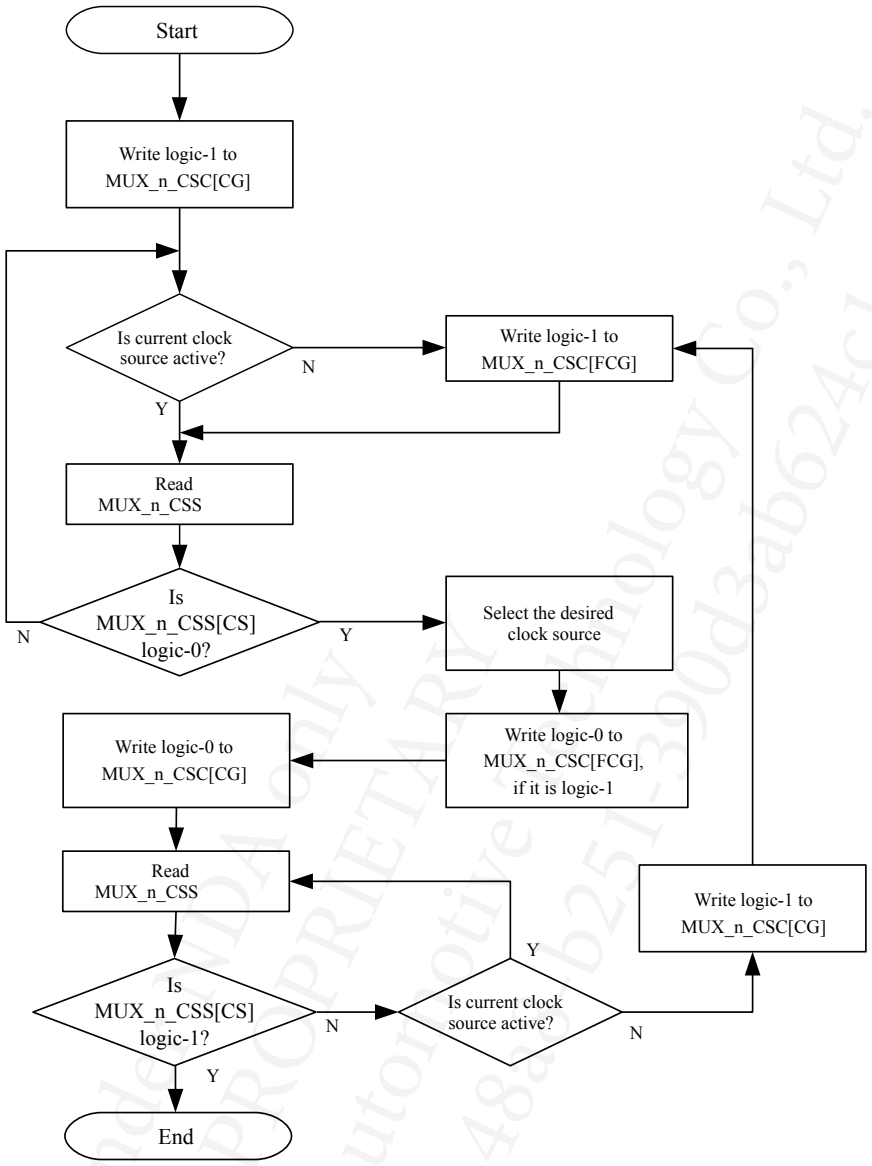


Figure 99. Flow for clock switch request on software controlled clock multiplexer

NOTE

- Ensure that before using the force gate bit, any IP or other logic using the clock of MC_CGM MUX is in the inactive state or a clock glitch resulting from usage of forced gating does not effect the IP (that is, it is clock gated after the MC_CGM).
- In Figure 99, the clock source to be selected should be active at the time of clock switch, else the MUX_n_CSS[CS] field will remain set to logic-0. In case the clock source to be selected becomes inactive (that is, loss of clock, and so on) during the switching operation, the switch to another clock source can be initiated by writing both the MUX_n_CSC[CG] and MUX_n_CSC[FCG] fields to logic-1.
- Writing a 'reserved' value for the MUX_n_CSC[SELCTL] field may result in an unpredictable clock at the output of the clock multiplexer.

NOTE

The above flowchart steps can be preceded by points 1 and 3 below:

1. Disable the divider.
2. Switch clocks through software multiplexer.
3. Enable and configure the dividers (atomic write instruction).

24.9.2 PCFS

MC_CGM implements PCFS when changing clock source at an MC_CGM clock mux. The PCFS is only implemented in a hardware-controlled clock multiplexer and not in a software-controlled clock multiplexer. It allows a gradual load change for a power/voltage supply unit by employing a gradual frequency changeover from one to another. The frequency changeover is achieved by clock division of input clock source with a sequence of division values, where frequency ramp down and ramp up is achieved when the sequence of division values are ascending and descending in nature, respectively. As the clock division (fractional) is implemented in digital logic, therefore, it is a coarse-level clock division rather than being an accurate-level division, implying that the duty cycle of the progressively divided clock can vary with time.

The PCFS feature is utilized when a drain current to frequency relationship is known, that is, for a given drain current what is the maximum allowed change in frequency (f_{chg}). The f_{chg} is the first input parameter for PCFS and other parameters for PCFS are specified or calculated in relation to clk_src_0.

The PCFS hardware generates the clock division factors based on certain values that are programmed into the PCFS configuration register. The following pseudo codes represent the generation of clock division value sequence (d_i).

```
/* ramp down division values (d_n) with k steps*/
      delta_1 = RATE/1000;
      delta_2 = RATE/1000;
      d_0     = 1.0;
      for i=1 to k-1 do
        d_i = d_i-1 + delta_1;
        delta_1 = delta_1 + delta_2;
      endfor
```

```
/* ramp up division values (d_n) with k steps*/
      delta_1 = RATE*k/1000;
      delta_2 = RATE/1000;
      d_0     = 1.0 + RATE*k*(k+1)/2;
      for i=1 to k-1 do
        d_i = d_i-1 - delta_1;
        delta_1 = delta_1 - delta_2;
    endfor
```

As the generation of clock division values is not a closed-bounded function, calculating RATE for a given f_{chg} is an iterative process. Find a value of RATE that produces a clock division sequence, which when used does not lead to a frequency change greater than f_{chg}. See Table 106 that tabulates some of the RATE values against a_{max}, where

$$a_{max}=f_{chg}/F_i$$

where, F_i is the frequency of i^{th} input clock source of the clock mux.

Table 106. PCFS RATE values

a_{max}	PCFS rate
0.005	12
0.01	48

Table continues on the next page...

Table 106. PCFS RATE values (continued)

a_{max}	PCFS rate
0.15	112
0.20	184

The last clock division factor in case of ramp-down or the first clock in case of ramp-up (following a ramp-down procedure), should be such that clock switch from any input clock source to another should be termed as safe, indicating that load changes have sustainable power effects. This frequency level is referred to as "safe frequency", equivalent to frequency of clk_src_0 referred to as "safe clock" with frequency "safe clock frequency" (f_{safe}). The last clock division factor in the sequence of clock division factors happens after "k" steps. The factor k (=steps) is calculated by using the following formula:

$$k = ceil(0.5 + sqrt(0.25 - (2000 * (1 - (F_i/f_{safe})) / RATE)))$$

Using the above formula, all the PCFS register configuration values can be calculated for a given frequency of a clock source:

$$PCFS\_DIVEi.DIV = (F_i/f_{safe})*1000-1;$$

```
        PCFS_DIVCi.INIT = RATE * k;
        PCFS_DIVCi.RATE = RATE;
        PCFS_DIVSi.DIV  = 999 + (RATE * k * (k+1)/2);
```

See Figure 100 that shows a graphical representation of the change in frequency, which happens during PCFS ramp-down and ramp-up.

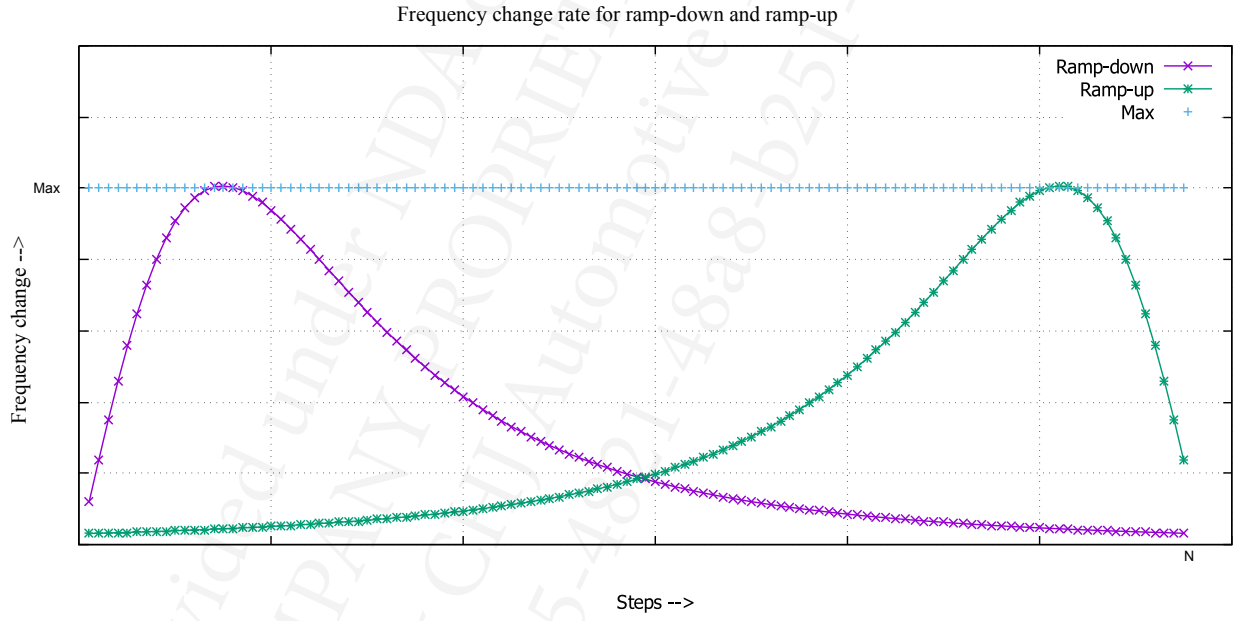


Figure 100. PCFS steps vs frequency changes

For any given clock source, if its frequency is less than that of clk_src_0, then its corresponding registers should be programmed to default values, where the default values are such that PCFS divider start and PCFS divider end values are same and equal to divide-by-1. The default values ensure that no progressive clock division is done when a clock switch request is given to switch from or to that source.

NOTE

Calculate the minimum frequency during the PCFS RAMPDOWN and RAMPUP operations by using the formula:

```
(clk_src_0/((PCFS_DIVE+1)/1000)) MHz
```

24.9.2.1 PCFS control

The PCFS operation is configured by a set of configuration registers. One set pertains to the calculation of the clock division factors, which are PCFS_DIVC, PCFS_DIVE, PCFS_DIVS, and PCFS_SDUR, while the registers MUX_n_CSC and MUX_n_CSS implement trigger and status of the PCFS operation. The clock division factors are expected to be programmed before any other MC_CGM operation is initiated and remain unchanged. All the registers corresponding to the clock division factors should be programmed with clk_src_0 as the configuration clock and before doing any clock switch or PCFS operation on any of the MC_CGM mux. It should be noted that the default values of the register corresponding to clock division factors are such that only the clock division factor is calculated by hardware that is divide-by-1. The PCFS operation is always triggered when the safe clock request is generated except when there is an on-going clock switch without ramp-up or ramp-down. Therefore, the software needs to ensure that the PCFS configuration is complete and correct.

While configuring MUX_n_CSC, only valid PCFS and clock switch requests should be provided. PCFS or clock switch requests should only be provided if the PCFS operation is in the idle state. If there is an ongoing PCFS operation, it is recommended not to provide any new PCFS triggers (except switch to safe clock) until the ongoing operation is completed. Switch to safe clock via hardware or by register configuration can be provided at any instance of time and is always completed. Valid combinations of PCFS and clock switches triggers are listed in Table 107. All the PCFS commands should be atomic in nature, which means a single register write should provide complete PCFS sequence to be executed that is ramp-down, clock switch, and ramp-up.

Table 107. Valid PCFS and clock switch requests

PCFS operation state	Command
Idle	Ramp-down, clock switch, and ramp-up
Idle	Clock switch only (without ramp-up or ramp-down)

When a switch to safe clock is provided by writing to MUX_n_CSC, then writes to other register fields are ignored.

24.9.2.2 Clock source powerup and selection

This section provides guidelines for powering up a given clock source and selecting it at the MC_CGM clock multiplexer.

Following is the powerup procedure for a clock source:

1. Configure the parameter, if any.
2. Configure the powerup (or powerdown) control field.
3. Wait for the powerup status indication.

After a powerup indication, the clock source can be selected to provide output clock at an MC_CGM clock multiplexer. A clock-monitoring setup can also be activated on the powered-up clock source.

See Figure 101 that shows a flow chart representation of this sequence.

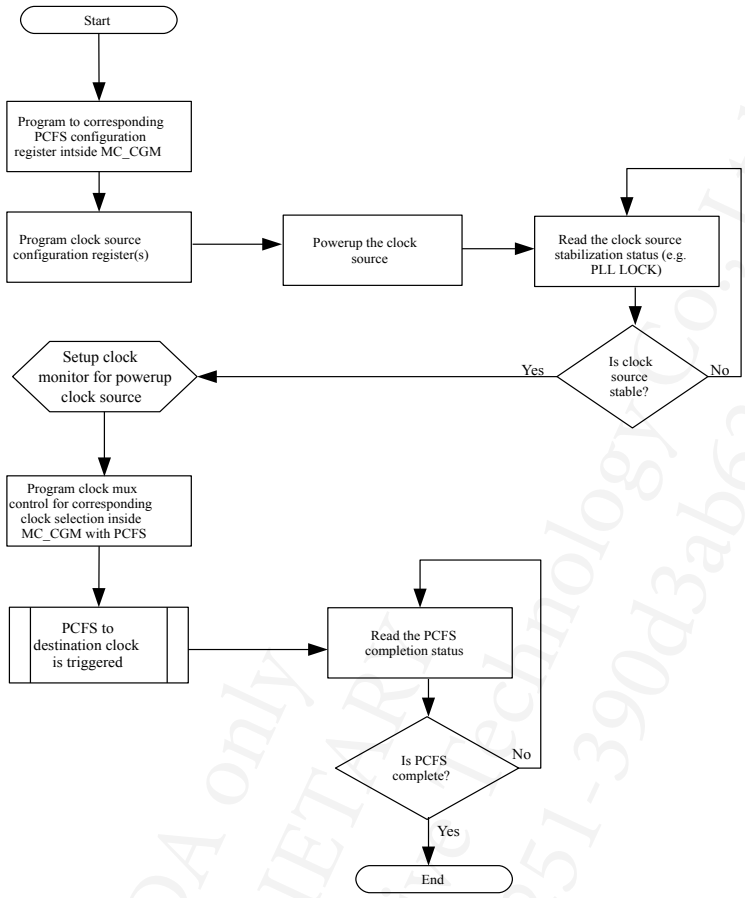


Figure 101. Clock source powerup and selection procedure

24.9.2.3 Clock source powerdown and deselection

This section provides guidelines for powering down a given clock source and deselecting it at MC_CGM clock multiplexer.

Following is the powerdown procedure for a clock source:

1. Deselect the clock source at all MC_CGM clock multiplexers.
2. Configure the powerup (or powerdown) control field.
3. Wait for the powerdown status indication.

See Figure 102 that shows a flow chart representation of this sequence.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4888-b251-390d3ab624c1

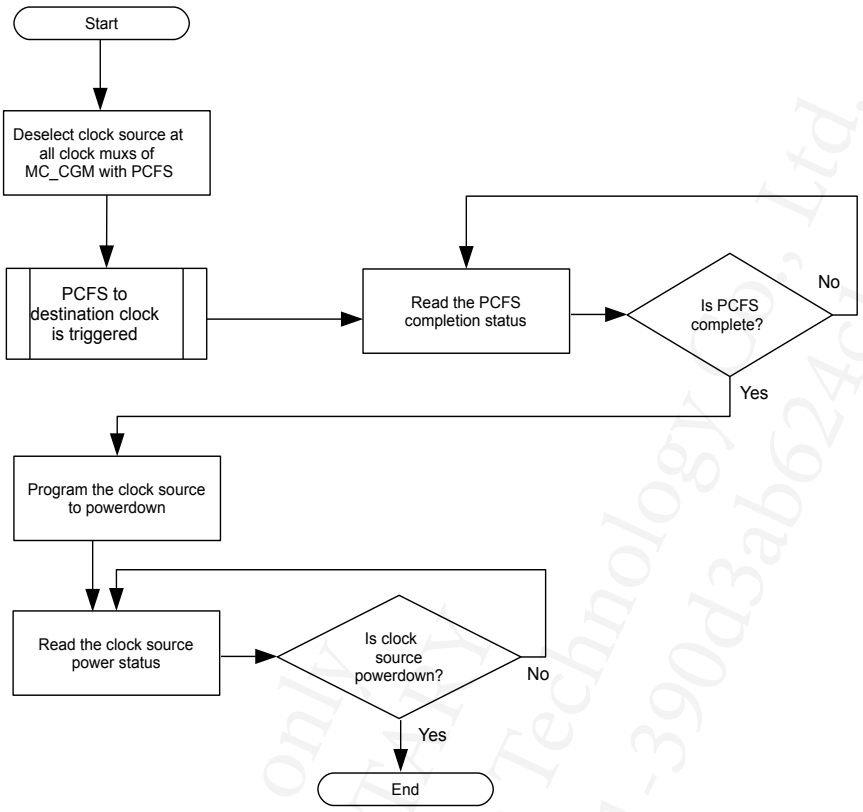


Figure 102. Clock source powerdown and clock source deselection procedure

24.9.2.4 Clock switch with load change

This section provides guidelines to switch between two high-frequency clock sources along with load changes in the system. Load change is referred to switching ON or OFF of logic/peripherals in the system, which has an effect of significant capacitance changes on the chip. This triggers the voltage regulation for the chip.

When a large number of peripherals or digital logic is enabled or disabled, it is recommended that this step should be performed at a low frequency. Independent of whether a clock switch is required, this criteria needs to be met. When a clock switch is required at two high frequencies, the recommended sequence is as follows:

1. Intermediately switch to clk_src_0.
2. Change load (that is, enable/disable peripherals).
3. Switch to the target clock.

See the following figure that shows a flow chart representation of this sequence.

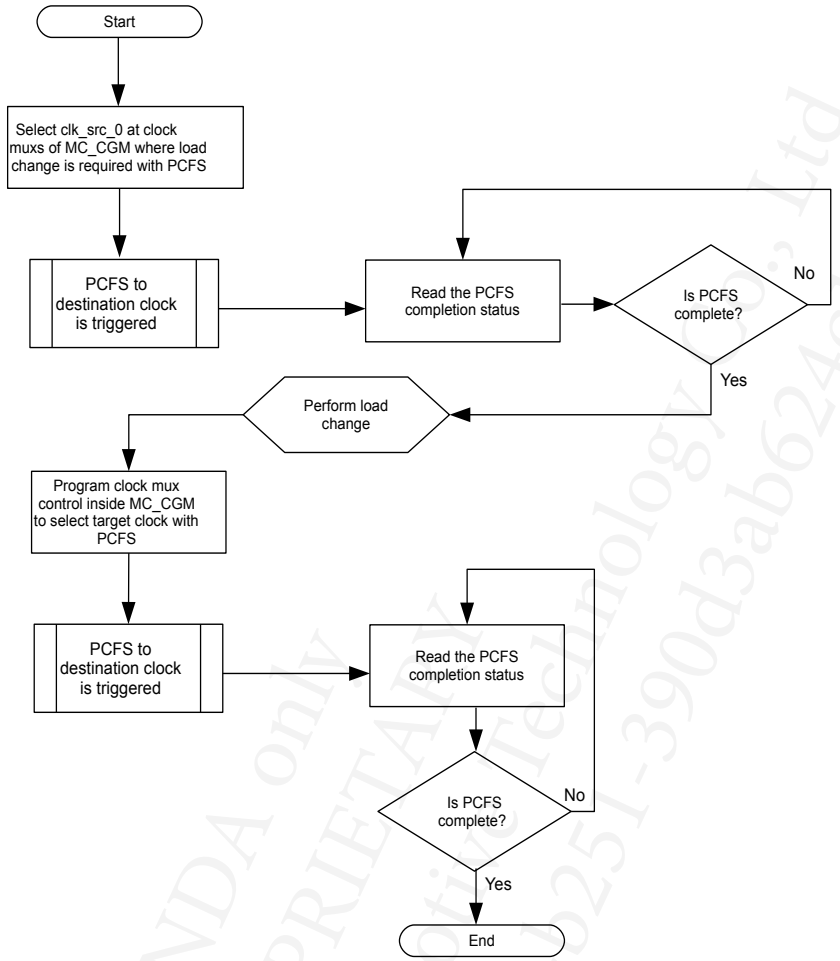


Figure 103. Clock switch with load change procedure

24.9.3 Clock dividers

Clock dividers are used for the generation of a divided clock that is used for running IPs or peripherals. The MC_CGM provides the following built-in dividers at each clock mux:

- 50% clock dividers

Each divider can be controlled by the Divider Enable (DE) bit and the Division Value (DIV) field. If a divider has its DE bit set to logic-0 in the respective configuration register, then that divider is disabled and the output divided clock is held to logic-0. If the DE bit is logic-1, the divider is enabled and provides a divided clock according to the value set in the DIV field.

24.9.3.1 50% clock divider

50% duty cycle dividers generate a real divided clock. The division factor is always an integer but is not restricted to even numbers. The rising edge of the divided clock is always synchronous to the rising edge of the divider source clock, but the falling edge is synchronous to the rising edge or the falling edge depending on whether the division factor is even or odd, respectively. If the input clock has a duty cycle of 50%, the divided output clock maintains the same 50% duty cycle. See Figure 104 that shows the 50% clock divider operation and its associated signals.

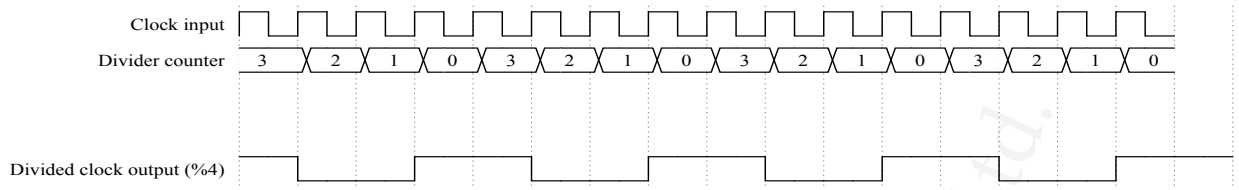


Figure 104. 50% clock divider programmed for divide by 4 (DIV=3)

A 50% divided clock can be considered asynchronous (not edge aligned) to other divided clocks from dividers at the same clock mux. 50% clock dividers are implemented without an active closed loop, and are expected not to get stuck if the input clock glitches for a single cycle.

24.9.3.2 Clock dividers update

To update the division value or the divider enable, the software should follow the procedure as shown in Figure 105. These updates happen only after the current division cycle has elapsed. However, if the phase of the clock divider is updated, the update happens independently from the state of the division cycle. Any update to the clock divider fields does not result in clock glitch either at the divided clock output or the phase-divided clock output.

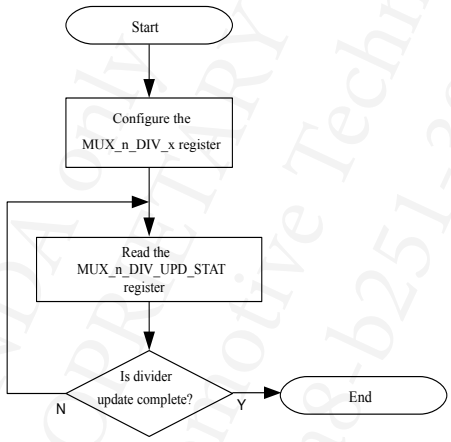


Figure 105. Flow for clock divider programming update

Recommended software sequence for ensuring no undivided output at MC_CGM:

1. Reset is de-asserted
2. MC_RGM goes to IDLE
3. Enable the clock dividers of MC_CGM to provide FIRC clock so that reset of fixed dividers can be lifted.
4. Program the MC_CGM as per use case division values.
5. Switch the clock of MC_CGM Mux to desired one, and run the system.

24.10 Glossary

PCFS	Progressive clock frequency switching
LCM	Least common multiple

Chapter 25
Digital Frequency Synthesizer (DFS)

25.1 Introduction

This block generates different clock outputs of programmable frequency from an input clock source provided by a corresponding PLL. The DFS block provides clock signals as outputs on its ports as per the respective configuration of the ports. Also, the block provides an indication of lock for each output port.

25.1.1 Block diagram

Here is the DFS block diagram:

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

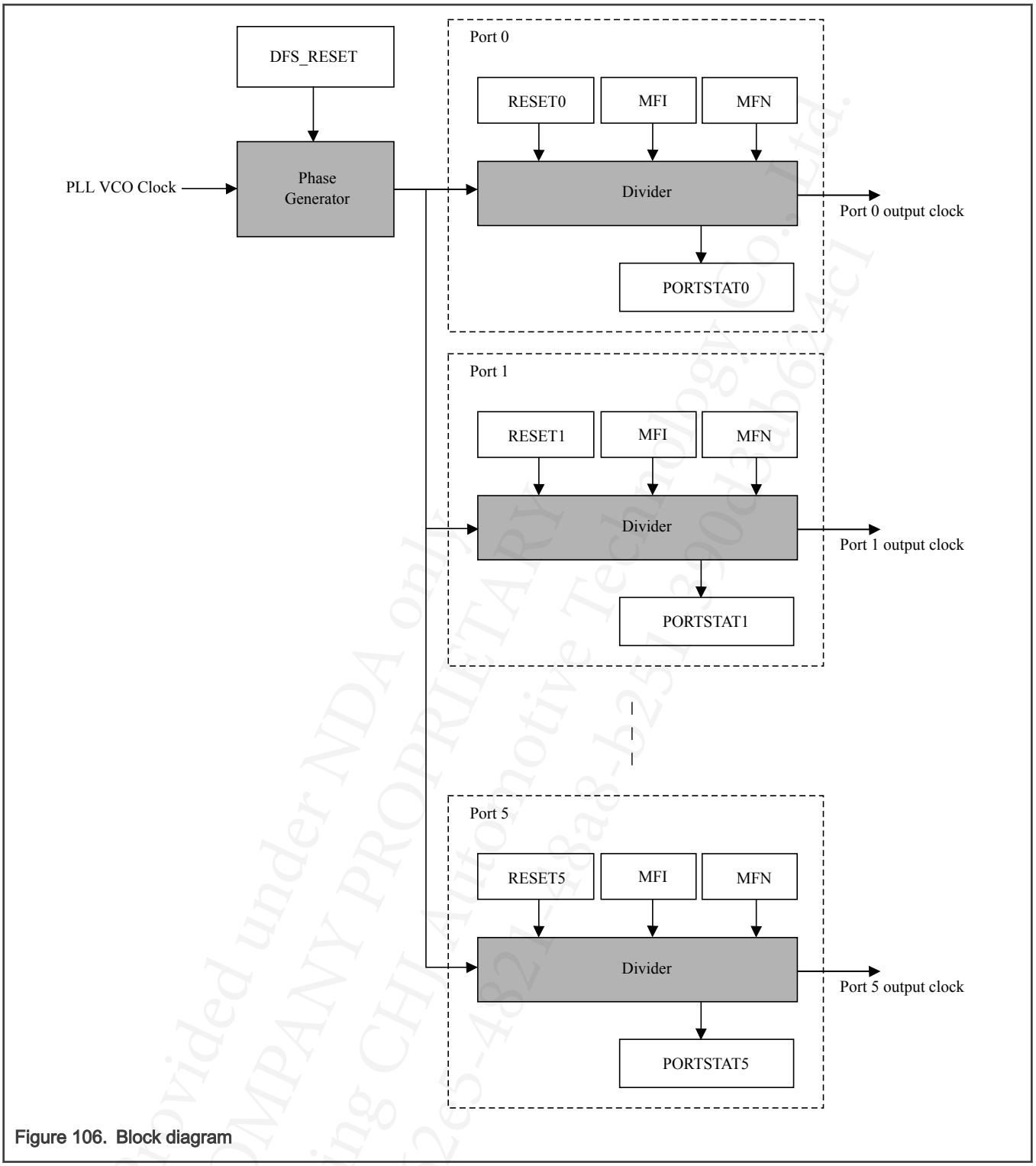


Figure 106. Block diagram

25.1.2 Features

DFS includes the following features:

- Six fractional clock dividers, with
  - individually configurable division factors.

  - individual enable/disable control.
  - individual lock status reporting.
- software controllable master DFS reset.

25.1.3 Modes of operation

The following lists the various supported DFS modes of operation.

Table 108. Modes of operation

Mode	CTL[RESET]	PORTRESET[RESET*n*]	Description
Non-functional	1	x	The DFS phase generator is in reset.
Functional	0	0	DFS port *n* is enabled.
	0	1	DFS port *n* is disabled.

Any combination of enabled and disabled ports is supported when the DFS is in functional mode.

25.2 Initialization information

Initialization

The following steps describe the procedure for DFS initialization:

1. Place the master DFS and output ports of analog DFS in reset by:
   - Writing 3Fh to PORTRESET[RESET*n*]
   - Check that PORTSR is 0.
   - Writing 1 to CTL[DFS_RESET]
2. Set the PLL clock output for the analog DFS to stabilize and lock the PLL.
3. Configure each DFS port with the desired division factor via the DVPORT*n* registers.
4. Write a 0 to CTL[DFS_RESET] to deassert the reset to master RESET of the analog DFS block.
5. Write a 0 to PORTRESET[RESET*n*] to deassert the reset to the output ports of the analog DFS block.
6. Poll PORTSR until all configured analog DFS output ports are locked.

Re-programing a DFS port without disabling the main DFS reset

The following steps describe the procedure for re-programing a DFS port without disabling the main DFS reset (when DFS ports have already been locked):

1. Program DFS using the procedure mentioned in the aforementioned initialization sequence.
2. Assert port reset of output port whose programming you want to change by configuring the corresponding field in PORTRESET.
3. Wait for the corresponding field in PORTSR to become 0.
4. Change configuration of MFI and MFN in the corresponding divide register (DVPORT0).
5. Deassert port reset of output port whose programming you want to change by configuring the corresponding field in PORTRESET.
6. Poll PORTSR to check if the corresponding DFS output port is locked.

Initialization after DFS LOL:

The following steps describe the procedure for initialization after DFS LOL:

1. Clear DFS LOL by writting 1 to PORTLOLSR[LOLF*n*]
2. Place the master DFS and output ports of analog DFS in reset by:
    a. Writing 3Fh to PORTRESET[RESET*n*].
    b. Writing 1 to CTL[DFS_RESET].
3. Set the PLL clock output for the analog DFS to stabilize and lock the PLL.
4. Configure each DFS port with the desired division factor via the DVPORT*n* registers.
5. Write 0 to CTL[DFS_RESET] to deassert the reset to master RESET of the analog DFS block.
6. Wait for 200 ns.
7. Write 0 to PORTRESET[RESET*n*] to deassert the reset to the output ports of the analog DFS block.
8. Poll PORTSR until all configured analog DFS output ports are locked.

Disabling DFS

The following steps describe the procedure to disable DFS:

1. Write 3Fh to PORTRESET[RESET*n*].
2. Wait for PORTSR to be 0.
3. Write 1 to CTL[DFS_RESET].

25.3 DFS register descriptions

This section provides the memory map and the detailed description of all the DFS registers.

The memory map shows address offsets from the module base address.

25.3.1 DFS memory map

CORE_DFS base address: 4005_4000h

PERIPH_DFS base address: 4005_8000h

Offset	Register	Width (In bits)	Access	Reset value
Ch	Port Status Register (PORTSR)	32	RO	0000_0000h
10h	Port Loss of Lock Status (PORTLOLSR)	32	W1C	0000_0000h
14h	Port Reset (PORTRESET)	32	RW	0000_003Fh
18h	Control (CTL)	32	RW	0000_0002h
1Ch - 30h	Divider for Port n (DVPORT0 - DVPORT5)	32	RW	0000_0000h

Preliminary Information
COMPANY CONFIDENTIAL
Beijing eSOL Automotive Technology Co., Ltd.
c8c46e54-821-48a8-b251-390d3ab624c1

25.3.2 Port Status Register (PORTSR)

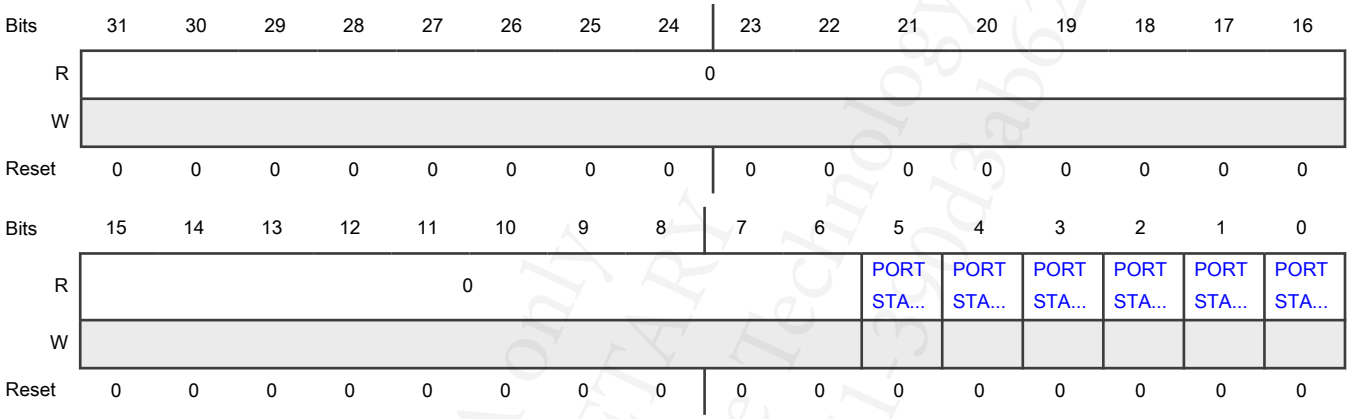
Offset

Register	Offset
PORTSR	Ch

Function

This register reflects the lock status of DFS output ports.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 PORTSTATn	Lock status for port n This field reflects the lock status of the DFS output ports of the DFS analog block. Each bit represents a port from the analog block. For example, PORTSR[PORTSTAT[0]] represents port 0. 0b - Port *n* is not locked. 1b - Port *n* is locked.

25.3.3 Port Loss of Lock Status (PORTLOLSR)

Offset

Register	Offset
PORTLOLSR	10h

Function

This register contains the loss-of-lock status flags for the DFS output ports.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										LOLF5	LOLF4	LOLF3	LOLF2	LOLF1	LOLF0
W											W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-6 —	Reserved
5-0 LOLFn	Loss of lock flag DFS IP does not have real time clock monitor to declare lock or loss of lock. If PLL IP is disabled before DFS then loss of lock will be generated. This field is sticky. This means that if lock is re-acquired, the field remains 1 until you clear it by writing 1 to it or by asserting reset. 0b - The DFS detects no loss of lock for channel port *n* output. 1b - The DFS detected a loss of lock for channel port *n* output.

25.3.4 Port Reset (PORTRESET)

Offset

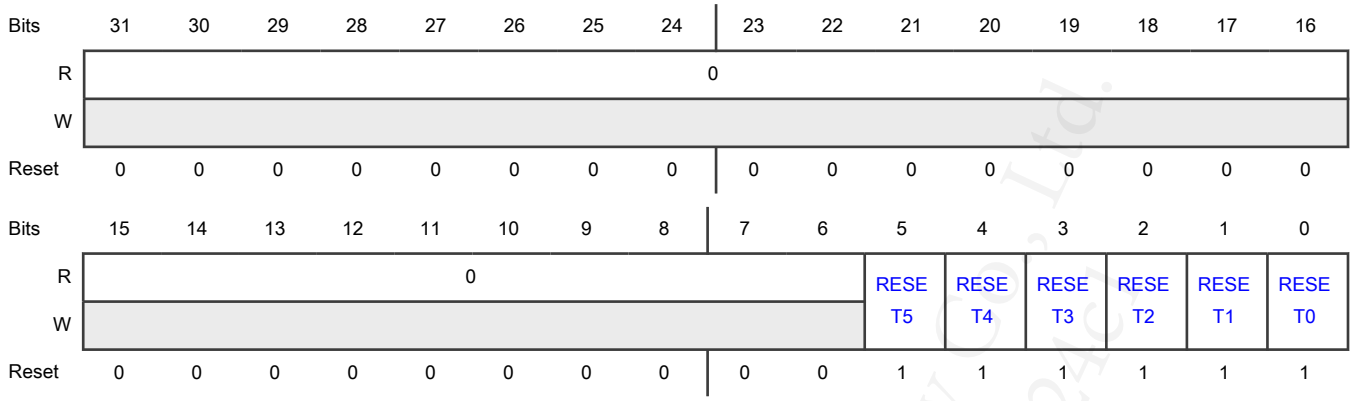
Register	Offset
PORTRESET	14h

Function

This register controls the enabling and disabling of the DFS output ports.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-8251-390d8ab624

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 RESETn	Enable control for port *n* 0b - Port *n* is enabled. 1b - Port *n* is disabled.

25.3.5 Control (CTL)

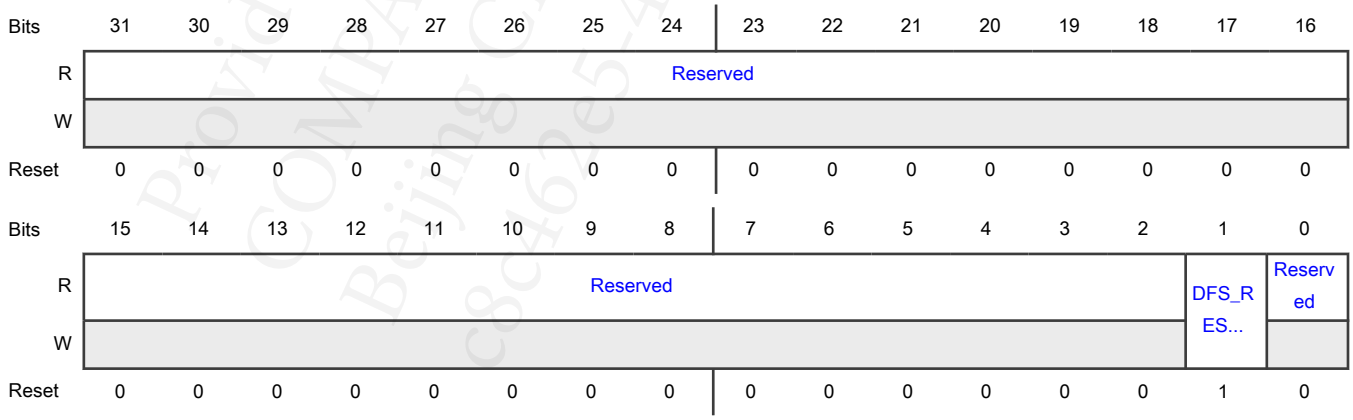
Offset

Register	Offset
CTL	18h

Function

This register configures the DFS.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e54-4821-48a8-b251-390d3ab624c2

Fields

Field	Function
31-2 —	Reserved
1 DFS_RESET	This field controls the master reset of the DFS. For sequencing with PORTRESET[RESET] bits. 0b - The DFS phase generator is out of reset. You can enable individual ports by programming Port Reset (PORTRESET). 1b - The DFS phase generator is in reset. You cannot enable any of the ports.
0 —	Reserved

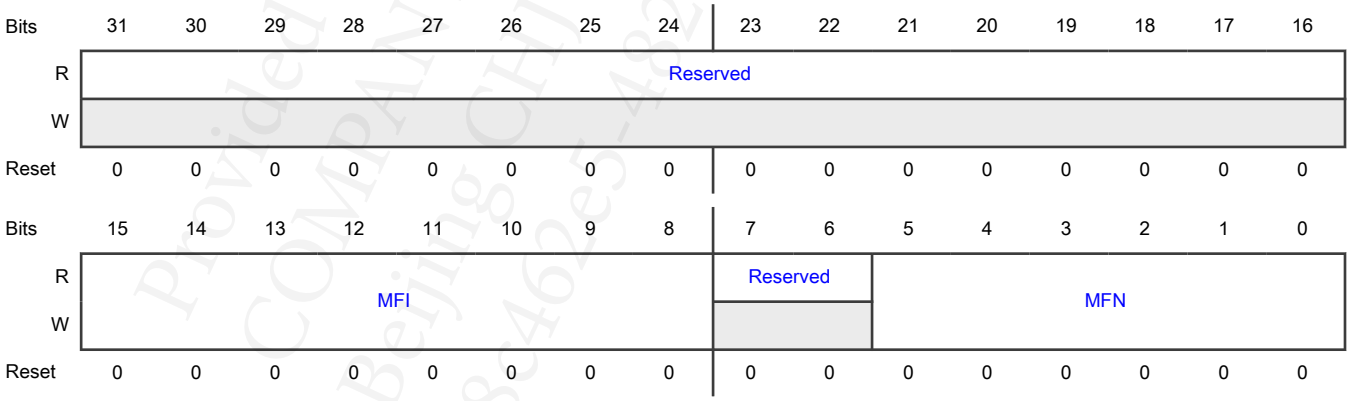
25.3.6 Divider for Port n (DVPORT0 - DVPORT5)

Offset

Register	Offset
DVPORT0	1Ch
DVPORT1	20h
DVPORT2	24h
DVPORT3	28h
DVPORT4	2Ch
DVPORT5	30h

Function
This register holds the divider configurations for the DFS output ports.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8462e54-8821-48a8-b251-390d3ab62fc1

Fields

Field	Function
31-16 —	Reserved
15-8 MFI	This field provides the integer part of division factor for port *n*.
7-6 —	Reserved
5-0 MFN	This field provides the numerator of fractional part of division factor for port *n*.

25.4 Functional description

25.4.1 Port divider configuration

DFS IP takes input clock from PLL and generates multiple phases of clock. Six independent phase dividers then use these phases. You control the division factor of each port's divider by programming the DVPORT*n*[MFI] and DVPORT*n*[MFN] fields. The following equation describes the relationship between the input and output clock of each phase divider:

$$f_{\text{dfsclkout}}n = \frac{f_{\text{dfsclkin}}n}{2\times\left(\text{DVPORTn[MFI]}+\frac{\text{DVPORTn[MFN]}}{36}\right)}$$

Figure 107. Equation for input and output clocks of each port divider

In this equation:

- *n* is the DFS output port number.
- DVPORT*n*[MFI] is the integer part of the division value, and can range from 1d to 255d.
- DVPORT*n*[MFN] is the numerator of the fractional part of the division value, and can range from 0d to 35d.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-42a8-b251-90d3ab024c1

Chapter 26
PLL Digital Interface (PLLDIG)

26.1 Introduction

The PLL can multiply or divide the frequency of a given clock input.

26.1.1 Features

PLL includes the following features:

- Provision for selecting multiple clock sources as the reference clock
- Programmable frequency modulation
- Multiple integer dividers on PLL outputs
- Lock detection circuitry reports when the PLL achieves frequency lock
- Continuous monitoring of lock status to report Loss of Lock (LOL) condition
- Powering down the module for low-power operation (Power-down mode)

26.1.2 Block diagram

The following figure shows the block diagram for PLL.

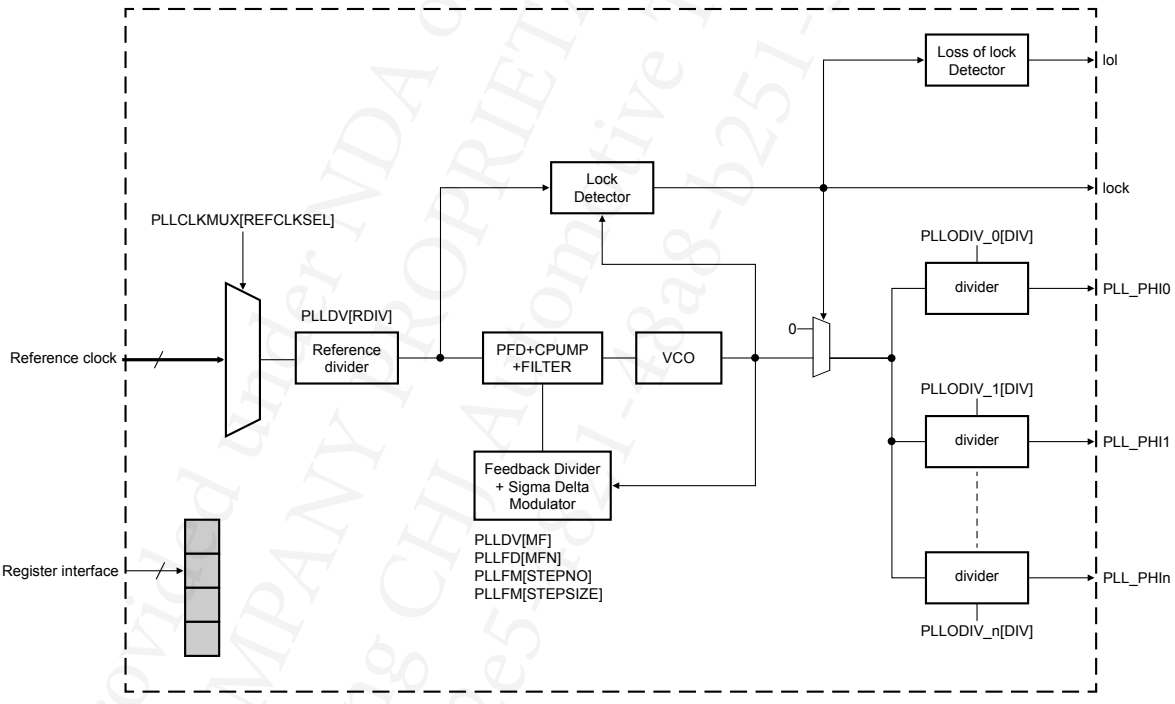


Figure 108. Block diagram

Number of output dividers can vary with the module instance. Refer to the Clocking chapter to confirm the number of PLL output dividers.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHIPSAIL Automotive Technology Co., Ltd.
c8c162e5-3a81-4a8-b251-890d3ab624c1

26.2 Modes of operation

Table 109. Modes of operation

PLLCR[PLLPD]	PLLFD[SDMEN]	PLLFM[SSCGBYP]	Description
1	x	x	PLL is disabled.
0	0	1	Functional mode – PLL operates in integer-only mode. See Clock configuration.
0	1	1	Functional mode – PLL operates in Fractional mode (non-Frequency modulation). See Clock configuration.
0	1	0	Functional mode – PLL operates in Frequency Modulation mode. See Frequency modulation.

26.3 ACCEL_PLL register descriptions

This section provides the memory map and detailed descriptions of registers used for configuring the PLL. The table below shows the memory map. Addresses are given as offsets from the module base address. All registers are accessed using 8-bit, 16-bit, or 32-bit addressing.

26.3.1 ACCEL_PLL memory map

ACCEL_PLL base address: 4004_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PLL Control Register (PLLCR)	32	RW	8000_0000h
4h	PLL Status Register (PLLSR)	32	W1C	0000_0300h
8h	PLL Divider (PLLDV)	32	RW	0C3F_1032h
Ch	PLL Frequency Modulation (PLLFM)	32	RW	4000_0000h
10h	PLL Fractional Divider (PLLFD)	32	RW	0000_0000h
20h	PLL Clock Multiplexer (PLLCLKMUX)	32	RW	0000_0000h
80h - 84h	PLL Output Divider (PLLODIV_0 - PLLODIV_1)	32	RW	0000_0000h

26.3.2 PLL Control Register (PLLCR)

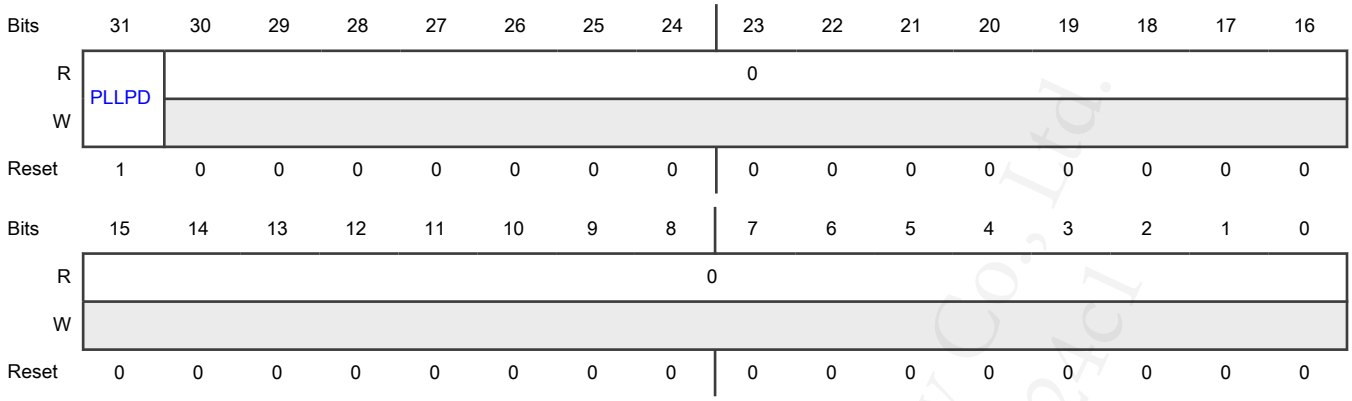
Offset

Register	Offset
PLLCR	0h

Function

Configures PLL functionality.

Diagram



Fields

Field	Function
31 PLLPD	PLL power down Powers down PLL. 0b - PLL is powered up 1b - PLL is powered down
30-0 —	Reserved

26.3.3 PLL Status Register (PLLSR)

Offset

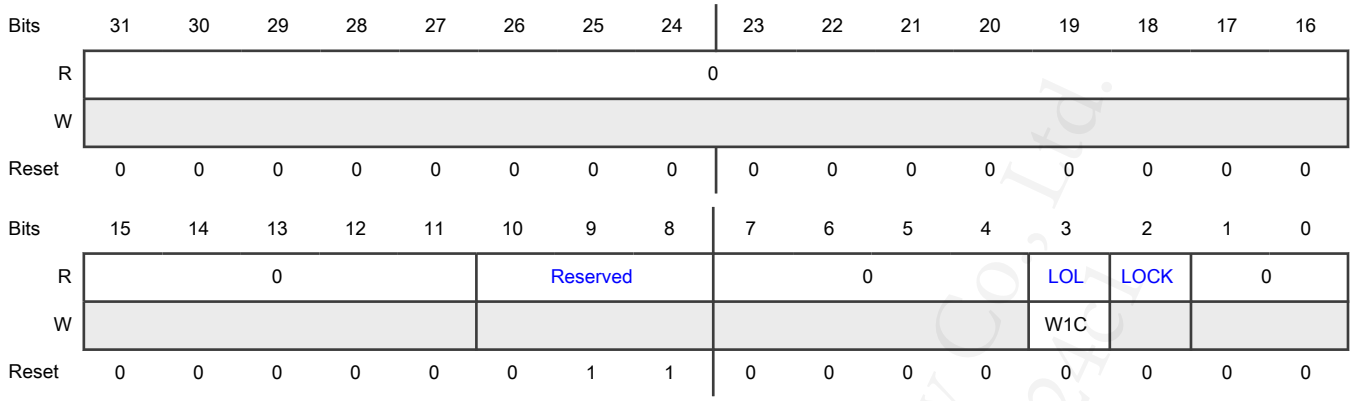
Register	Offset
PLLSR	4h

Function

Shows the PLL status.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-11 —	Reserved
10-8 —	Reserved
7-4 —	Reserved
3 LOL	Loss-Of-Lock flag Indicates the current PLL lock status. 0b - No loss of lock detected 1b - Loss of lock detected
2 LOCK	Lock status Indicates that PLL has acquired lock. 0b - PLL is unlocked 1b - PLL is locked
1-0 —	Reserved

26.3.4 PLL Divider (PLLDV)

Offset

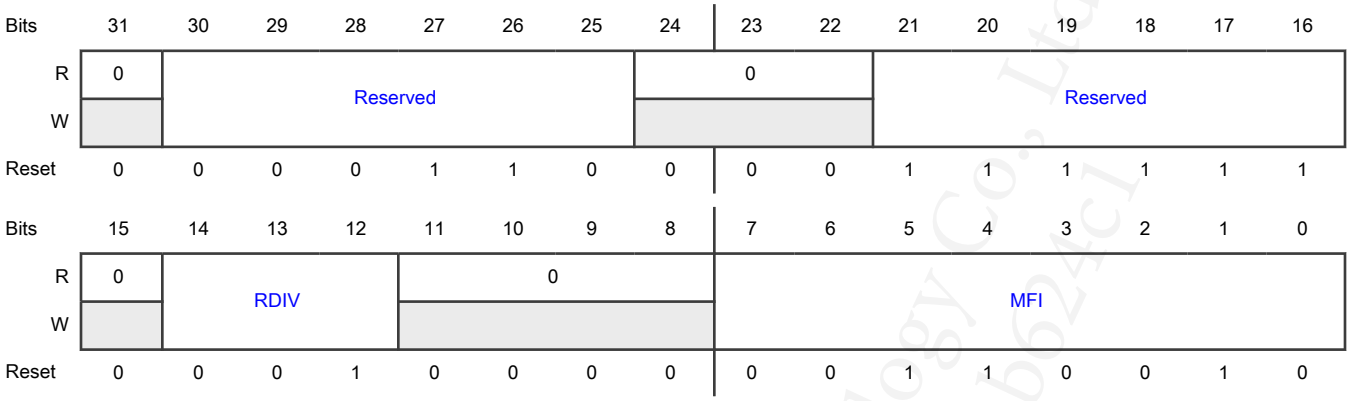
Register	Offset
PLLDV	8h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8ab62

Function

Divides input clocks for PLL output generation.

Diagram



Fields

Field	Function
31 —	Reserved
30-25 —	Reserved
24-22 —	Reserved
21-16 —	Reserved
15 —	Reserved
14-12 RDIV	Input clock predivider Sets the input clock divider. The output of the predivider circuit generates the PLL loop reference clock. 000b - Divide by 1 001b - Divide by 1 010b - Divide by 2 011b - Divide by 3 100b - Divide by 4 101b - Divide by 5 110b - Divide by 6

Table continues on the next page...

Table continued from the previous page...

Field	Function
	111b - Divide by 7
11-8 —	Reserved
7-0 MFI	Integer portion of loop divider. Sets the value of the divider in the PLL feedback loop, and the value specified establishes the multiplication factor applied to the reference frequency. Divider value = MFI, where the chosen MFI does not violate VCO frequency specifications.

26.3.5 PLL Frequency Modulation (PLLFM)

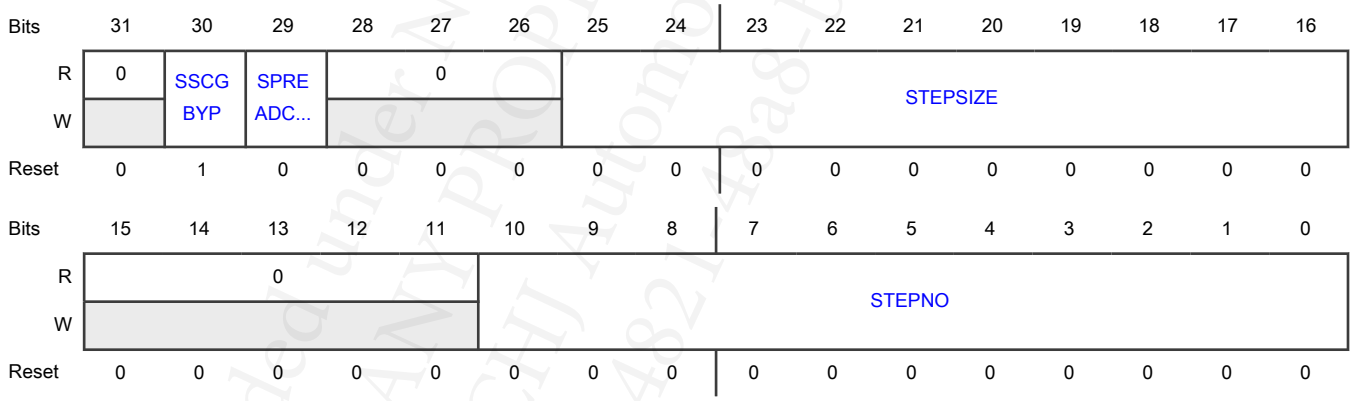
Offset

Register	Offset
PLLFM	Ch

Function

Configures PLL frequency modulation parameters.

Diagram



Fields

Field	Function
31 —	Reserved
30	Frequency modulation (spread spectrum clock generation) bypass

Table continues on the next page...

Table continued from the previous page...

Field	Function
SSCGBYP	Allows frequency modulation bypassing. 0b - Not bypassed 1b - Bypassed
29 SPREADCTL	Modulation type selection States that the modulation is centered around the nominal frequency. 0b - Modulation centered around nominal frequency 1b - Reserved
28-26 —	Reserved
25-16 STEPSIZE	Frequency modulation step size In Frequency Modulation mode, STEPSIZE provides the step size for modulation depth and frequency (see Frequency modulation).
15-11 —	Reserved
10-0 STEPNO	Number of steps of modulation period/frequency modulation In Frequency Modulation mode, STEPNO provides the number of steps to achieve modulation depth (see Frequency modulation).

26.3.6 PLL Fractional Divider (PLLFD)

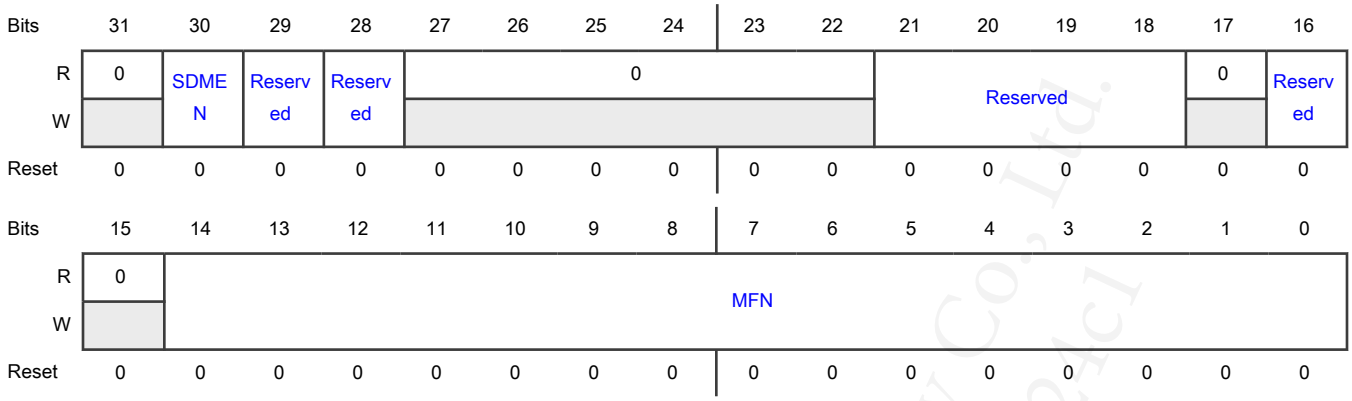
Offset

Register	Offset
PLLFD	10h

Function
Enables and configures frequency modulation.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-3901ab624c1

Diagram



Fields

Field	Function
31 —	Reserved
30 SDMEN	Fractional mode enable Enables Fractional mode. 0b - Disabled 1b - Enabled
29 —	Reserved
28 —	Reserved
27-22 —	Reserved
21-18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-0 MFN	Numerator of fractional loop division factor The value of this field must be less than 18432. In integer mode, it must be set to 15'd0.

26.3.7 PLL Clock Multiplexer (PLLCLKMUX)

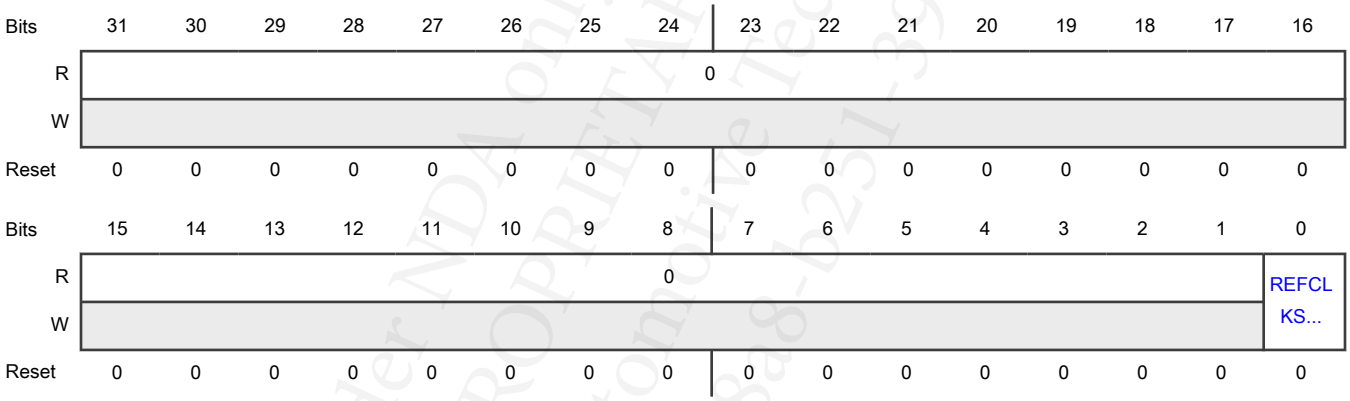
Offset

Register	Offset
PLLCLKMUX	20h

Function

Selects the PLL clock source.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 REFCLKSEL	Reference clock select Selects the PLL clock source. 0b - FIRC_CLK is source 1b - FXOSC_CLK is source

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-590d3ab624c1

26.3.8 PLL Output Divider (PLLODIV_0 - PLLODIV_1)

Offset

Register	Offset
PLLODIV_0	80h
PLLODIV_1	84h

Function

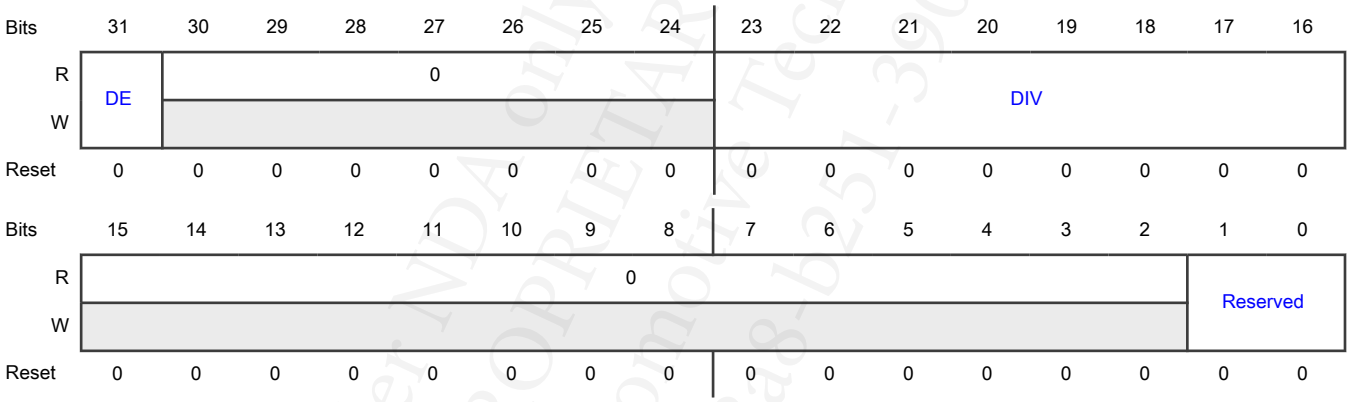
These registers control the PLL output clock divider settings. This divider is 50% duty cycle divider.

NOTE

These registers support only word accesses. Other write accesses lead to the following:

- Unpredictable behavior
- No transfer error getting generated

Diagram



Fields

Field	Function
31 DE	Divider enable Enables PLL output divider. Divider must be disabled before disabling PLL. 0b - Disabled 1b - Enabled
30-24 —	Reserved
23-16 DIV	Division value Provides the division value for the output clock divider. The clock period of the clock after division is DIV + 1 times the time period of the divider input clock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-2 —	Reserved
1-0 —	Reserved Do not write any value other than the reset value.

26.4 CORE_PLL register descriptions

This section provides the memory map and detailed descriptions of registers used for configuring the PLL. The table below shows the memory map. Addresses are given as offsets from the module base address. All registers are accessed using 8-bit, 16-bit, or 32-bit addressing.

26.4.1 CORE_PLL memory map

CORE_PLL base address: 4003_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PLL Control Register (PLLCR)	32	RW	8000_0000h
4h	PLL Status Register (PLLSR)	32	W1C	0000_0300h
8h	PLL Divider (PLLDV)	32	RW	0C3F_1032h
Ch	PLL Frequency Modulation (PLLFM)	32	RW	4000_0000h
10h	PLL Fractional Divider (PLLFD)	32	RW	0000_0000h
20h	PLL Clock Multiplexer (PLLCLKMUX)	32	RW	0000_0000h
80h - 84h	PLL Output Divider (PLLODIV_0 - PLLODIV_1)	32	RW	0000_0000h

26.4.2 PLL Control Register (PLLCR)

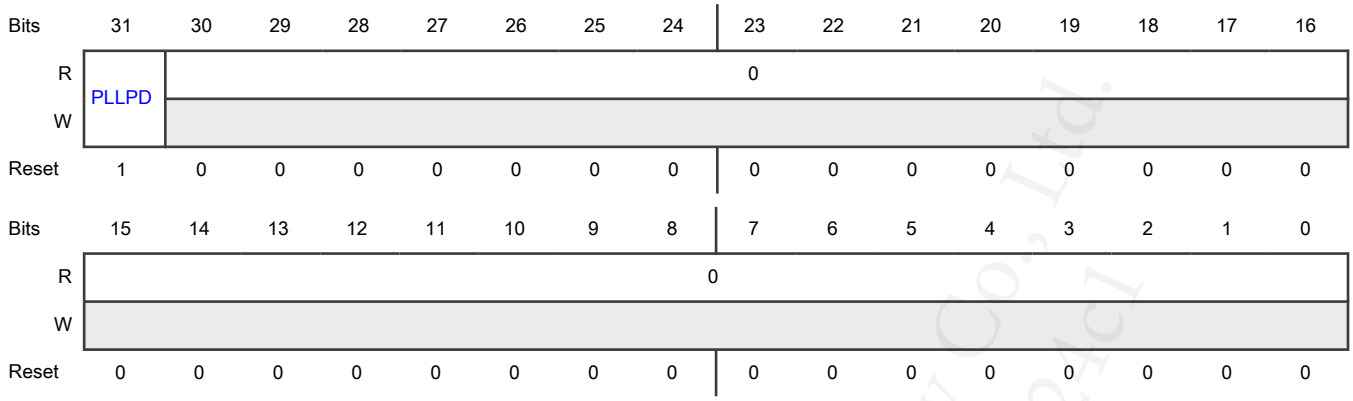
Offset

Register	Offset
PLLCR	0h

Function
Configures PLL functionality.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab9b4c1

Diagram



Fields

Field	Function
31 PLLPD	PLL power down Powers down PLL. 0b - PLL is powered up 1b - PLL is powered down
30-0 —	Reserved

26.4.3 PLL Status Register (PLLSR)

Offset

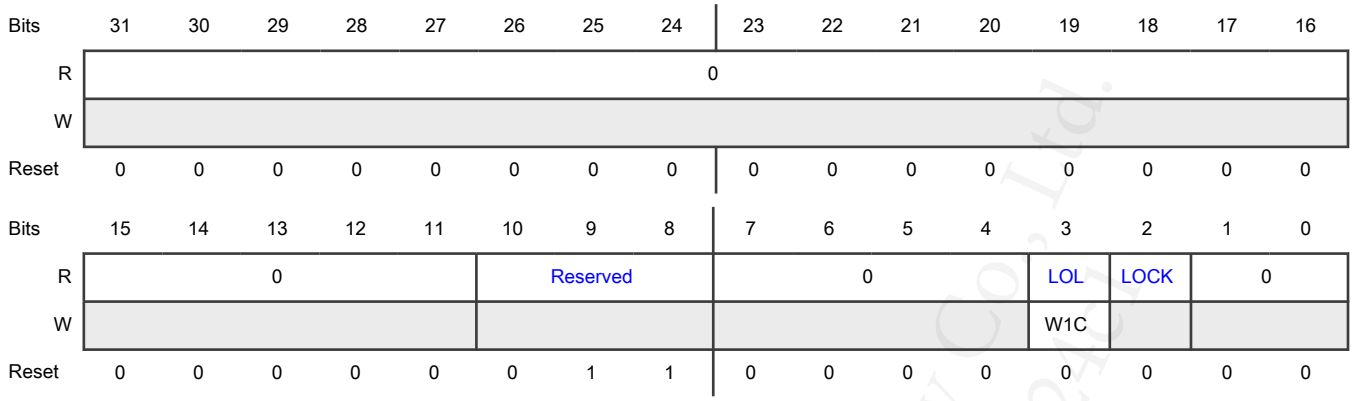
Register	Offset
PLLSR	4h

Function

Shows the PLL status.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-11 —	Reserved
10-8 —	Reserved
7-4 —	Reserved
3 LOL	Loss-Of-Lock flag Indicates the current PLL lock status. 0b - No loss of lock detected 1b - Loss of lock detected
2 LOCK	Lock status Indicates that PLL has acquired lock. 0b - PLL is unlocked 1b - PLL is locked
1-0 —	Reserved

26.4.4 PLL Divider (PLLDV)

Offset

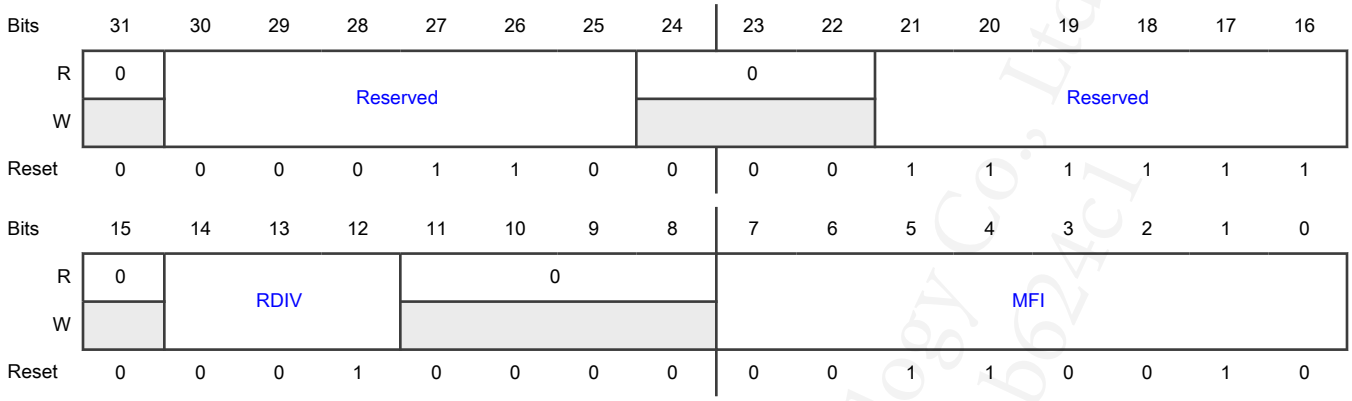
Register	Offset
PLLDV	8h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab62

Function

Divides input clocks for PLL output generation.

Diagram



Fields

Field	Function
31 —	Reserved
30-25 —	Reserved
24-22 —	Reserved
21-16 —	Reserved
15 —	Reserved
14-12 RDIV	Input clock predivider Sets the input clock divider. The output of the predivider circuit generates the PLL loop reference clock. 000b - Divide by 1 001b - Divide by 1 010b - Divide by 2 011b - Divide by 3 100b - Divide by 4 101b - Divide by 5 110b - Divide by 6

Table continues on the next page...

Table continued from the previous page...

Field	Function
	111b - Divide by 7
11-8 —	Reserved
7-0 MFI	Integer portion of loop divider. Sets the value of the divider in the PLL feedback loop, and the value specified establishes the multiplication factor applied to the reference frequency. Divider value = MFI, where the chosen MFI does not violate VCO frequency specifications.

26.4.5 PLL Frequency Modulation (PLLFM)

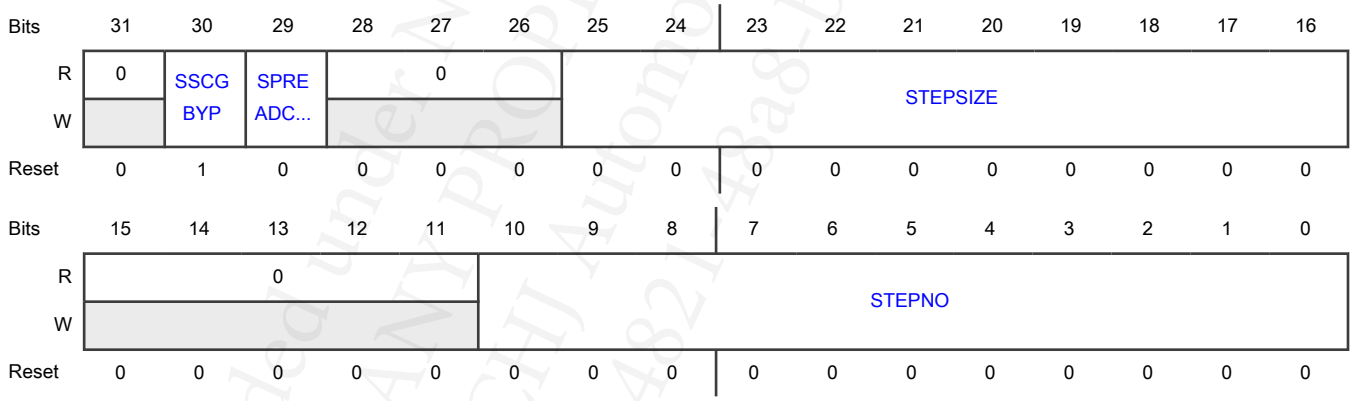
Offset

Register	Offset
PLLFM	Ch

Function

Configures PLL frequency modulation parameters.

Diagram



Fields

Field	Function
31 —	Reserved
30	Frequency modulation (spread spectrum clock generation) bypass

Table continues on the next page...

Table continued from the previous page...

Field	Function
SSCGBYP	Allows frequency modulation bypassing. 0b - Not bypassed 1b - Bypassed
29 SPREADCTL	Modulation type selection States that the modulation is centered around the nominal frequency. 0b - Modulation centered around nominal frequency 1b - Reserved
28-26 —	Reserved
25-16 STEPSIZE	Frequency modulation step size In Frequency Modulation mode, STEPSIZE provides the step size for modulation depth and frequency (see Frequency modulation).
15-11 —	Reserved
10-0 STEPNO	Number of steps of modulation period/frequency modulation In Frequency Modulation mode, STEPNO provides the number of steps to achieve modulation depth (see Frequency modulation).

26.4.6 PLL Fractional Divider (PLLFD)

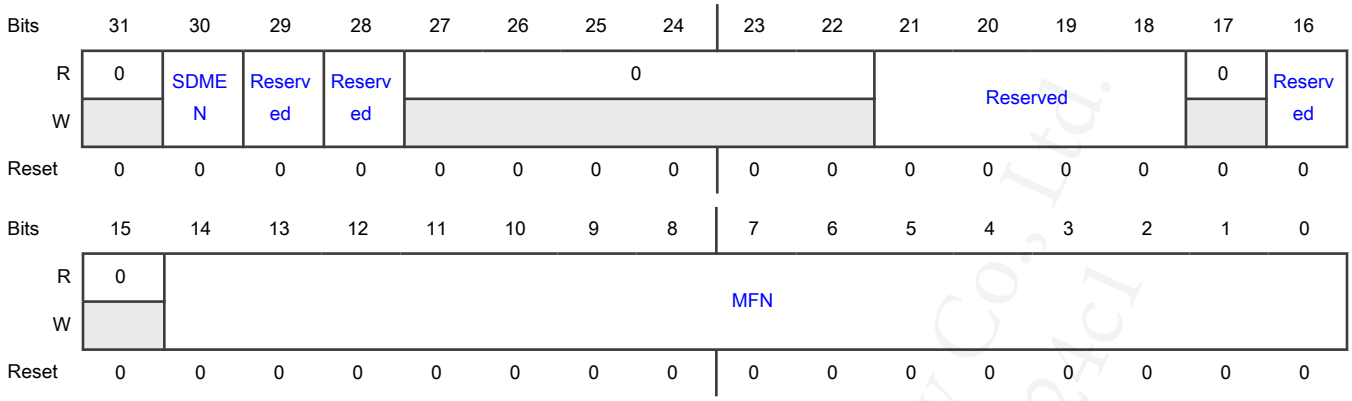
Offset

Register	Offset
PLLFD	10h

Function
Enables and configures frequency modulation.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-3901ab624c1

Diagram



Fields

Field	Function
31 —	Reserved
30 SDMEN	Fractional mode enable Enables Fractional mode. 0b - Disabled 1b - Enabled
29 —	Reserved
28 —	Reserved
27-22 —	Reserved
21-18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-0 MFN	Numerator of fractional loop division factor The value of this field must be less than 18432. In integer mode, it must be set to 15'd0.

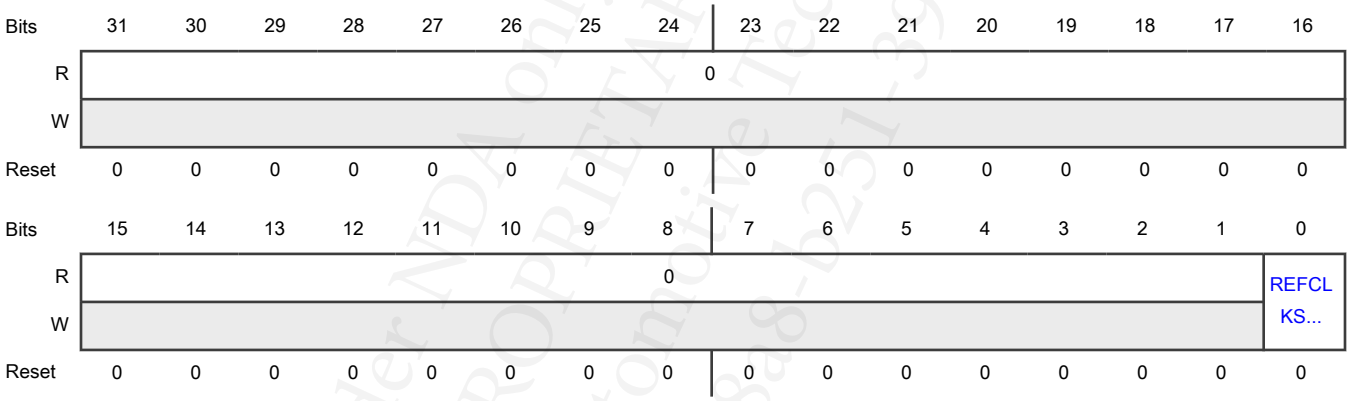
26.4.7 PLL Clock Multiplexer (PLLCLKMUX)

Offset

Register	Offset
PLLCLKMUX	20h

Function
Selects the PLL clock source.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 REFCLKSEL	Reference clock select Selects the PLL clock source. 0b - FIRC_CLK is source 1b - FXOSC_CLK is source

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-590d3ab624c1

26.4.8 PLL Output Divider (PLLODIV_0 - PLLODIV_1)

Offset

Register	Offset
PLLODIV_0	80h
PLLODIV_1	84h

Function

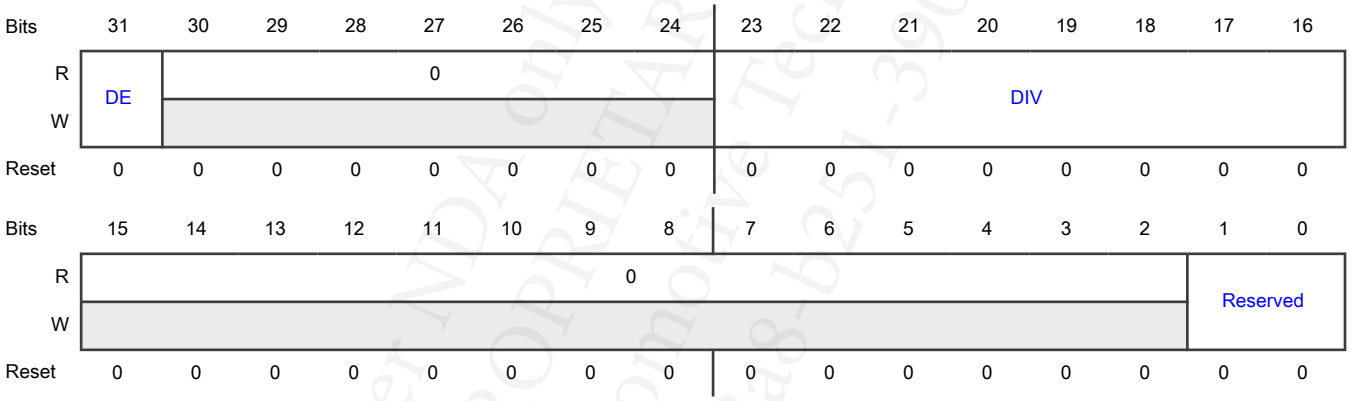
These registers control the PLL output clock divider settings. This divider is 50% duty cycle divider.

NOTE

These registers support only word accesses. Other write accesses lead to the following:

- Unpredictable behavior
- No transfer error getting generated

Diagram



Fields

Field	Function
31 DE	Divider enable Enables PLL output divider. Divider must be disabled before disabling PLL. 0b - Disabled 1b - Enabled
30-24 —	Reserved
23-16 DIV	Division value Provides the division value for the output clock divider. The clock period of the clock after division is DIV + 1 times the time period of the divider input clock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-2 —	Reserved
1-0 —	Reserved Do not write any value other than the reset value.

26.5 DDR_PLL register descriptions

This section provides the memory map and detailed descriptions of registers used for configuring the PLL. The table below shows the memory map. Addresses are given as offsets from the module base address. All registers are accessed using 8-bit, 16-bit, or 32-bit addressing.

26.5.1 DDR_PLL memory map

DDR_PLL base address: 4004_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PLL Control Register (PLLCR)	32	RW	8000_0000h
4h	PLL Status Register (PLLSR)	32	W1C	0000_0300h
8h	PLL Divider (PLLDV)	32	RW	0C3F_1032h
Ch	PLL Frequency Modulation (PLLFM)	32	RW	4000_0000h
10h	PLL Fractional Divider (PLLFD)	32	RW	0000_0000h
20h	PLL Clock Multiplexer (PLLCLKMUX)	32	RW	0000_0000h
80h	PLL Output Divider (PLLODIV_0)	32	RW	0000_0000h

26.5.2 PLL Control Register (PLLCR)

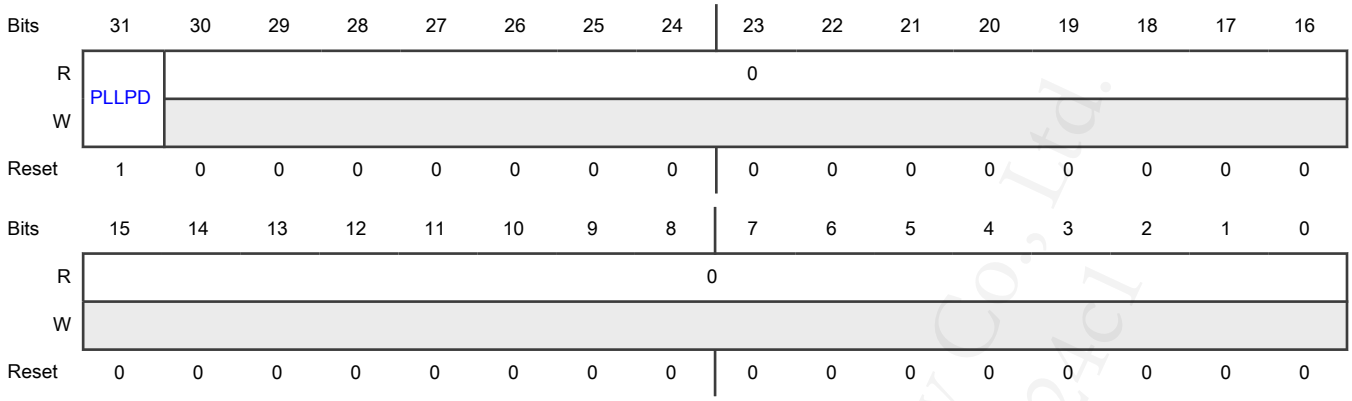
Offset

Register	Offset
PLLCR	0h

Function
Configures PLL functionality.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab9b4c1

Diagram



Fields

Field	Function
31 PLLPD	PLL power down Powers down PLL. 0b - PLL is powered up 1b - PLL is powered down
30-0 —	Reserved

26.5.3 PLL Status Register (PLLSR)

Offset

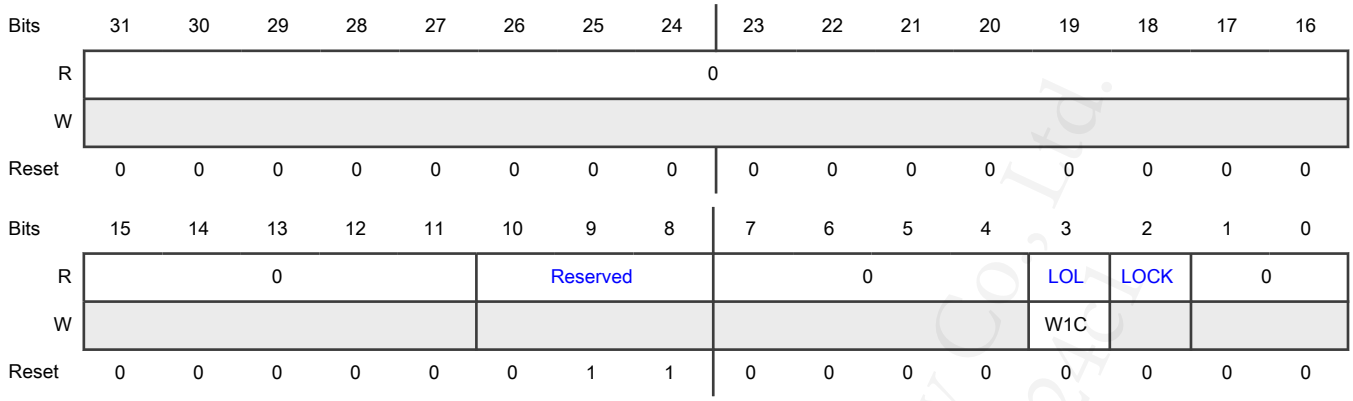
Register	Offset
PLLSR	4h

Function

Shows the PLL status.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-11 —	Reserved
10-8 —	Reserved
7-4 —	Reserved
3 LOL	Loss-Of-Lock flag Indicates the current PLL lock status. 0b - No loss of lock detected 1b - Loss of lock detected
2 LOCK	Lock status Indicates that PLL has acquired lock. 0b - PLL is unlocked 1b - PLL is locked
1-0 —	Reserved

26.5.4 PLL Divider (PLLDV)

Offset

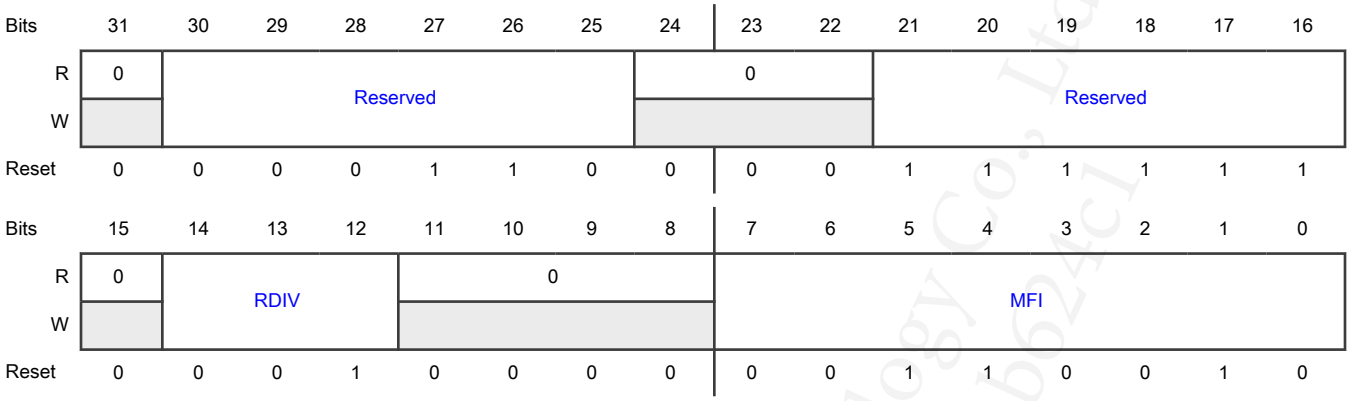
Register	Offset
PLLDV	8h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8ab62

Function

Divides input clocks for PLL output generation.

Diagram



Fields

Field	Function
31 —	Reserved
30-25 —	Reserved
24-22 —	Reserved
21-16 —	Reserved
15 —	Reserved
14-12 RDIV	Input clock predivider Sets the input clock divider. The output of the predivider circuit generates the PLL loop reference clock. 000b - Divide by 1 001b - Divide by 1 010b - Divide by 2 011b - Divide by 3 100b - Divide by 4 101b - Divide by 5 110b - Divide by 6

Table continues on the next page...

Table continued from the previous page...

Field	Function
	111b - Divide by 7
11-8 —	Reserved
7-0 MFI	Integer portion of loop divider. Sets the value of the divider in the PLL feedback loop, and the value specified establishes the multiplication factor applied to the reference frequency. Divider value = MFI, where the chosen MFI does not violate VCO frequency specifications.

26.5.5 PLL Frequency Modulation (PLLFM)

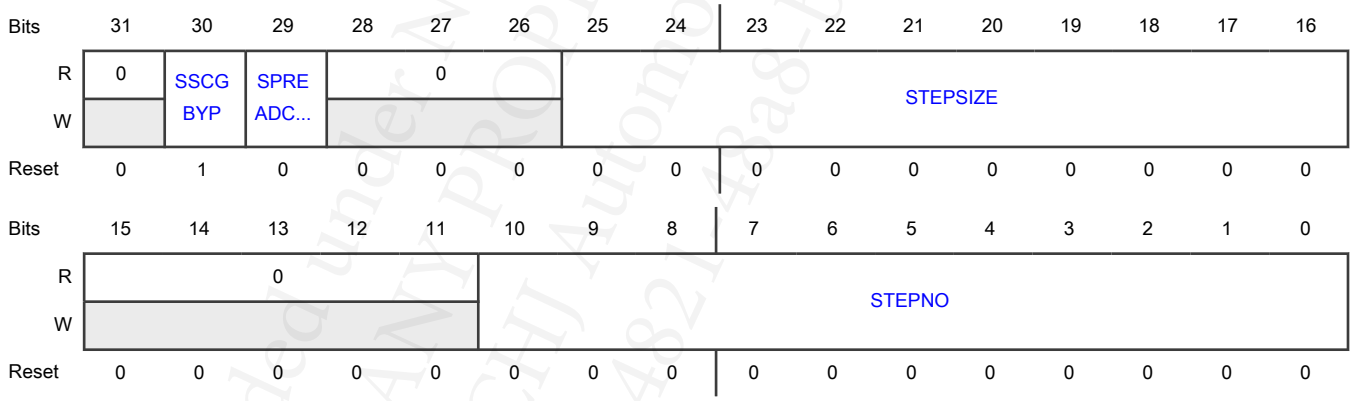
Offset

Register	Offset
PLLFM	Ch

Function

Configures PLL frequency modulation parameters.

Diagram



Fields

Field	Function
31 —	Reserved
30	Frequency modulation (spread spectrum clock generation) bypass

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a8-b251-390d3ab6941

Table continued from the previous page...

Field	Function
SSCGBYP	Allows frequency modulation bypassing. 0b - Not bypassed 1b - Bypassed
29 SPREADCTL	Modulation type selection States that the modulation is centered around the nominal frequency. 0b - Modulation centered around nominal frequency 1b - Reserved
28-26 —	Reserved
25-16 STEPSIZE	Frequency modulation step size In Frequency Modulation mode, STEPSIZE provides the step size for modulation depth and frequency (see Frequency modulation).
15-11 —	Reserved
10-0 STEPNO	Number of steps of modulation period/frequency modulation In Frequency Modulation mode, STEPNO provides the number of steps to achieve modulation depth (see Frequency modulation).

26.5.6 PLL Fractional Divider (PLLFD)

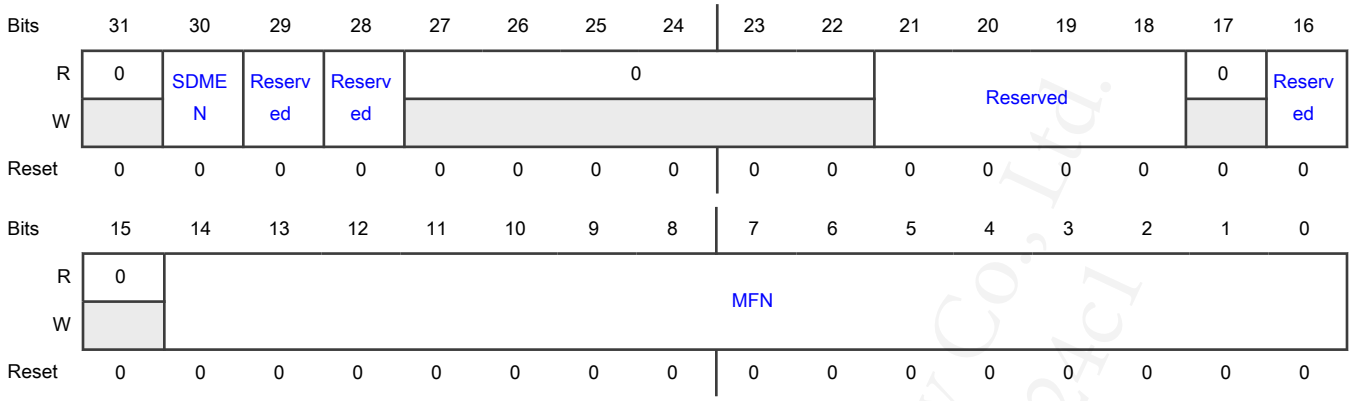
Offset

Register	Offset
PLLFD	10h

Function
Enables and configures frequency modulation.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-3901ab624c1

Diagram



Fields

Field	Function
31 —	Reserved
30 SDMEN	Fractional mode enable Enables Fractional mode. 0b - Disabled 1b - Enabled
29 —	Reserved
28 —	Reserved
27-22 —	Reserved
21-18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-0 MFN	Numerator of fractional loop division factor The value of this field must be less than 18432. In integer mode, it must be set to 15'd0.

26.5.7 PLL Clock Multiplexer (PLLCLKMUX)

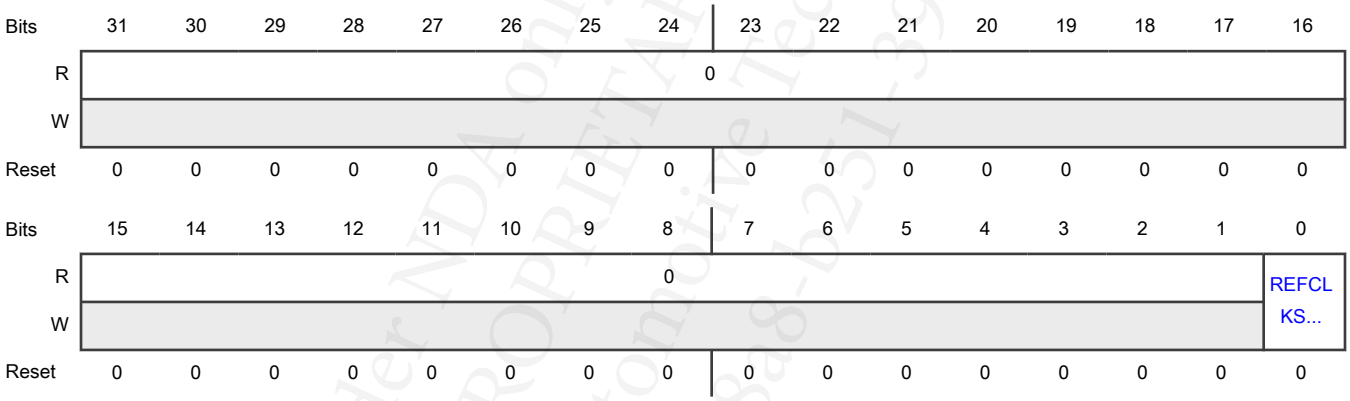
Offset

Register	Offset
PLLCLKMUX	20h

Function

Selects the PLL clock source.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 REFCLKSEL	Reference clock select Selects the PLL clock source. 0b - FIRC_CLK is source 1b - FXOSC_CLK is source

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-590d3ab624c1

26.5.8 PLL Output Divider (PLLODIV_0)

Offset

Register	Offset
PLLODIV_0	80h

Function

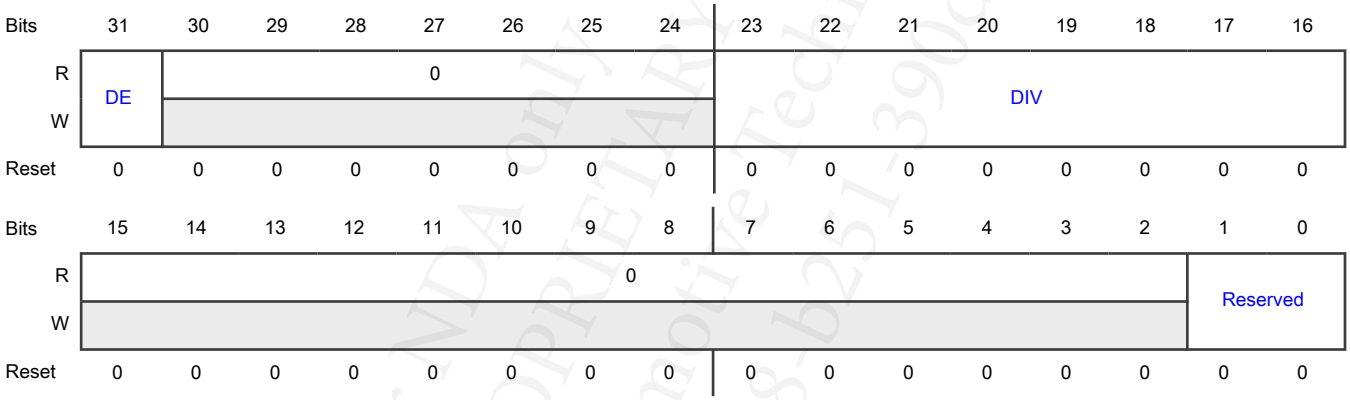
These registers control the PLL output clock divider settings. This divider is 50% duty cycle divider.

NOTE

These registers support only word accesses. Other write accesses lead to the following:

- Unpredictable behavior
- No transfer error getting generated

Diagram



Fields

Field	Function
31 DE	Divider enable Enables PLL output divider. Divider must be disabled before disabling PLL. 0b - Disabled 1b - Enabled
30-24 —	Reserved
23-16 DIV	Division value Provides the division value for the output clock divider. The clock period of the clock after division is DIV + 1 times the time period of the divider input clock.
15-2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1-0 —	Reserved Do not write any value other than the reset value.

26.6 PERIPH_PLL register descriptions

This section provides the memory map and detailed descriptions of registers used for configuring the PLL. The table below shows the memory map. Addresses are given as offsets from the module base address. All registers are accessed using 8-bit, 16-bit, or 32-bit addressing.

26.6.1 PERIPH_PLL memory map

PERIPH_PLL base address: 4003_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PLL Control Register (PLLCR)	32	RW	8000_0000h
4h	PLL Status Register (PLLSR)	32	W1C	0000_0300h
8h	PLL Divider (PLLDV)	32	RW	0C3F_1032h
10h	PLL Fractional Divider (PLLFD)	32	RW	0000_0000h
20h	PLL Clock Multiplexer (PLLCLKMUX)	32	RW	0000_0000h
80h - 9Ch	PLL Output Divider (PLLODIV_0 - PLLODIV_7)	32	RW	0000_0000h

26.6.2 PLL Control Register (PLLCR)

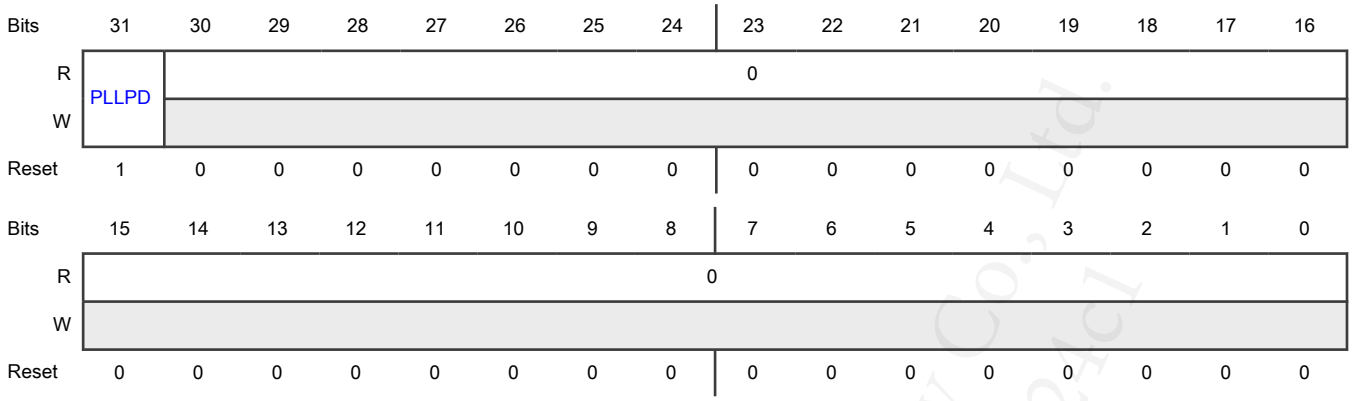
Offset

Register	Offset
PLLCR	0h

Function
Configures PLL functionality.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-3003d3ab624c1

Diagram



Fields

Field	Function
31 PLLPD	PLL power down Powers down PLL. 0b - PLL is powered up 1b - PLL is powered down
30-0 —	Reserved

26.6.3 PLL Status Register (PLLSR)

Offset

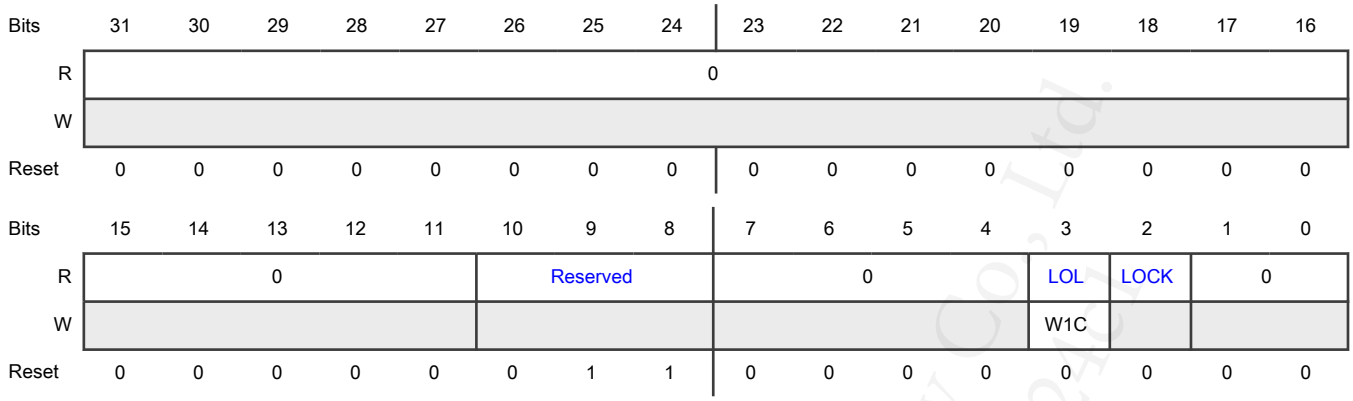
Register	Offset
PLLSR	4h

Function

Shows the PLL status.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-11 —	Reserved
10-8 —	Reserved
7-4 —	Reserved
3 LOL	Loss-Of-Lock flag Indicates the current PLL lock status. 0b - No loss of lock detected 1b - Loss of lock detected
2 LOCK	Lock status Indicates that PLL has acquired lock. 0b - PLL is unlocked 1b - PLL is locked
1-0 —	Reserved

26.6.4 PLL Divider (PLLDV)

Offset

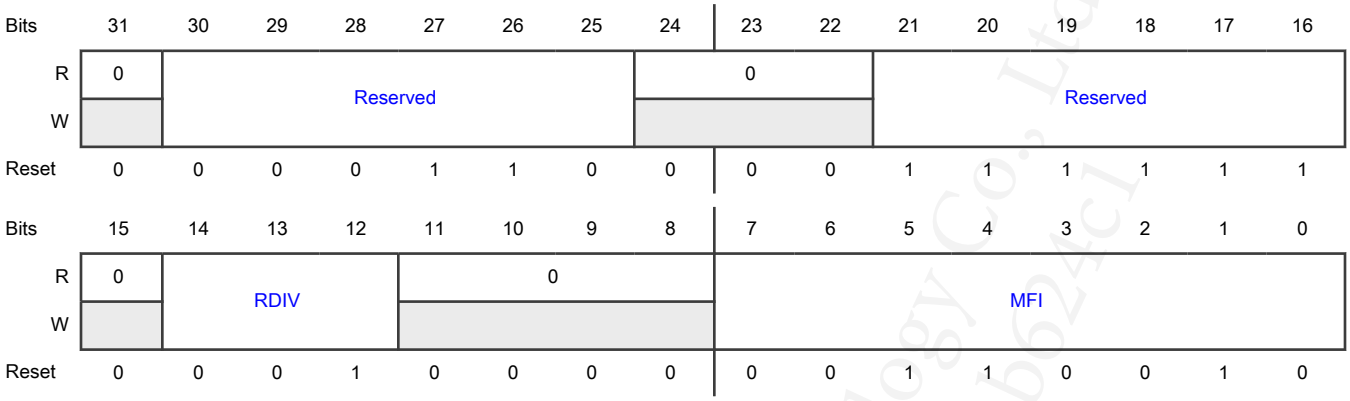
Register	Offset
PLLDV	8h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8ab62

Function

Divides input clocks for PLL output generation.

Diagram



Fields

Field	Function
31 —	Reserved
30-25 —	Reserved
24-22 —	Reserved
21-16 —	Reserved
15 —	Reserved
14-12 RDIV	Input clock predivider Sets the input clock divider. The output of the predivider circuit generates the PLL loop reference clock. 000b - Divide by 1 001b - Divide by 1 010b - Divide by 2 011b - Divide by 3 100b - Divide by 4 101b - Divide by 5 110b - Divide by 6

Table continues on the next page...

Table continued from the previous page...

Field	Function
	111b - Divide by 7
11-8 —	Reserved
7-0 MFI	Integer portion of loop divider. Sets the value of the divider in the PLL feedback loop, and the value specified establishes the multiplication factor applied to the reference frequency. Divider value = MFI, where the chosen MFI does not violate VCO frequency specifications.

26.6.5 PLL Fractional Divider (PLLFD)

Offset

Register	Offset
PLLFD	10h

Function

Enables and configures frequency modulation.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	SDMEN	Reserved	Reserved	0						Reserved				0	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	MFN														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30	Fractional mode enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
SDMEN	Enables Fractional mode. 0b - Disabled 1b - Enabled
29 —	Reserved
28 —	Reserved
27-22 —	Reserved
21-18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14-0 MFN	Numerator of fractional loop division factor The value of this field must be less than 18432. In integer mode, it must be set to 15'd0.

26.6.6 PLL Clock Multiplexer (PLLCLKMUX)

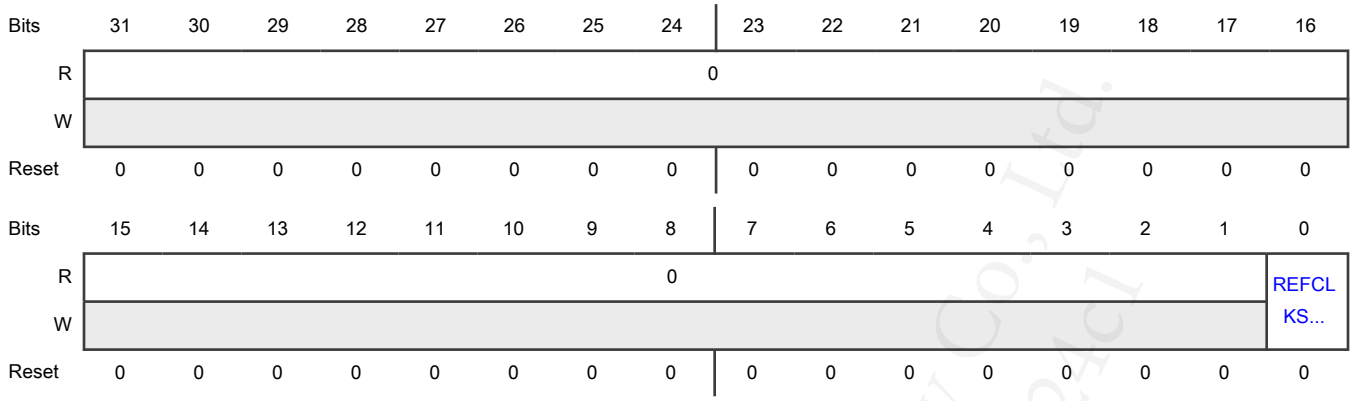
Offset

Register	Offset
PLLCLKMUX	20h

Function
Selects the PLL clock source.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4ca8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved
0 REFCLKSEL	Reference clock select Selects the PLL clock source. 0b - FIRC_CLK is source 1b - FXOSC_CLK is source

26.6.7 PLL Output Divider (PLLODIV_0 - PLLODIV_7)

Offset

For a = 0 to 7:

Register	Offset
PLLODIV_a	80h + (a × 4h)

Function

These registers control the PLL output clock divider settings. This divider is 50% duty cycle divider.

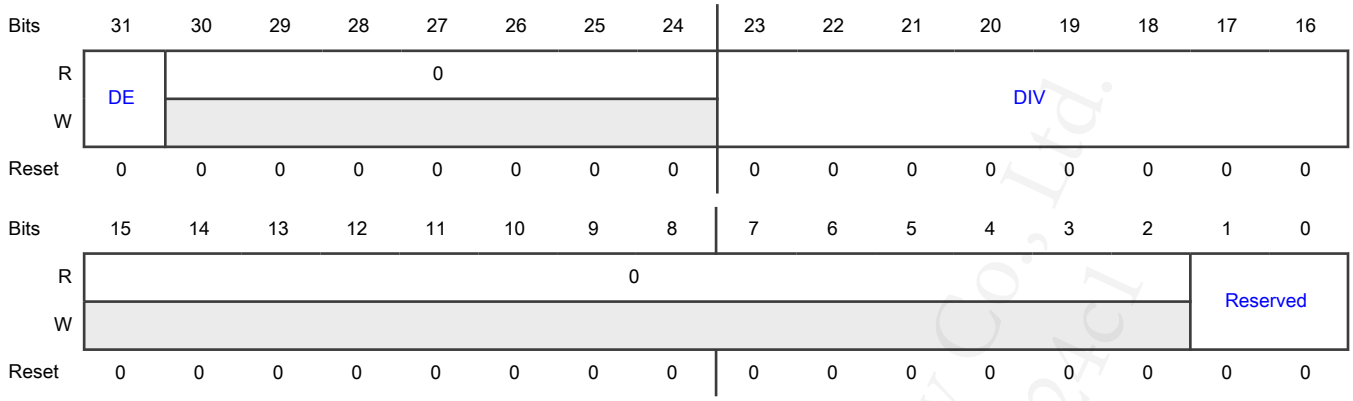
NOTE

These registers support only word accesses. Other write accesses lead to the following:

- Unpredictable behavior
- No transfer error getting generated

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e265-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 DE	Divider enable Enables PLL output divider. Divider must be disabled before disabling PLL. 0b - Disabled 1b - Enabled
30-24 —	Reserved
23-16 DIV	Division value Provides the division value for the output clock divider. The clock period of the clock after division is DIV + 1 times the time period of the divider input clock.
15-2 —	Reserved
1-0 —	Reserved Do not write any value other than the reset value.

26.7 Functional description

This section explains the operation and configuration of the PLL module.

26.7.1 Input clock frequency

The PLL is designed to operate over a specified input clock frequency range. PLL source frequency limits are discussed in this chip's data sheet.

26.7.2 Clock configuration

See the equations below and the corresponding register configuration that determine the relationship between VCO frequency (f_{VCO}) and PLL reference frequency.

- Integer-only mode:

— When PLLDV[RDIV] is 0:

$$f_{pll\_VCO} = f pll\_ref \times PLLDV[MFI]$$

Figure 109. PLL VCO frequency in integer-only mode when PLLDV[RDIV] is 0

— When PLLDV[RDIV] is not 0:

$$f_{pll\_VCO} = \frac{f_{pll\_ref}}{PLLDV[RDIV]} \times PLLDV[MFI]$$

Figure 110. PLL VCO frequency in integer-only mode when PLLDV[RDIV] is not 0

- Fractional mode:

— When PLLDV[RDIV] is 0:

$$f_{pll\_VCO} = f_{pll\_ref} \times \left(PLLDV[MFI] + \frac{PLLFD[MFN]}{18432}\right)$$

Figure 111. PLL VCO frequency in Fractional mode when PLLDV[RDIV] is 0

— When PLLDV[RDIV] is not 0:

$$f_{pll\_VCO} = \frac{f_{pll\_ref}}{PLLDV[RDIV]} \times \left(PLLDV[MFI] + \frac{PLLFD[MFN]}{18432}\right)$$

Figure 112. PLL VCO frequency in Fractional mode when PLLDV[RDIV] is not 0

See the equation below and the corresponding register configuration that determine the relationship between reference and PLL_PHI*n* output frequencies.

$$f_{pll\_phi} = \frac{f_{pll\_VCO}}{PLLODV\_n[DIV+1]}$$

Figure 113. PLL PHI output frequency

When configuring the PLL, user software must not violate the maximum system clock frequency or max/min VCO frequency specification of the PLL (see this chip's data sheet for frequency limits).

NOTE

The PLL must be disabled by writing PLLCR[PLLPD] = 1 before any PLL configuration or input clock are modified.

NOTE

The PLL must be disabled by writing PLLCR[PLLPD] = 1 for at least 5 µs before enabling the PLL (writing PLLCR[PLLPD] = 0).

The recommended procedure to program the PLL and engage normal mode is shown in Initialization information.

26.7.3 LOL

PLL provides LOL indication. The LOL indication can only be generated when the PLL is in functional mode (see Modes of operation). When the PLL detects a LOL, it asserts its LOL event output.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c62e3-42f4-48a8-25...

NOTE

- The PLL does not detect loss of reference clock. If the reference clock stops after the PLL achieves lock, PLL continues to indicate lock. It is assumed that monitoring of the PLL's reference clock that is done outside the PLL is enabled while the PLL is in operation.
- PLL LOL is intended for detection of gross failures. Use CMUs for accurate frequency monitoring.

26.7.4 Frequency modulation

In Frequency Modulation mode, the PLL generates a frequency-modulated clock. The modulation depth and modulation frequency are calculated using the equations shown in Frequency modulation programming.

The field PLLFM[SPREADCTL] must set to 0b to select center-spread modulation. See Figure 114 that shows an example of center-spread modulation.

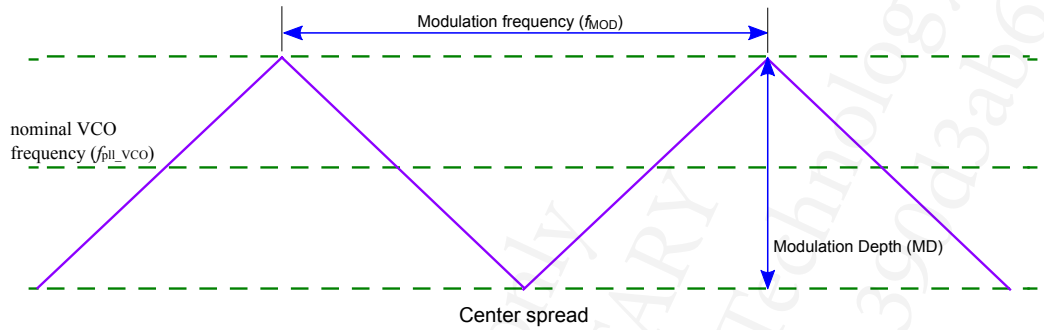


Figure 114. Frequency modulation

26.7.4.1 Frequency modulation programming

Modulation depth and modulation frequency programming uses step number (PLLFM[STEPNO]) and step size (PLLFM[STEPSIZE]). The table below shows variables used during calculations when programming the PLL for frequency modulation.

Table 110. Variables for configuring modulation depth and frequency

Variable	Description
f_{REF}	Input clock frequency
f_{MOD}	Expected modulation frequency
MD	Expected modulation depth in percentage
LDF	Loop division factor

Use the following equations to configure the PLL for frequency modulation.

$$LDF = PLLDV[MFI] + \frac{PLLFD[MFN]}{18432}$$

Figure 115. LDF

$$\text{PLLFM[STEPNO]} = \frac{f_{\text{REF}}}{\left(2 \times f_{\text{MOD}} \times \text{PLLDV[RDIV]}\right)}$$

Figure 116. Step number

$$\text{PLLFM[STEPSIZE]} = \frac{\text{MD} \times \text{LDF}}{100 \times \text{PLLFM[STEPNO]}} \times 18432$$

Figure 117. Step size

Frequency Modulation is only possible if the condition shown in Figure 118 is met.

$$(\text{PLLFM[STEPSIZE]} \times \text{PLLFM[STEPNO]}) < 18432$$

Figure 118. Requirement to achieve FM

NOTE

PLLFM[SSCGBYP] must be cleared and PLLFD[SDMEN] must be set for frequency modulation to be enabled.

CAUTION

The effective modulation depth may differ from the intended modulation depth because of rounding operations applied to PLLFM[STEPSIZE] and PLLFM[STEPNO].

26.8 Initialization information

Perform the following steps to complete the PLL initialization:

1. Confirm that PLLODIV_*n*[DE] = 0 for all dividers.
2. Confirm that PLLCR[PLLPD] = 1.
3. Program PLLCLKMUX to select appropriate PLL reference clock.
4. Program the following as needed:
   - PLLDV
   - PLLFD
   - PLLFM to the desired value.
5. Program field PLLODIV_*n*[DIV] to the desired value.
6. Wait for the PLL reference clock to be stable.
7. Write PLLCR[PLLPD] = 0.
8. Wait for field PLLSR[LOCK] = 1.
9. Write PLLODIV_*n*[DE] = 1.

Perform the following steps to complete the PLL shut-down:

1. Write PLLODIV_n[DE] = 0 for all dividers.
2. Write PLLCR[PLLPD] = 1.

Preliminary Information
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-4821-48a8-b251-390d3b624c1

Chapter 27
Fast Crystal Oscillator Digital Controller (FXOSC)

27.1 Introduction

27.1.1 Overview

This module provides the fast crystal oscillator to the chip.

27.1.2 Features

FXOSC features are as follows:

- Status register shows current module state.
- Control register can:
  - Select a mode of operation:
    - Crystal mode
    - Single-Input Bypass mode using EXTAL clock input
    - Differential Input Bypass mode using EXTAL and XTAL inputs
  - Disable the module (Power-Down mode)

27.2 FXOSC register descriptions

This section provides the descriptions of all registers used for configuring the FXOSC.

27.2.1 FXOSC memory map

Use 8-bit, 16-bit, or 32-bit addressing to access registers.

FXOSC base address: 4005_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	FXOSC Control Register (CTRL)	32	RW	019D_00C0h
4h	Oscillator Status Register (STAT)	32	RO	0000_0000h

27.2.2 FXOSC Control Register (CTRL)

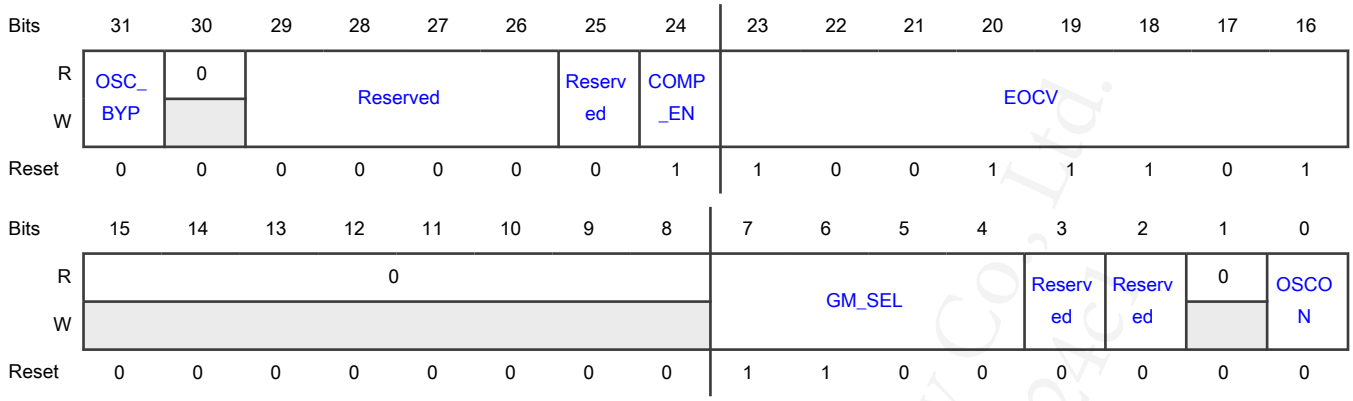
Offset

Register	Offset
CTRL	0h

Function
Configures FXOSC operation.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 OSC_BYP	Oscillator bypass Bypasses the internal oscillator. 0b - Internal oscillator not bypassed 1b - Internal oscillator bypassed
30 —	Reserved
29-26 —	Reserved
25 —	Reserved
24 COMP_EN	Comparator enable Enables or disables the comparator. • For Crystal mode or Differential Bypass mode set this field to 1. • For Single-Input Bypass mode set this field to 0. 0b - Comparator disabled 1b - Comparator enabled
23-16 EOCV	End of count value Specifies the end-of-count.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c62e5-4a21-48a8-b251-390d8ab62

Table continued from the previous page...

Field	Function
	The oscillator counter runs on the crystal clock divided by 4 and counts up to EOCV × 128. This counting period ensures that the external oscillator clock signal is stable before the system selects FXOSC as a source. **NOTE** • You must set EOCV to the appropriate value to allow clock and duty cycle to stabilize and guarantee that OSC_STAT becomes set within the crystal startup time. — In Crystal mode, EOCV value must to be calculated to appropriate value based on the crystal specification using the equation in Initializing FXOSC. — In Differential Bypass mode, EOCV value must be configured corresponding to at least 1 ms. — In Single-Input Bypass mode, EOCV value is irrelevant. FXOSC holds the counter in reset. • Before modifying EOCV, FXOSC must be disabled.
15-8 —	Reserved
7-4 GM_SEL	Crystal overdrive protection Selects the transconductance applied by the FXOSC amplifier. This setting depends on crystal specification. **NOTE** • In Crystal mode and Differential Bypass mode FXOSC will not function with zero transconductance (GM_SEL = 0000b). • For details on how to set this field, see Initializing FXOSC. 0000b - 0× 0001b - 0.145× 0010b - 0.223× 0011b - 0.368× 0100b - 0.291× 0101b - 0.436× 0110b - 0.514× 0111b - 0.660× 1000b - 0.339× 1001b - 0.485× 1010b - 0.563× 1011b - 0.708×

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1100b - 0.631× 1101b - 0.776× 1110b - 0.854× 1111b - 1.0×
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 OSCON	Crystal oscillator power-down control Enables or disables FXOSC 0b - Disables FXOSC 1b - Enables FXOSC

27.2.3 Oscillator Status Register (STAT)

Offset

Register	Offset
STAT	4h

Function

Shows current state of FXOSC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OSC_STAT	0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462654-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 OSC_STAT	Crystal oscillator status Indicates the crystal oscillator status. **NOTE** OSC_STAT value is is not valid if transconductance is set to 0. 0b - Crystal oscillator is off or on but not stable. 1b - Crystal oscillator is on and providing a stable clock.
30-0 —	Reserved

27.3 Functional description

The table below shows configuration settings for the different FXOSC modes set by the CTRL register:

Table 111. FXOSC operation mode settings

Mode	CTRL[OSCON]	CTRL[OSC_BYP]	CTRL[COMP_EN]	FXOSC_CLK
Power-Down mode	0	X	X	0
Crystal mode	1	0	1	Crystal clock
Single-Input Bypass mode	1	1	0	EXTAL
Differential Bypass mode	1	0	1	EXTAL and XTAL

Power-Down mode is the FXOSC default condition after any reset: POR, Destructive, or Functional.

27.3.1 Clock generation in Crystal mode

Counter logic and FXOSC_CLK start when counter value reaches EOCV × 128.

27.3.2 Clock generation in Single-Input Bypass mode

This mode bypasses the oscillator and uses a single-input external clock (EXTAL input) for FXOSC_CLK.

27.3.3 Clock generation in Differential Input Bypass mode

This mode bypasses the oscillator and uses differential-input external clocks (EXTAL and XTAL inputs) for FXOSC_CLK.

27.4 Initialization information

Initializing FXOSC

Initialize FXOSC as follows:

1. Write the desired value to CTRL[OSC_BYP] and CTRL[COMP_EN] to select an operation mode as shown in Table 111.

NOTE

FXOSC must be disabled when the operation mode is modified.

2. Configure CTRL[GM_SEL].
   - In Crystal mode configure the transconductance based on the module specification in the chip data sheet.
   - In Single-Input Bypass mode write 0000b to this field.
   - In Differential Bypass mode write 0001b to this field.

   NOTE

   In Crystal mode and Differential Bypass mode FXOSC will not function with zero transconductance (GM_SEL = 0000b).
3. Set CRTL[EOCV] calculating the value as follows:
   - EOCV (in decimal) = (stabilization time in ns) ÷ (4 × 128 × (period of clock in ns))
4. Write 1 to CRTL[OSCON] to enable FXOSC.
5. Confirm the clock is stable (STAT[OSC_STAT] = 1) before using it.

Disabling FXOSC

Write 0 to CRTL[OSCON] to disable FXOSC when FXOSC_CLK is running and stable.

FXOSC enters Power-down mode after at least four crystal clocks. No glitches occur during the transition to Power-Down mode because synchronizers are used.

NOTE

After disabling FXOSC:

- Wait for at least 2 µs before enabling FXOSC again.
- You must not change other values in FXOSC registers for at least 16 FXOSC_CLK cycles.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 28
Reset

28.1 Introduction

This chapter describes the reset sequence and behavior of the chip.

The reset sequence consists of defined entry points with a fixed sequence that leads to a known and deterministic state of the chip after completion. One or more modules on the chip triggers a reset event, and these reset events determine the entry points to a reset sequence.

This chapter discusses the following details:

- Global reset behavior of the chip
- Various reset sources
- Possible reset sequences and their stages
- Software resettable domains that can be reset independently without affecting the functionality of other modules
- Status of modules during the reset process
- Reset behavior of the chip during low-power (Standby) mode entry and exit

The figure below illustrates the high-level reset architecture that involves various module interactions with the Reset Generation Module (MC_RGM) during different stages of the reset sequence.

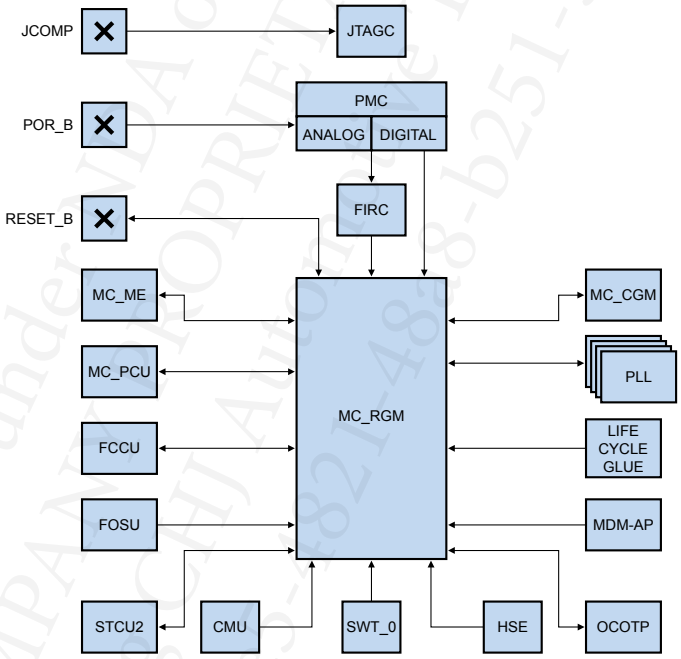


Figure 119. Reset top-level diagram

NOTE

- Bidirectional arrow: The RGM handshakes with these modules during the reset sequence.
- Unidirectional arrow: These modules are sources of reset.

The following table shows the types of reset events that can lead to their respective reset sequences.

Table 112. Description of reset events

Reset event	Description
Functional reset	Chip operation is still reliable and SRAM memory content can be preserved. Leads to a reset of modules for which functionality is not guaranteed.
Destructive reset	Resets most of the chip except for a few selected modules. Memory content is considered to be lost after this reset event.
Power-on reset (POR)	The entire chip is reset.

28.2 Reset features

The reset architecture of this chip provides the following features:

- Reset sequence controls with three entry points:
  - POR
  - Destructive reset
  - Functional reset
- Status reporting of destructive and functional reset events
- Functional reset escalation to destructive reset
- Destructive reset escalation to stay in reset until next POR
- SRAM content protection against corruption on any functional reset event
- Software resettable domains management
- Reset management after chip exits Standby mode
- Chip configuration through fuses

28.3 High-level reset sequence

The following diagram shows the high-level reset sequence overview for POR, destructive, and functional reset events. When the reset process has reached the 'Out of Reset' state, the chip is operational.

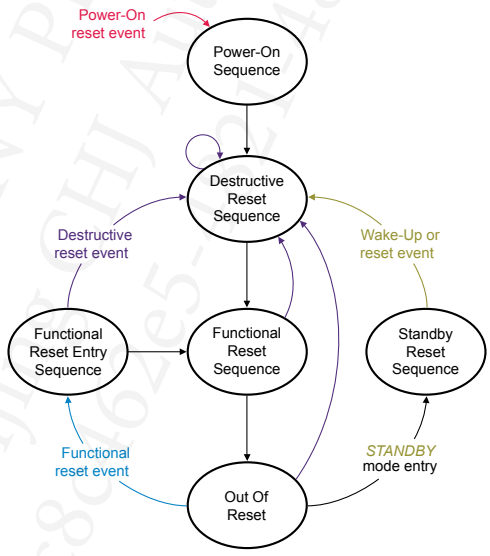


Figure 120. High-level reset sequence flow

A POR event causes an immediate entry into the beginning of the Power-On Sequence state regardless of whether the chip is in one of the reset sequences or the Out Of Reset state. A single red arrow represents this in order to simplify the above diagram.

The following table shows the entry points into the high-level reset sequence for each type of reset event.

Table 113. Reset event reactions in reset sequence

Reset event type	From	Reaction
POR	Anywhere	Move to beginning of power-on sequence
Destructive	Power-on sequence	No reset sequence change
Destructive	Anywhere else	Move to beginning of destructive reset sequence
Functional	Out of reset	Move to beginning of functional reset entry sequence
Functional	Standby reset sequence before VDD_CORE_OK is asserted or FIRC is stable	Move to beginning of destructive reset sequence
	Anywhere else	No reset sequence change

The chip does the following for each reset event captured by MC_RGM:

- Writes a 1 to the corresponding reset event status field in MC_RGM_DES and MC_RGM_FES registers.
- Places its pins into their default states.
- Enters the reset sequence as described in the table above.

28.4 Reset sources

All POR sources always assert RESET_B when active. The POR cannot be demoted to an IRQ. No escalation is needed.

All destructive reset sources always assert RESET_B when active.

If fields in the MC_RGM_DES and MC_RGM_FES registers are both 1 after a reset sequence, it means that the chip experienced a destructive reset when the functional reset source was active. In this case, ignore F_DR_8, F_DR_13, and F_DR_18 in the MC_RGM_DES register, because these are spuriously set.

Table 114. Reset sources

Name	Description	Source module	Index	RESET_B assertion	Can be demoted to IRQ	Escalation
POR reset sources						
POR	Core supply	PMC	D[0]	Always	No	No
POR	PMC supply	PMC	D[0]	Always	No	No
POR	Critical SPD error	Critical SPDs (CSPD)	D[0]	Always	No	No
POR	POR_B pin assertion	POR_B pin	D[0]	Always	No	No
POR	POR watchdog timeout	POR_WDG (in PMC)	D[0]	Always	No	No
Destructive reset sources						

Table continues on the next page...

Table 114. Reset sources (continued)

Name	Description	Source module	Index	RESET_B assertion	Can be demoted to IRQ	Escalation
NC_SPD_RST	Non-critical SPD fail	Non-critical SPDs (NCSPD)	D[1]	Always	No	No
FCCU_FTR	FCCU failure to react	FOSU	D[3]	Always	No	Yes
STCU_URF	STCU unrecoverable fault	STCU	D[4]	Always	No	Yes
MC_RGM_FRE	Functional reset escalation	MC_RGM	D[6]	Always	No	Yes
FXOSC_FAIL	FXOSC failure	CMU_0	D[8]	Always	No	Yes
CORE_LOL	CORE_PLL loss of lock	CORE_PLL	D[9]	Always	No	Yes
PERIPH_LOL	PERIPH_PLL loss of lock	PERIPH_PLL	D[10]	Always	No	No
DDR_LOL	DDR_PLL loss of lock	DDR_PLL	D[11]	Always	No	No
ACC_LOL	ACCEL_PLL loss of lock	ACCELERATOR_PLL	D[12]	Always	No	No
XBAR_DIV3_CLK_FAIL	XBAR_DIV3_CLK failure	CMU_5	D[13]	Always	No	No
HSE_LC_RST	Life-cycle error	HSE life-cycle checker	D[16]	Always	No	Yes
HSE_SNVS_RST	HSE SNVS tamper detected	HSE SNVS	D[17]	Always	No	No
HSE_SWT_RST	HSE SWT timeout	HSE SWT	D[18]	Always	No	Yes
SW_DEST	Software destructive reset	MC_ME	D[30]	Always	No	Yes
DEBUG_DEST	Debug destructive reset	MDM_AP	D[31]	Always	No	No
Functional reset sources						
EXT[1]	RESET_B pin assertion	RESET_B pin	F[0]	Always	Yes	No
FCCU_RST	FCCU reset reaction	FCCU soft reaction	F[3]	Always	No	Yes
ST_DONE	Self-test done	STCU	F[4]	Always	No	No
SWT0_RST	SWT0 timeout	SWT0	F[6]	Always	No	Yes
HSE_RAM_ECC_RST	HSE_H memory ECC error	HSE_H ECC module	F[18]	Always	No	No
HSE_BOOT_ERR_RST	Boot failure error	HSE_H	F[20]	Always	No	Yes

Table continues on the next page...

Table 114. Reset sources (continued)

Name	Description	Source module	Index	RESET_B assertion	Can be demoted to IRQ	Escalation
HSE_CORE_LOCK_RST	HSE_H M7 core lock	HSE	F[21]	Always	No	Yes
SW_FUNC	Software functional reset	MC_ME	F[30]	Always	No	No
DEBUG_FUNC	Debug functional reset	MDM_AP	F[31]	Always	Yes	No

1. This reset source is intended to be used for DEBUG purpose only.

28.5 Reset sources descriptions

28.5.1 POR sources

28.5.1.1 On-chip POR

During chip power-up, the chip asserts its internal POR signal as soon as the on-chip POR detection circuitry is functional (see the "Power management" chapter for supplies monitored during POR).

During normal chip operation, the internal power-on reset signal asserts when any of the chip voltages monitored by the on-chip POR detection circuitry fall below their specified minimum thresholds. When the chip asserts the internal POR signal, the signal remains asserted until all the chip voltages monitored by the on-chip POR detection circuitry rise above their minimum functional thresholds.

The on-chip POR circuitry exists in the CSPDs attached to the following power rails (see the chapter "Power Management Controller (PMC)" for details):

- Logic required for the reset sequence and initial boot
- Analog portion of the FIRC
- Critical pins (for example, RESET_B)

28.5.1.2 Critical LVDs

During chip power-up, the chip asserts each critical LVD reset signal as soon as the signal's corresponding on-chip LVD detection circuits are functional.

During normal chip operation, the chip asserts each critical LVD reset signal as soon as the corresponding monitored voltage falls below its corresponding specified threshold.

When the chip asserts a critical LVD reset signal, the chip continues to assert the signal until the corresponding chip voltage rises above its threshold.

The monitored LVD chip voltages that cause reset assertion when they fall below their specified voltages are listed in the Power Management chapter, in the "Minimum off-chip LVDs and HVDs" table.

The critical on-chip LVD circuitry exists in CSPDs. They are connected to the power rails which are required to be at specified levels for the boot process to execute properly.

28.5.1.3 External POR_B pin

The active-low POR_B pin uses a weak pull-down on its input.

During chip power-up, off-chip power management circuitry asserts the POR_B pin as soon as it becomes functional. The off-chip power management circuit may consist of a single Power Management IC (PMIC) or a combination of chips.

During normal operation, the off-chip power management circuit asserts the POR_B pin when any of voltages monitored by this circuitry falls below its corresponding threshold.

When off-chip power management circuitry asserts the POR_B pin, this circuitry continues to assert POR_B until all of its monitored voltages rise above their corresponding thresholds.

Off-chip power management circuitry monitors all off-chip-supplied voltages.

28.5.1.4 POR watchdog timer

The chip asserts POR_WDG reset immediately when the POR watchdog timer expires.

The chip clears the POR watchdog timer and deasserts POR_WDG reset immediately after the chip enters a POR sequence.

The following reset events initiate the POR watchdog timer count:

1. All POR events (after completion of the power-on sequence)
2. All destructive reset events (immediately after event detection)
3. All functional reset events (immediately after event detection)

The default timeout value of POR_WDOG is 960 SIRC clock cycles (30 ms).

The following conditions mentioned must be true for the POR watchdog to function:

- Fuses do not disable the POR watchdog.
- Nothing triggers a destructive reset escalation threshold.

28.5.2 Destructive reset sources

28.5.2.1 SPD fail

SPDs detect disturbances in the connections between the associated regulator and the monitored power. Such a disturbance may be:

- A broken wire between the PMIC and the chip
- An intermittent connection
- A high-resistance connection

During chip operation, the chip asserts each SPD fail signal as soon as the corresponding chip voltage monitored by the SPD circuitry falls to a level below the precision LVD minimum voltage, but still above the level at which the logic needed to process the ensuing reset can function.

When the chip asserts an SPD fail signal, it continues to assert that signal until the monitored voltage rises above its threshold.

The reaction of NCSPDs is disabled on

- Power-on reset
- Destructive sequence entry
- Functional reset

The NCSPD_CTL register inside the PMC is reset upon functional reset. The default configuration of this register is such that reactions of all NCSPDs are disabled.

You can enable NCSPDs via software.

Software can determine which NCSPD has caused a destructive reset by reading a write-1-clear status register in the PMC.

You cannot disable CSPDs.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-482-48a8-b251-390d3ab624c1

28.5.2.2 FCCU failure to react

The FOSU module monitors the FCCU and asserts the FCCU failure-to-react reset if the FCCU fails to react to a fault within the specified amount of time.

The FCCU failure-to-react reset is deasserted immediately on destructive reset and POR sequence entries.

28.5.2.3 STCU2 unrecoverable fault

The STCU2 asserts an unrecoverable fault based on the STCU2's configuration of multiple possible error conditions.

The STCU2 unrecoverable fault reset is deasserted immediately on destructive reset and POR sequence entries.

28.5.2.4 Functional reset escalation to destructive reset

Functional reset escalation reset is asserted by MC_RGM immediately upon the functional reset event that causes the escalation counter to increment and reach the threshold value in the MC_RGM_FRET[FRET] register field.

The functional reset escalation counter is cleared and functional reset escalation reset is deasserted immediately on destructive reset and POR sequence entries.

The functional reset escalation feature can be enabled and disabled via software.

The functional reset escalation threshold value can be set via software.

The default configuration of the functional reset escalation feature immediately after destructive reset and POR sequence exit is enabled with the FRET value set to 15.

If enabled, the functional reset escalation counter increments by one with each functional reset event that occurs when the chip is out of reset.

The functional reset escalation counter can be cleared by software.

28.5.2.5 FXOSC failure

FXOSC failure reset is asserted by the clock monitor that is monitoring the FXOSC output clock as soon as the monitor detects that the FXOSC output clock's frequency has gone out of the specified minimum/maximum frequency threshold range.

FXOSC failure reset is deasserted immediately on destructive reset and POR sequence entries.

28.5.2.6 CORE_PLL loss of lock

CORE_PLL loss of lock is asserted by the CORE_PLL as soon as it loses its lock after having previously acquired it.

CORE_PLL loss of lock is deasserted immediately on any reset sequence entry.

28.5.2.7 PERIPH_PLL loss of lock

PERIPH_PLL loss of lock is asserted by the PERIPH_PLL as soon as it loses its lock after having previously acquired it.

PERIPH_PLL loss of lock is deasserted immediately on any reset sequence entry.

PERIPH_PLL is reset on functional reset. The PERIPH_PLL loss of lock generates destructive reset and therefore is reset on destructive reset sequence entry.

28.5.2.8 DDR_PLL loss of lock

DDR_PLL loss of lock is asserted by the DDR_PLL as soon as it loses its lock after having previously acquired it.

DDR_PLL loss of lock is deasserted immediately on any reset sequence entry.

DDR_PLL is reset on functional reset. The DDR_PLL loss of lock generates destructive reset and therefore is reset on destructive reset sequence entry.

28.5.2.9 ACCEL_PLL loss of lock

The ACCEL_PLL loss of lock is asserted by the ACCEL_PLL as soon as it loses its lock after having previously acquired it.

The ACCEL_PLL loss of lock is deasserted immediately on any reset sequence entry.

ACCEL_PLL gets reset on functional reset. The ACCEL_PLL loss of lock will generate the destructive reset and hence will get reset on destructive reset sequence entry.

28.5.2.10 XBAR_DIV3 clock failure

The minimum frequency threshold for the XBAR_DIV3_CLK failure monitor is typically set lower than for the precision monitoring done by the CMU that provides an FCCU reaction. This is because this monitor is checking primarily for a loss of clock in this clock domain.

XBAR_DIV3_CLK failure reset is deasserted immediately on destructive reset and POR sequence entries.

28.5.2.11 Life-cycle error

The security Life-cycle fault reset is asserted by the Life-cycle block immediately on the detection of an error in the Life-cycle value because of, for example:

- A failure during the reading of the Life-cycle value
- Reading an invalid Life-cycle value
- A corruption of the stored Life-cycle value

The security Life-cycle fault reset is deasserted immediately on destructive reset and POR sequence entries.

28.5.2.12 Software destructive reset

Software destructive reset is asserted immediately upon a mode change request to a destructive reset via the MC_ME.

Software destructive reset is cleared immediately on destructive reset and POR sequence entries.

28.5.2.13 Debug destructive reset

Debug destructive reset is asserted immediately when the debugger sets the destructive reset field in the DAP's MDM_AP register to 1.

Debug destructive reset is deasserted immediately when the debugger sets the destructive reset field in the DAP's MDM_AP register to 0 (the field is automatically set to 0 because the debugger logic is working on the destructive reset itself).

28.5.2.14 HSE_H SWT timeout

The HSE_H SWT timeout signal is asserted as soon as the HSE_H SWT has not been serviced by the software running on the HSE_H core before the SWT's reset timer has expired.

HSE_H SWT is reset on functional reset. The HSE_H SWT timeout signal is deasserted immediately on any reset.

28.5.2.15 HSE SNVS tamper detected

The HSE tamper-detected signal is asserted as soon as the SNVS has detected tampering that potentially jeopardizes the security of the chip.

The HSE SNVS tamper-detected signal is deasserted automatically by the SNVS after assertion and immediately on destructive reset and POR sequence entries.

The HSE software can determine which tamper was detected by reading a status register in the SNVS.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c4c462e5-4821-8a8-b251-290d3ab624c1

28.5.3 Functional reset sources

28.5.3.1 FCCU reset reaction

The FCCU reset reaction reset is asserted by the FCCU when a fault is detected by the FCCU and the fault is configured to generate a soft reaction.

The FCCU reset reaction reset is deasserted automatically by the FCCU after assertion and immediately on destructive reset and POR sequence entries.

Software can determine which fault caused the FCCU reset reaction by reading a status register in the FCCU.

28.5.3.2 Self-test done

The self-test done reset is asserted by the STCU2 after the execution of the BIST (LBIST/MBIST) has completed on the main reset domain.

The self-test done reset is deasserted automatically by the STCU2 after assertion and immediately on destructive and POR sequence entry.

28.5.3.3 SWT0 timeout

The SWT0 timeout reset request is asserted as soon as SWT0 has not been serviced by the software running on "Cortex-M7 cluster0" before the SWT's reset timer has expired. The SWT0 timeout signal is deasserted immediately on any reset sequence entry.

28.5.3.4 HSE memory ECC error

The HSE memory ECC error signal is asserted when the security subsystem's ECC circuitry detects an ECC error in any of the HSE's memories.

The HSE memory ECC error signal is deasserted immediately on any reset sequence entry.

28.5.3.5 HSE boot error

The HSE boot error signal is asserted when the security subsystem detects an error during boot code execution. The HSE boot error signal is deasserted immediately on any reset sequence entry.

28.5.3.6 HSE core lock

The HSE core lock signal is asserted when the security subsystem detects a lock condition on its Cortex-M7 core. The HSE core lock signal is deasserted immediately on any reset sequence entry.

28.5.3.7 Software functional reset

Software functional reset is asserted immediately upon a mode change request to a functional reset via the MC_ME.

Software functional reset is cleared immediately on functional reset, destructive reset, and POR sequence entries.

28.5.3.8 Debug functional reset

Debug functional reset is asserted immediately upon the setting of the functional reset field in the DAP's MDM_AP register by the debugger.

Debug functional reset is deasserted immediately upon the clearing of the functional reset field in the DAP's MDM_AP register by the debugger.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co., Ltd.
c8c62e5482148adb25139003ab624c1

28.5.4 Non-MC_RGM and indirect reset sources

28.5.4.1 JCOMP pin

The active-high JCOMP pin is input-only with a weak pulldown.

The JCOMP pin is asserted whenever JTAG operation is needed (for example, during debug) and is continuously deasserted when JTAG operation is not needed.

The JCOMP reset is asserted when the JCOMP pin is deasserted.

The JCOMP reset is deasserted when the JCOMP pin is asserted.

28.5.5 Destructive reset escalation to stay in reset

The destructive reset escalation reset is asserted by the MC_RGM immediately upon the destructive reset event that causes the escalation counter to increment and reach the threshold value in MC_RGM_DRET[DRET].

When the destructive reset escalation reset is asserted, the reset sequence immediately enters DEST0 state and remains there until a POR event starts a POR sequence.

The destructive reset escalation feature can be enabled and disabled via software.

The destructive reset escalation threshold value can be set via software.

The default configuration of the destructive reset escalation feature immediately after power-on sequence exit is disabled.

If enabled, the destructive reset escalation counter increments by one with each destructive reset event.

The destructive reset escalation counter can be cleared by software.

28.5.6 RESET_B

The active-low RESET_B pin is open-drain bidirectional with a weak pullup.

The input value of the RESET_B pin is ignored when the chip is asserting the RESET_B pin.

MC_RGM always asserts the RESET_B pin on destructive reset and POR sequence entry.

The default configuration, which is set immediately on destructive and power-on reset sequence entries, is that all functional reset events cause RESET_B to be asserted.

After MC_RGM has asserted the RESET_B pin, it continues to assert it until all the conditions for reset sequence exit have been met except for the check that RESET_B has been deasserted.

For all reset sequences, the reset sequence exit cannot occur until the RESET_B pin has reached its deassertion state; in other words, not until the RESET_B pin is not being asserted by MC_RGM nor by the external system.

The external reset event is asserted when an assertion edge is detected on the RESET_B pin when the RESET_B pin is not being asserted by MC_RGM.

The RESET_B pin input signal is filtered against high and low glitches with the following characteristics:

- Maximum pulse rejected: 20 ns
- Minimum pulse passed: 400 ns

The RESET_B pin is asserted by the off-chip power management circuitry starting before the device supplies begin ramping up and at least until the POR_B pin is deasserted. This prevents the RESET_B pin from behaving erratically when its supply is ramping up.

Preliminary Information COMPANY CONFIDENTIAL Beijing CHJ Automotive Technology Co., Ltd. c8c46e5-4821-4a4d-9251-39da3ab624c1

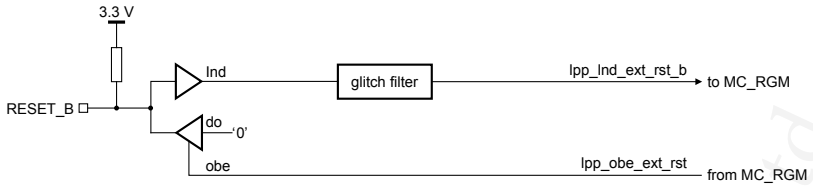


Figure 121. RESET_B overview

28.6 Reset process

28.6.1 Overview

The reset process is a sequence of several reset stages based on the reset event that has occurred. All reset events follow the same chip reset sequence with only the entry points varying depending on the reset event type. These stages are sequential in terms of execution. Each stage is triggered by the MC_RGM on completion of the previous stage. These stages are executed in a specific order to ensure a deterministic state of the system on exit from any reset event.

28.6.2 Reset process sequence

28.6.2.1 POR sequence

This stage is entered when a POR event occurs (for example, any of the POR sources are asserted). In this stage, the chip waits for the power-up sequence to complete. This stage is exited when all the POR sources clear.

The following modules are involved in POR reset sequence:

- PMC
- FIRC
- POR_B
- JTAGC/JCOMP

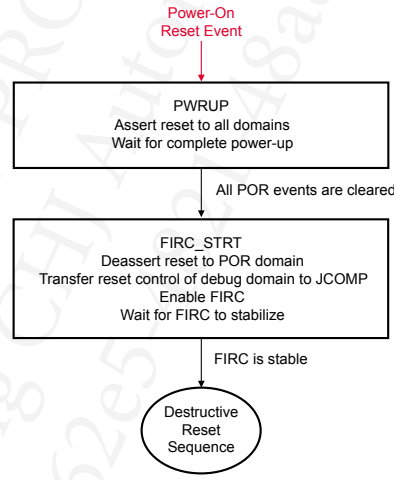


Figure 122. POR sequence flow diagram

28.6.2.2 Destructive reset sequence

The following modules are involved in the destructive reset sequence:

- OCOTP

- PMC
- FIRC
- SIRC
- Life-cycle logic

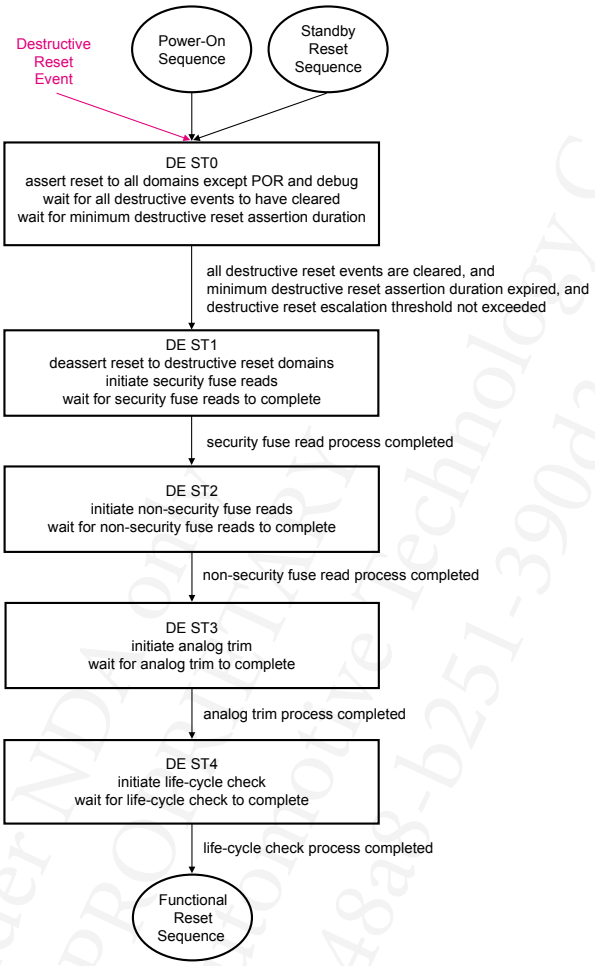


Figure 123. Destructive sequence flow diagram

NOTE

Since FIRC is the only clock source available during the reset sequence, and non-FIRC clock domains may glitch on destructive reset sequence entry, all IP in the POR domain run on the FIRC clock or TCK. One small exception to this is the POR_WDOG, which has an additional internally generated and independent clock that it uses for its watchdog timer.

28.6.2.3 Functional reset sequence

FUNC0 to FUNC7 in the flow diagram below represents the functional reset entry sequence. The following modules are involved in the functional reset entry sequence:

- STCU2
- FCCU/FOSU
- CMU

- SWT
- Debug/ETR/MDM-AP
- HSE
- NoC
- CGM
- DFS
- PLL
- FXOSC

FUNC8 to FUNC10 in the flow diagram below represent the functional reset exit sequence. The following modules are involved in the functional reset exit sequence:

- MTR/Memory repair
- RESET_B
- Boot configuration pins (Refer to the 'Boot modes' section of the 'Boot' chapter for details on the pins which are sampled during the reset sequence for boot mode selection and configuration)

NOTE

To allow the boot configuration pins to be properly sampled, you must ensure that the corresponding I/O power supplies are within their specified ranges. If this is not the case, the reset sequence is not prevented from completing, but could lead to an incorrect boot.

Provided under NDA of NXP
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39fd3b624c1

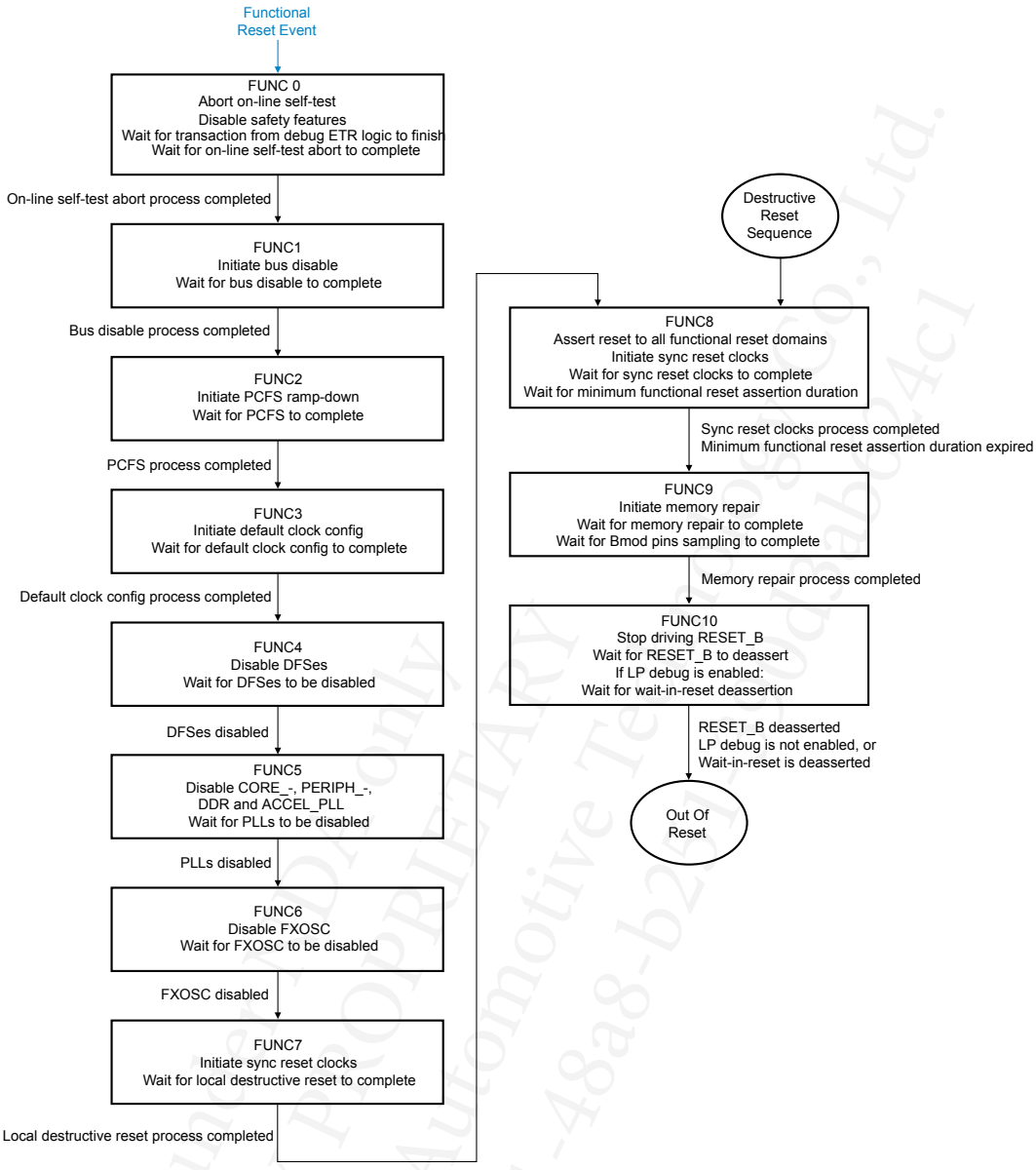


Figure 124. Functional reset entry sequence flow diagram

28.6.2.4 Standby reset sequence

The following modules are involved in the Standby reset sequence:

- Debug/TPIU/MDM-AP
- MC_PCU
- FIRC
- PMC
- SIRC

Preliminary Information
COMPANY CONFIDENTIAL

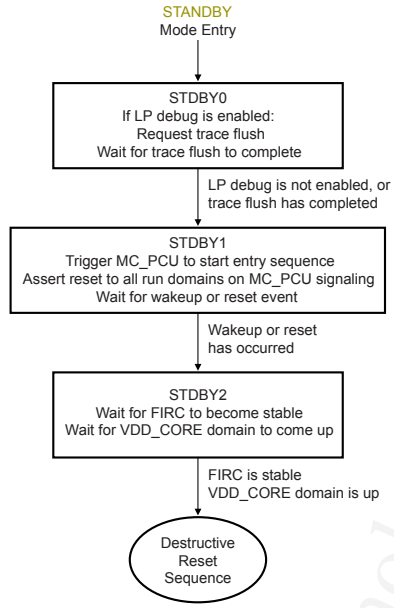


Figure 125. Standby reset sequence flow diagram

28.7 Chip status on reset exit

Table 115. Status of chip after POR

Functionality	After POR			
	Initial	During reset sequence	During Run	During Standby
Clock sources	FIRC and SIRC on, all others off			
Clock selection	Default values (FIRC for system)			
Clock dividers	Default values			
Reset status flags	Only F_POR is set			
MC_ME previous mode	Reset			
FCCU fault information	Cleared			
HSE RAM	Initialized			
SRAM content	Invalid			
DRAM content	Invalid			
Standby RAM content	Invalid			
Configuration from fuses	Loaded	Reloaded		
Cores	HSE on, all others off			
Resettable subsystem	All off and in reset			
POR domain (Run)	On with default configuration			

Table continues on the next page...

Table 115. Status of chip after POR (continued)

Functionality	After POR			
	Initial	During reset sequence	During Run	During Standby
POR domain (Standby)	On with default configuration			
Destructive reset domain (Run)	On with default configuration			
Destructive reset domain (Standby)	On with default configuration			
Functional reset domain (Run)	On with default configuration			
Standby domain	On with default configuration			
Standby pads	Default configuration			
Non-Standby pads	Default configuration			

Table 116. Status of chip after destructive reset

Functionality	During reset sequence	During Run	During Standby
Clock sources	FIRC and SIRC on, all others off		
Clock selection	Default value (FIRC for system)		
Clock dividers	Default values		
Reset status flags	F_DR_* is additionally set		F_DR_* and RGM_RDSS[DES_RES] are set
MC_ME previous mode	Reset		
FCCU fault information	Cleared		
HSE RAM	Initialized		
SRAM content	Invalid		
DRAM content	Invalid		
Standby RAM content	Invalid		
Configuration from fuses	Reloaded		
Cores	HSE on, all others off		
Resettable subsystem	All off and in reset		
POR domain (Run)	On with previous configuration		On with default configuration
POR domain (Standby)	On with previous configuration		
Destructive reset domain (Run)	On with default configuration		
Destructive reset domain (Standby)	On with default configuration		
Functional reset domain (Run)	On with default configuration		
Standby domain	On with default configuration		

Table continues on the next page...

Table 116. Status of chip after destructive reset (continued)

Functionality	During reset sequence	During Run	During Standby
Standby pads	Default configuration		
Non-Standby pads	Default configuration		

Table 117. Status of chip after functional reset

Functionality	During Run	During Standby
Clock sources	FIRC and SIRC on, all others off	
Clock selection	Default values (FIRC for system)	
Clock dividers	Default values	
Reset status flags	F_FR_* is additionally set	F_FR_* and RGM_RDSS[FES_RES] are set
MC_ME previous mode	Reset	
FCCU fault information	Retained	Cleared
HSE RAM	Initialized	
SRAM content	Retained	Invalid
DRAM content	Retained if retention is enabled, otherwise invalid	Invalid
Standby RAM content	Retained	
Configuration from fuses	Retained	
Cores	HSE on, all others off	
Resettable subsystem	All off and in reset	
POR domain (Run)	On with previous configuration	On with default configuration
POR domain (Standby)	On with previous configuration	
Destructive reset domain (Run)	On with previous configuration	On with default configuration
Destructive reset domain (Standby)	On with previous configuration	
Functional reset domain (Run)	On with default configuration	
Standby domain	On with default configuration	
Standby pads	Default configuration	
Non-Standby pads	Default configuration	

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-48c2-4e25-390d-3b124c1

Table 118. Status of chip after wake-up

Functionality	During Standby
Clock sources	FIRC on, SIRC as on Standby entry, all others off
Clock selection	Default values (FIRC for system)
Clock dividers	Default values
Reset status flags	No flags set
MC_ME previous mode	Standby
FCCU fault information	Cleared
HSE RAM	Initialized
SRAM content	Invalid
DRAM content	Retained if retention is enabled, otherwise invalid
Standby RAM content	Retained
Configuration from fuses	Retained
Cores	HSE on, all others off
Resettable subsystem	All off and in reset
POR domain (Run)	On with default configuration
POR domain (Standby)	On with previous configuration
Destructive reset domain (Run)	On with default configuration
Destructive reset domain (Standby)	On with previous configuration
Functional reset domain (Run)	On with default configuration
Standby domain	Unknown
Standby pads	As on Standby entry
Non-Standby pads	Default configuration

28.8 Module status during reset process

Table 119. Module status during reset

Module name	POR	Destructive reset	Functional reset	JCOMP	PRSTn	Power domain	Part of LBIST	Comment
MC_RGM	Y	N	N	N	N	Standby	No	
RDC	Y	Y	Y	N	N	Run	No	
PMC	Y	N	N	N	N	Standby	No	
FIRC	Y	N	N	N	N	Run	No	
POR-WDOG	Y	N	N	N	N	Standby	No	

Table continues on the next page...

Table 119. Module status during reset (continued)

Module name	POR	Destructive reset	Functional reset	JCOMP	PRSTn	Power domain	Part of LBIST	Comment
MC_CGM	Y	Y	N	N	N	Run	No	
OCOTP	Y	Y	N	N	N	Standby	No	
HSE RAM Init Logic	Y	Y	N	N	N	Run	Yes	Debug logic, ETM on destructive reset
HSE CAAM	Y	Y	Y	N	N	Run	Yes	
HSE Cortex-M7	Y	Y	Y	N	N	Run	Yes	Debug logic, ETM on destructive reset
HSE NoC	Y	Y	Y	N	N	Run	Yes	
HSE_H SWT	Y	Y	Y	N	N	Run	Yes	
HSE STM	Y	Y	Y	N	N	Run	Yes	
HSE JDC	Y	Y	Y	Y	N	Run	Yes	
HSE CMU	Y	Y	Y	N	N	Run	Yes	
HSE RAM Controllers	Y	Y	Y	N	N	Run	Yes	
HSE SCP (Schuffler)	Y	Y	N	N	N	Run	Yes	
HSE Registers	Y	Y	Y	N	N	Run	Yes	
HSE SNVS	Y	Y	Y	N	N	Run	Yes	
HSE SNVS RTC	Y	Y	Y	N	N	Run	Yes	
HSE MU	Y	Y	Y	N	N	Run	Yes	
Life cycle checking logic	Y	Y	N	N	N	Standby	No	
STCU2	Y	Y	N	N	N	Run	No	
FCCU	Y	Y	N	N	N	Run	Yes	
FOSU	Y	Y	N	N	N	Run	Yes	
MTR and BIST engines	Y	Y	Y	N	N	Run	Yes	It gets reset on functional reset asserion but they get out of reset before the end of reset sequence to support the memory repair loading before the end of reset sequence
MC_PCU	Y	Y	N	N	N	Standby	No	
WKPU	Y	Y	N	N	N	Standby	No	

Table continues on the next page...

Table 119. Module status during reset (continued)

Module name	POR	Destructive reset	Functional reset	JCOMP	PRSTn	Power domain	Part of LBIST	Comment
SIRC	Y	N	N	N	N	Standby	No	
FXOSC	Y	Y	Y	N	N	Run	No	
PLL	Y	Y	Y	N	N	Run	No	
DFS	Y	Y	Y	N	N	Run	No	
RTC	Y	Y	N	N	N	Standby	No	
QuadSPI	Y	Y	Y	N	N	Run	Yes	
Lockstep Cortex-M7 cores	Y	Y	Y	N	N	Run	Yes	
NoC	Y	Y	Y	N	N	Run	Yes	
OCP2IPS	Y	Y	Y	N	N	Run	Yes	
SEMA42	Y	Y	Y	N	N	Run	Yes	
GIC500	Y	Y	Y	N	Y	Run	Yes	
Ncore	Y	Y	Y	N	N	Run	Yes	
SWT	Y	Y	Y	N	N	Run	Yes	
PIT	Y	Y	Y	N	N	Run	Yes	
STM	Y	Y	Y	N	N	Run	Yes	
FTM	Y	Y	Y	N	N	Run	Yes	
MC_ME	Y	Y	Y	N	N	Run	Yes	
SIUL2	Y	Y	Y	N	N	Run	Yes	
SAR_ADC	Y	Y	Y	N	N	Run	Yes	Calibration logic is on destuctive reset
CTU	Y	Y	Y	N	N	Run	Yes	
FlexCAN	Y	Y	Y	N	N	Run	Yes	
LINFlex	Y	Y	Y	N	N	Run	Yes	
SPI	Y	Y	Y	N	N	Run	Yes	
IIC	Y	Y	Y	N	N	Run	Yes	
FlexRay	Y	Y	Y	N	N	Run	Yes	
LFAST	Y	Y	Y	N	N	Run	Yes	
USDHC	Y	Y	Y	N	N	Run	Yes	
GMAC (Ethernet)	Y	Y	Y	N	N	Run	Yes	

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHANGXIN Automotive Chip Technology Co., Ltd.
8c462e48-21a8-4f80-b25f-5900f3ab624c1

Table 119. Module status during reset (continued)

Module name	POR	Destructive reset	Functional reset	JCOMP	PRSTn	Power domain	Part of LBIST	Comment
DDR Controller	Y	Y	Y	N	N	Run	Yes	
DDR PHY	Y	Y	Y	N	N	Run	Yes	Functional reset executes warm reset sequence—destructive reset executes cold reset sequence
MSCM	Y	Y	Y	N	N	Run	Yes	
XRDC	Y	Y	Y	N	N	Run	Yes	
ERM	Y	Y	Y	N	N	Run	Yes	
EIM	Y	Y	Y	N	N	Run	Yes	
CRC	Y	Y	Y	N	N	Run	Yes	
CMU	Y	Y	Y	N	N	Run	Yes	
SbSW	Y	Y	Y	N	N	Run	Yes	
TMU	Y	Y	Y	N	N	Run	Yes	
eDMA	Y	Y	Y	N	N	Run	Yes	
DMAMUX	Y	Y	Y	N	N	Run	Yes	
SRAM Controller	Y	Y	Y	N	N	Run	Yes	
Aurora	Y	Y	N	N	N	Run	Yes	
JDC	Y	N	N	Y	N	Run	Yes	
JTAGC	Y	N	N	Y	N	Run	No	
Cortex-A53 (*n*CPUPORESET[1:0])	Y	Y	Y	N	Y	Run	Yes	
Cortex-A53 (*n*CORERESET[1:0])	Y	Y	Y	N	Y	Run	Yes	
Cortex-A53 (*n*L2RESET)	Y	Y	Y	N	Y	Run	Yes	
Cortex-A53 (*n*PRSTDEBUG)	Y	Y	N	N	Y	Run	Yes	
LLCE	Y	Y	Y	N	Y	Run	Yes	
PPFE	Y	Y	Y	N	Y	Run	Yes	
Standby SRAM	Y	N	N	N	N	Standby	Yes	
Timestamp	Y	Y	Y	N	N	Run	Yes	

Table continues on the next page...

Table 119. Module status during reset (continued)

Module name	POR	Destructive reset	Functional reset	JCOMP	PRSTn	Power domain	Part of LBIST	Comment
USBOTG	Y	Y	Y	N	N	Run	Yes	
Core debug including SWJ-DP, DAP, and multiple APs	Y	Y	N	N	N	Run	Yes	TCK domain logic of DAP/SWJ-DP works on JCOMP reset
Y	Indicates that the entire module is reset on this particular reset.							
Y	Indicates that some portion of the module is reset on this particular reset and some portion is not.							
N	Indicates that no portion of the module is reset on this particular reset.							

28.9 Determining reset cause on reset exit

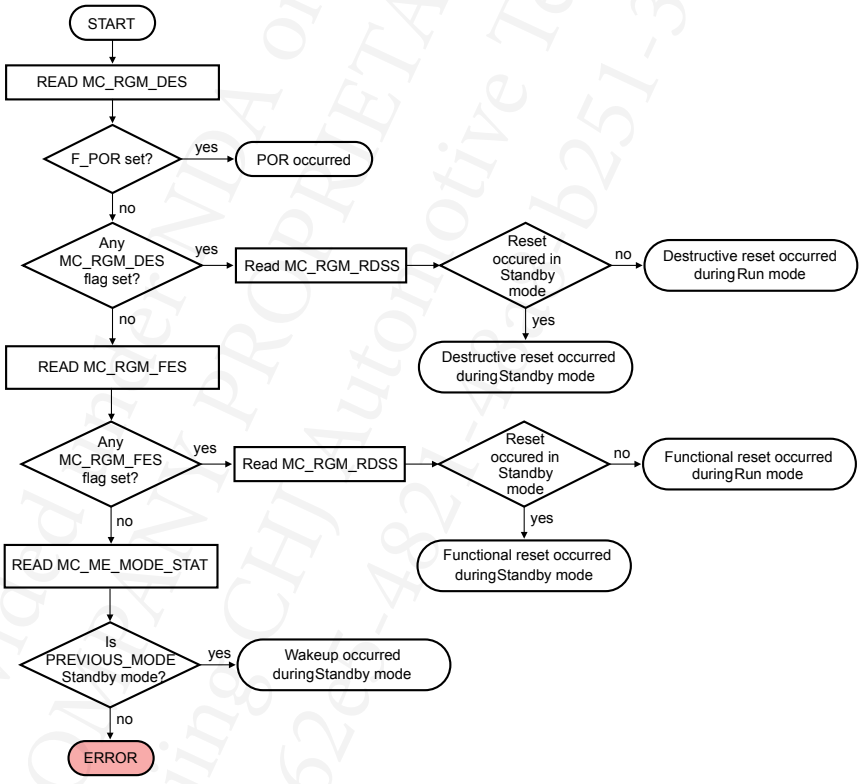


Figure 126. Reset cause on reset exit

28.10 Reset-related I/Os

Table 120. Reset-related I/Os

Name	Type	Pulls	Muxable	Description
POR_B	Input	Pulldown	No	Active-low power-on reset indicator from PMIC
RESET_B	Bidirectional	Pullup	No	Active-low open-drain external reset indicator
JCOMP	Input	Pulldown	No	JTAG compatibility input causes JTAG/debug logic to be reset when low

28.11 Pad states during reset sequence and immediately after reset exit

See the chip's IOMUX sheet.

28.12 Software resettable domains

28.12.1 Introduction

The chip is subdivided into several reset domains.

After POR, destructive, or functional reset, only the Main Reset Domain is released from reset. All other reset domains stay in reset until software writes the register field in MC_RGM that releases a given reset domain from reset.

Software can also write a register field in MC_RGM, and selectively put a reset domain back into reset, after ensuring a certain configuration. This configuration is controlled by MC_CGM, MC_ME, and RDC modules. These reset domains are referred to as software-resettable domains/partitions.

This reset structure results in better device availability. If a fault is detected in a software reset domain, that domain can be reset separately without impacting the operation of the rest of the chip.

MC_RGM provides 64 bits per reset domain/partition including the main reset domain. These bits are used for resetting the software-resettable domain/partition itself or cores.

- 32-bit register RGM_PRSTn_0[PERIPH_(64*n+31)_RST : PERIPH_(64*n)_RST]
- 32-bit register RGM_PRSTn_1[PERIPH_(64*n+63)_RST : PERIPH_(64*n+32)_RST]

The RGM_PRST0_0/1 registers correspond to the main reset partition, the RGM_PRST1_0/1 registers correspond to the first software resettable partition, the RGM_PRST2_0/1 registers (if present) correspond to the second software resettable partition, and so on.

RGM_PRSTn_0[PERIPH_(64*n)_RST] corresponds to partition reset.

Table 121. PRST fields for individual partitions and cores

PRST field	Partition reset	Core/Peripheral reset	Reset partition name	Core/Peripheral name
RGM_PRST0_0				
PERIPH_0_RST	No	Yes	Main	Cortex-M7 cluster0
PERIPH_1_RST	No	Yes	Main	Cortex-M7 cluster1
PERIPH_2_RST	No	Yes	Main	Cortex-M7 cluster2
PERIPH_3_RST	No	Yes	Main	DDR
PERIPH_4_RST	No	Yes	Main	PCIe
PERIPH_5_RST	No	Yes	Main	PCIe
PERIPH_16_RST	No	Yes	Main	PCIe_1

Table continues on the next page...

Table 121. PRST fields for individual partitions and cores (continued)

PRST field	Partition reset	Core/Peripheral reset	Reset partition name	Core/Peripheral name
PERIPH_17_RST	No	Yes	Main	PCIe_1
RGM_PRST1_0				
PERIPH_64_RST	Yes	No	Cortex-A53	NA
PERIPH_65_RST	No	Yes	Cortex-A53	Cortex-A53 CORE 0
PERIPH_66_RST	No	Yes	Cortex-A53	Cortex-A53 CORE 1
PERIPH_67_RST	No	Yes	Cortex-A53	Cortex-A53 CORE 2
PERIPH_68_RST	No	Yes	Cortex-A53	Cortex-A53 CORE 3
RGM_PRST2_0				
PERIPH_128_RST	Yes	No	PFE	
PERIPH_129_RST	No	Yes		
PERIPH_130_RST	No	Yes		
RGM_PRST3_0				
PERIPH_192_RST	Yes	No	LLCE	
PERIPH_193_RST	No	Yes		
PERIPH_194_RST	No	Yes		

28.12.2 List of software reset domains/partitions on the chip

The following table lists software resettable domains/partitions on the chip.

Table 122. Software-resettable domains/partitions list

Software-resettable domain	Modules	Reset control field in MC_RGM
RD1	Cortex-A53 Cluster	RGM_PRST1_0[PERIPH_64_RST]
RD2	PFE	RGM_PRST2_0[PERIPH_128_RST]
RD3	LLCE	RGM_PRST3_0[PERIPH_192_RST]

All other modules in the chip are part of the main reset domain (RD0).

NOTE

A module is disabled when the module clock is gated or reset is not de-asserted. Attempting to access a module that is disabled by such causes will result in invalid behavior with no error response, a transfer error, or a hang.

28.12.2.1 Cortex-A53 software-resettable domain/partition

This reset domain consists of:

- Two Cortex-A53 clusters
- One GIC 500 pair in lockstep for handling interrupts
- The coherency fabric (Ncore) including directory cache

- Delay and comparison units to support lockstep operation

Each cluster consists of two Cortex-A53 cores. Both clusters can be either in delayed lockstep to each other, or can be run independently, depending on the prior configuration before removing them from reset.

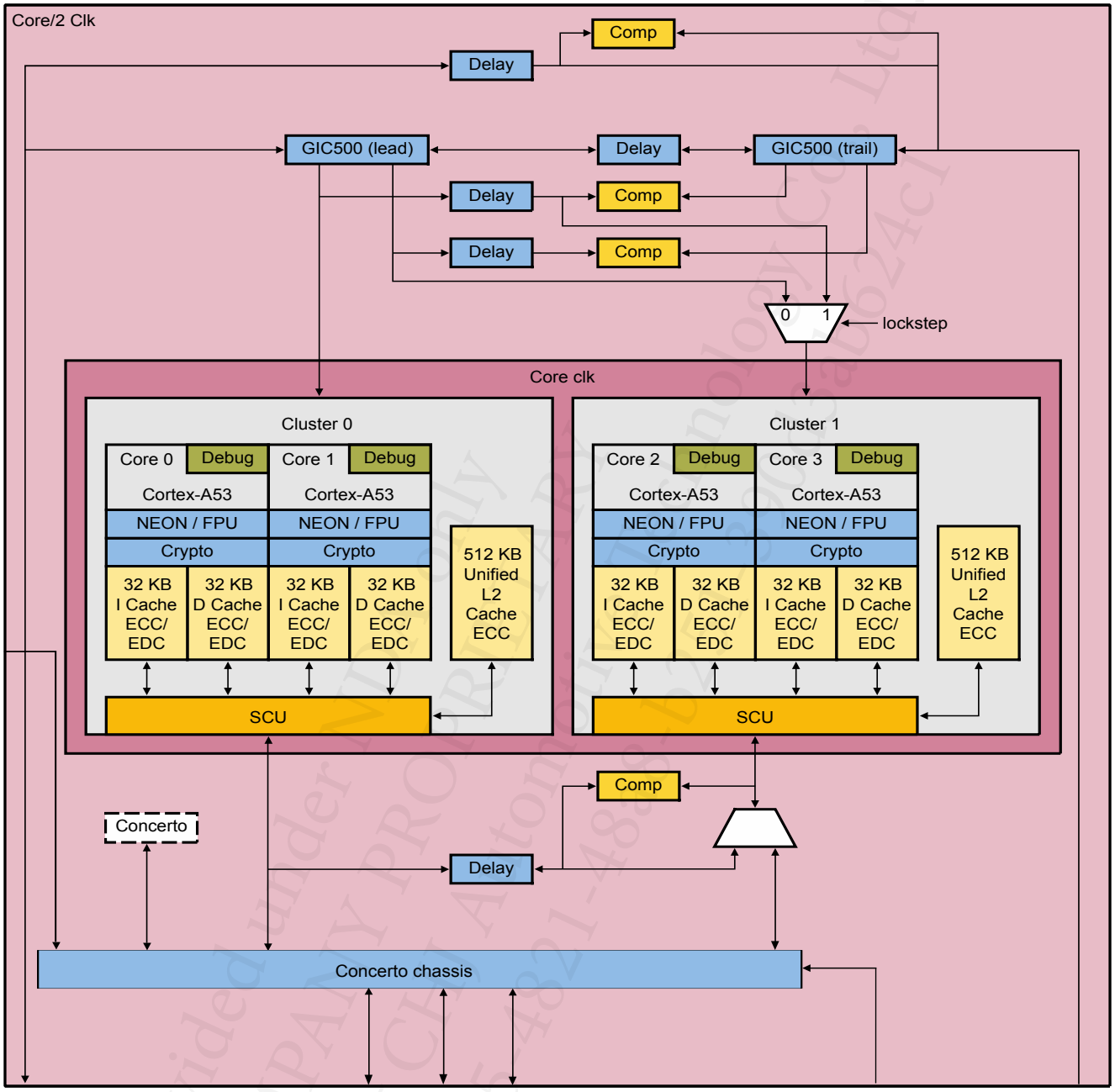


Figure 127. Software reset domain/partition

28.12.3 State of the software reset domains/partitions after power-up/destructive/functional reset sequence

All of the software resettable domains/partitions are in reset state. Clocks to the reset domains/partitions are gated. Outputs of the reset domains/partitions are safe-stated, that is, output isolation is asserted. The NoC/crossbar interface of bus masters in the software-resettable domain/partition is safe-stated with the NoC/crossbar.

28.12.4 Software reset partition turn-on flowchart

For the Cortex-A53 reset domain, clocks for the initial frequencies can be set up by configuring MC_CGM_1 to the FIRC clock. See the clocking chapter for details.

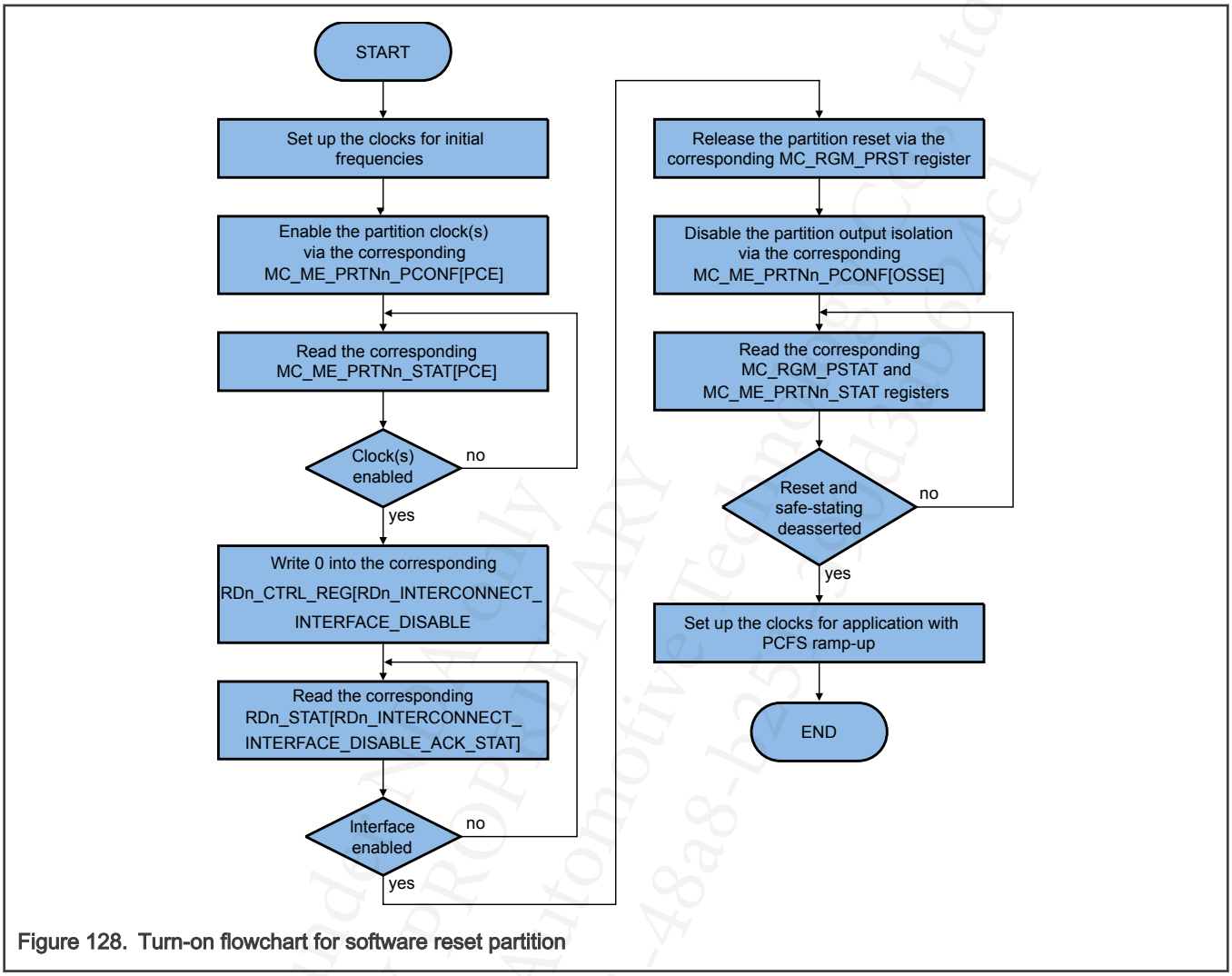


Figure 128. Turn-on flowchart for software reset partition

28.12.5 Software reset partition turn-off flowchart

For the Cortex-A53 reset domain, clocks for the initial frequencies can be set up by configuring MC_CGM_1 to the FIRC clock. See the clocking chapter for details.

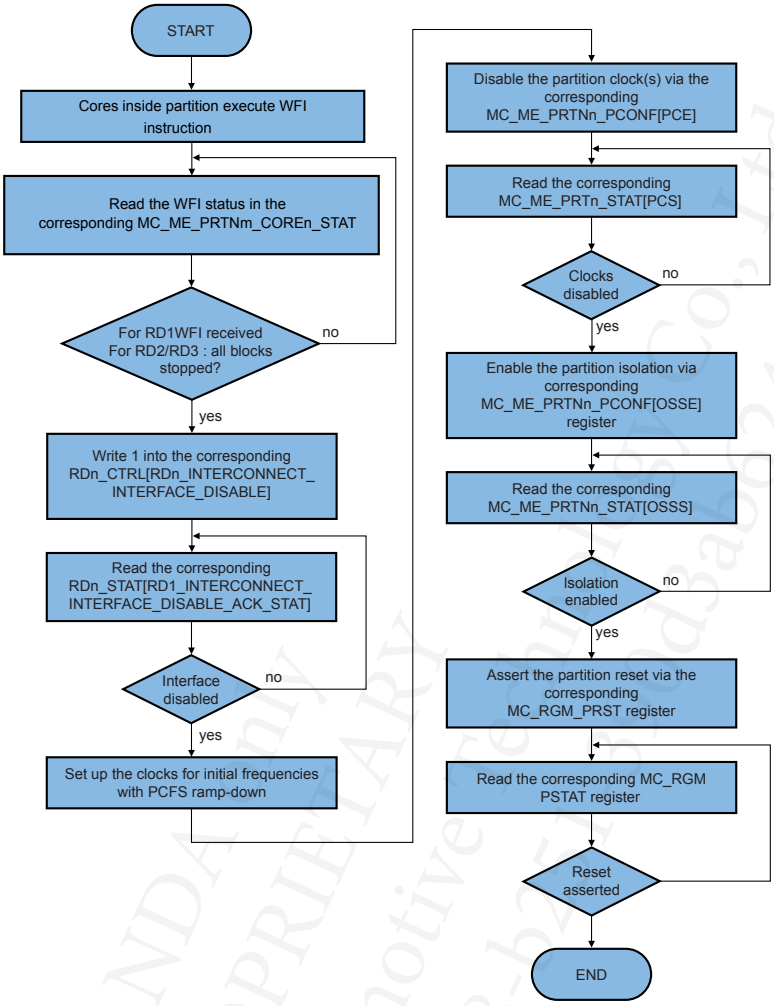


Figure 129. Turn-off flowchart for software reset partition

NOTE

The software must ensure that NoC masters inside a software reset domain remain in Idle state before executing the turn off sequence.

28.12.6 PFE software reset partition turn-off flowchart

For the PFE reset domain, clocks for the initial frequencies can be set up by configuring MC_CGM_1 to the FIRC clock. See the clocking chapter for details.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b25f-6dab624c1

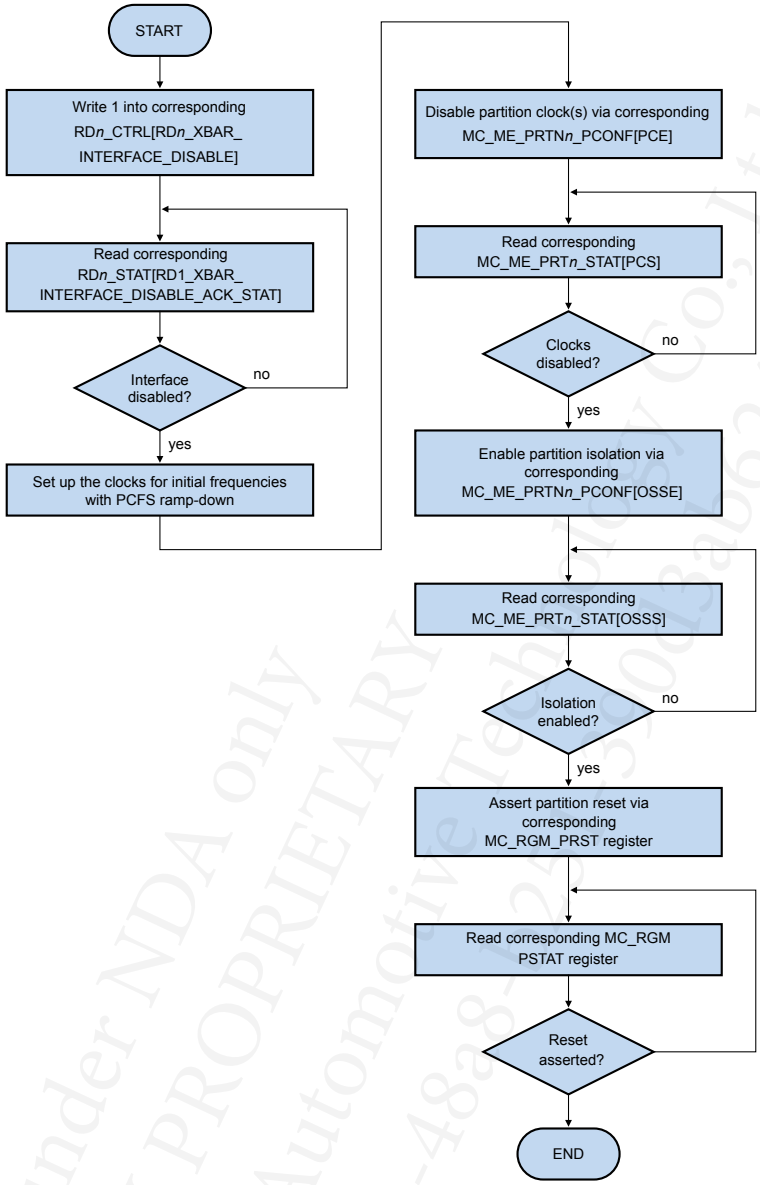


Figure 130. Turn-off flowchart for PFE software reset partition

28.12.7 Core turn-on sequence

This sequence can be used to turn on the Cortex-M7 application cores or the Cortex-A53 cluster cores.

Table 123. Core turn-on sequence

Core	Reset control field in MC_RGM
Cortex-M7 cluster 0	RGM_PRST0_0[PERIPH_0_RST]
Cortex-M7 cluster 1	RGM_PRST0_0[PERIPH_1_RST]
Cortex-M7 cluster 2	RGM_PRST0_0[PERIPH_2_RST]
Cortex-A53 core 0 – cluster 0	RGM_PRST1_0[PERIPH_65_RST]

Table continues on the next page...

Table 123. Core turn-on sequence (continued)

Core	Reset control field in MC_RGM
Cortex-A53 core 1 - cluster 0	RGM_PRST1_0[PERIPH_66_RST]
Cortex-A53 core 2 – cluster 1	RGM_PRST1_0[PERIPH_67_RST]
Cortex-A53 core 3 – cluster 1	RGM_PRST1_0[PERIPH_68_RST]

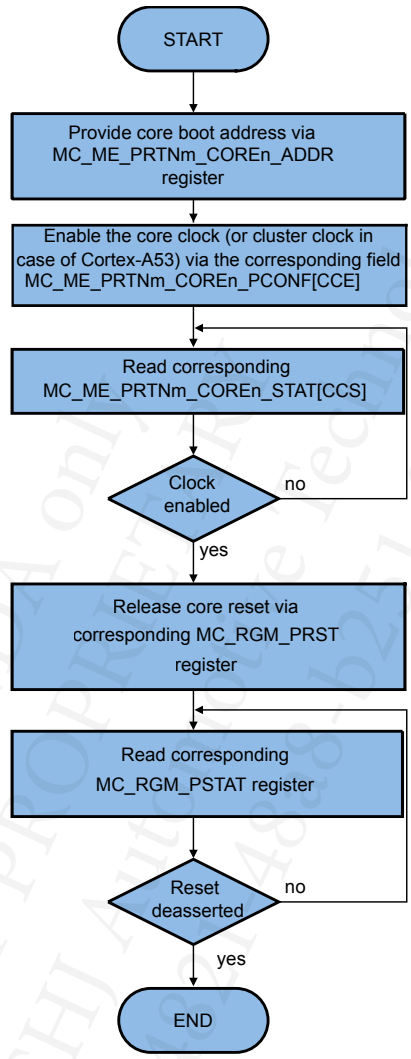


Figure 131. Core turn-on sequence

28.12.8 Core turn-off sequence

This sequence can be used to turn off only the Cortex-A53 cluster cores.

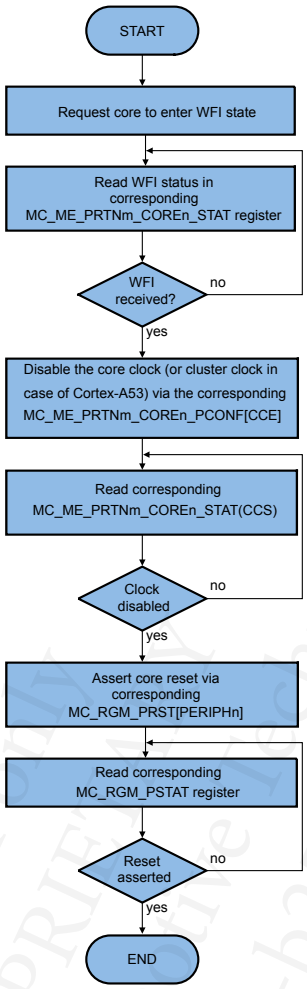


Figure 132. Core turn-off sequence

28.12.9 Self-triggered reset (warm reset) sequence for Cortex-M7 application core

This sequence can be used for performing the core-initiated warm reset of the Cortex-M7 application core by writing into its own register. The core automatically comes out of reset after the reset sequence is finished and starts executing from the provided boot address.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482a-48a8-b251-390d3ab624c1

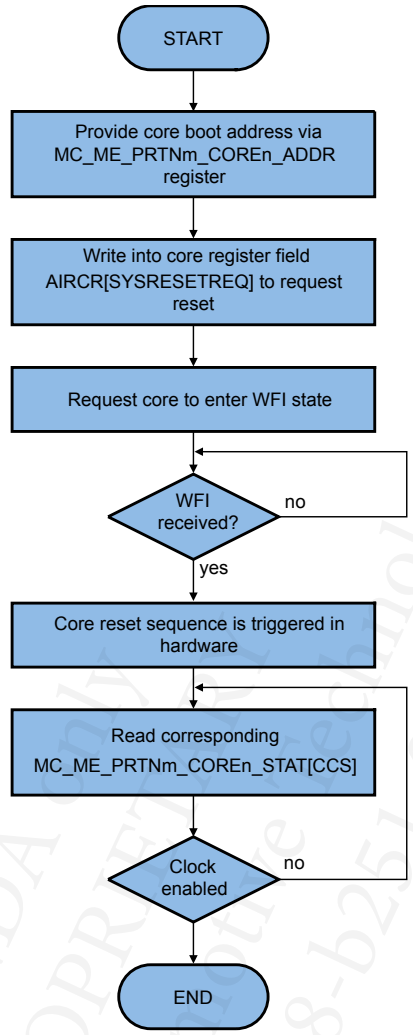


Figure 133. Self-triggered sequence

28.12.10 Cortex-A53 core warm reset sequence

You must follow these steps for the Cortex-A53 warm reset sequence.

1. Write RM[RR] to issue WARMRESETREQ.
2. Write 0 to SCTLR[C] to prevent further data cache allocation.
3. Clean and invalidate all data from the L1 data cache.
4. Disable data coherency with other cores in the cluster by clearing CPUECTLR[SMPEN].
5. Execute an ISB instruction to ensure that all register changes from the previous steps have been completed.
6. Execute a DSB SY instruction to ensure that all cache, TLB, and branch predictor maintenance operations issued by any core in the cluster device before the SMPEN field was cleared have completed.
7. Execute a WFI instruction and wait until the STANDBYWFI output is asserted.

Warm reset is issued to the core (that is, nCORERESET is asserted) when the WARMRESETREQ and STANDBYWFI signals are asserted and the SMPEN signal is deasserted. The duration of warm reset is 256 cycles of Cortex-A53_CORE_CLK.

For more information see Cortex-A53 documentation: Arm Cortex-A53 MPCore Processor Technical Reference Manual, Revision r0p4 at www.arm.com.

28.12.11 Software peripheral resets

Some peripherals (or logic on them) must exit reset only after software implements a particular configuration. In these cases, dedicated software fields ensure the peripherals exit reset.

Table 124. Fields for peripheral reset

Peripheral	Reset control field in MC_RGM	Description
DDR	RGM_PRST0_0[PERIPH_3_RST]	Reset for AXI interface and DDR controller
PCIe_0	RGM_PRST0_0[PERIPH_4_RST]	Reset control of PCIe_0
PCIe_0	RGM_PRST0_0[PERIPH_5_RST]	Power-on reset of SerDes subsystem
PCIe_1	RGM_PRST0_0[PERIPH_16_RST]	Reset control of PCIe_1
PCIe_1	RGM_PRST0_0[PERIPH_17_RST	Power-on reset of SerDes subsystem

NOTE

The peripheral resets, after deassertion, should not be asserted again using these register fields. They are asserted on the next functional, destructive, or POR reset.

NOTE

For a description of any soft reset feature supported by a peripheral, see the peripheral's dedicated chapter.

NOTE

The PRST0_0[PERIPH_3_RST] reset must deassert when DDR_CLK is selected as FIRC. DDR_CLK can be switched to functional frequency after the reset deasserts.

28.13 Reset Domain Controller (RDC) register descriptions

This section provides the memory map and descriptions of registers for configuring the system reset. All registers can be accessed using 8-bit, 16-bit, or 32-bit addressing.

28.13.1 RDC memory map

RDC base address: 4008_0000h

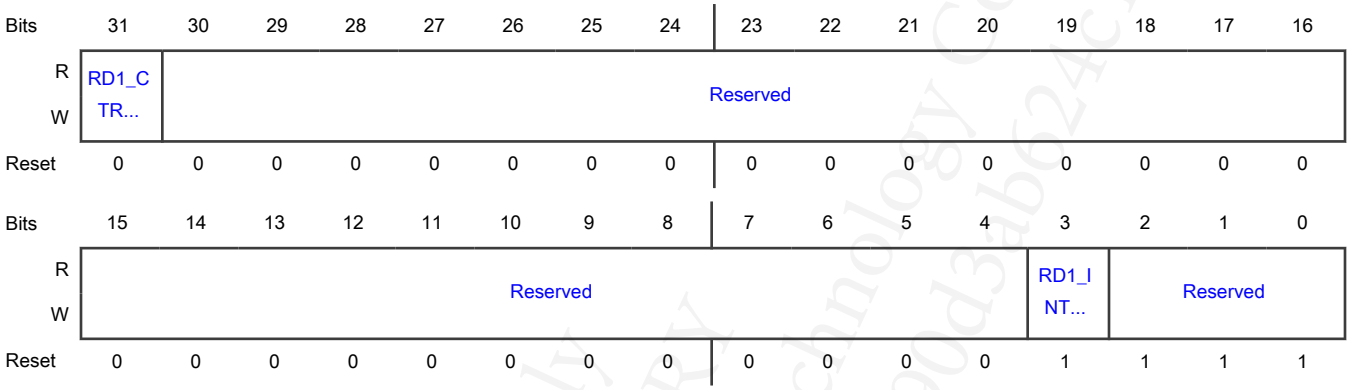
Offset	Register	Width (In bits)	Access	Reset value
4h	Software Reset Domain 1 Control (RD1_CTRL_REG)	32	RW	0000_000Fh
8h	Software Reset Domain 2 Control (RD2_CTRL_REG)	32	RW	0000_000Fh
Ch	Software Reset Domain 3 Control (RD3_CTRL_REG)	32	RW	0000_000Fh
84h	Software Reset Domain 1 Status (RD1_STAT_REG)	32	RO	0000_0018h
88h	Software Reset Domain 2 Status (RD2_STAT_REG)	32	RO	0000_0018h
8Ch	Software Reset Domain 3 Status (RD3_STAT_REG)	32	RO	0000_0018h

28.13.2 Software Reset Domain 1 Control (RD1_CTRL_REG)

Offset

Register	Offset
RD1_CTRL_REG	4h

Diagram



Fields

Field	Function
31 RD1_CTRL_UNLOCK	Reset Domain 1 Control Register Unlock 0b - Software Reset Domain 1 Control register fields are locked and cannot be updated except for this field 1b - Software Reset Domain 1 Control register fields are unlocked and can be updated
30-4 —	Reserved
3 RD1_INTERCONNECT_INTERFACE_DISABLE	Interconnect interface Disable of Software Reset Domain 1 0b - Enable interconnect interface of Software Reset Domain 1 1b - Disable interconnect interface of Software Reset Domain 1
2-0 —	Reserved

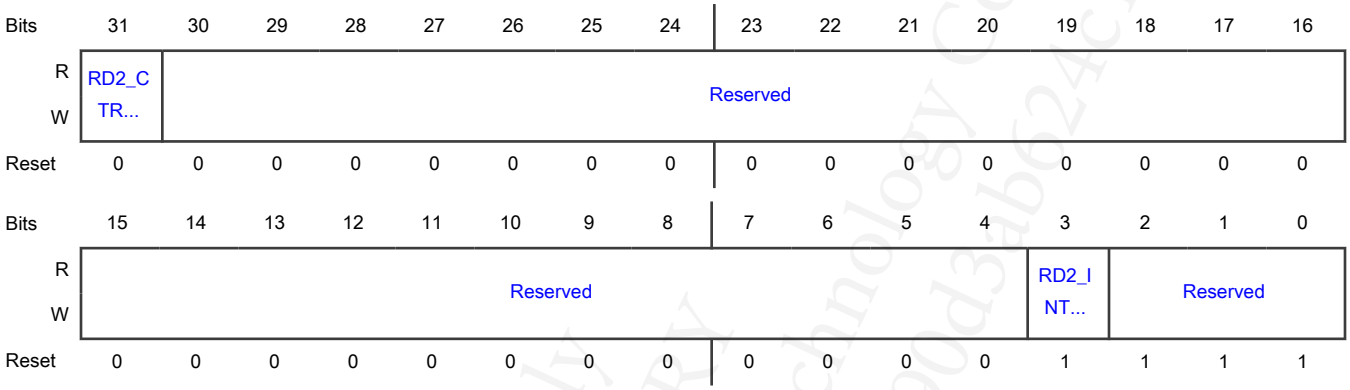
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-1251-390d34b624c1

28.13.3 Software Reset Domain 2 Control (RD2_CTRL_REG)

Offset

Register	Offset
RD2_CTRL_REG	8h

Diagram



Fields

Field	Function
31 RD2_CTRL_UNLOCK	Reset Domain 2 Control Register Unlock 0b - Software Reset Domain 2 control register fields are locked and cannot be updated except for this field 1b - Software Reset Domain 2 control register fields are unlocked and can be updated
30-4 —	Reserved
3 RD2_INTERCONNECT_INTERFACE_DISABLE	Interconnect interface Disable of Software Reset Domain 2 0b - Enable interconnect interface of Software Reset Domain 2 1b - Disable interconnect interface of Software Reset Domain 2
2-0 —	Reserved

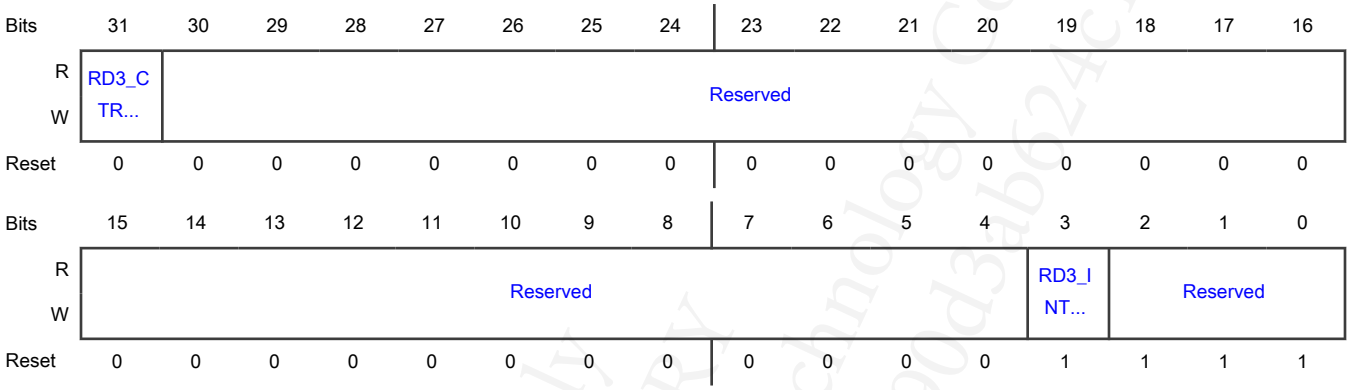
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-1251-390d34b624c1

28.13.4 Software Reset Domain 3 Control (RD3_CTRL_REG)

Offset

Register	Offset
RD3_CTRL_REG	Ch

Diagram



Fields

Field	Function
31 RD3_CTRL_UNLOCK	Reset Domain 3 Control Register Unlock 0b - Software Reset Domain 3 control register fields are locked and cannot be updated except for this field 1b - Software Reset Domain 3 control register fields are unlocked and can be updated
30-4 —	Reserved
3 RD3_INTERCONNECT_INTERFACE_DISABLE	Interconnect interface Disable of Software Reset Domain 3 0b - Enable interconnect interface of Software Reset Domain 3 1b - Disable interconnect interface of Software Reset Domain 3
2-0 —	Reserved

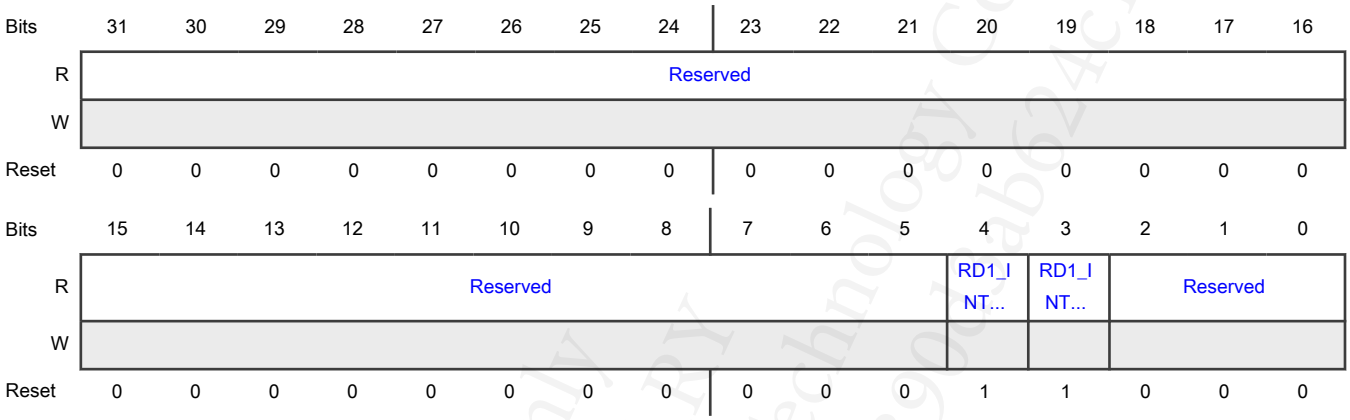
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-1251-390d34b624c1

28.13.5 Software Reset Domain 1 Status (RD1_STAT_REG)

Offset

Register	Offset
RD1_STAT_REG	84h

Diagram



Fields

Field	Function
31-5 —	Reserved
4 RD1_INTERCONNECT_INTERFACE_DISABLE_STAT	Interconnect Interface Disable Acknowledgment Status of Software Reset Domain 1 0b - Interconnect interface not disabled for Software Reset Domain 1 1b - Interconnect interface disabled for Software Reset Domain 1
3 RD1_INTERCONNECT_INTERFACE_DISABLE_REQ_ACK_STAT	Interconnect Interface Disable Request Acknowledgment Status of Software Reset Domain 1 0b - Interconnect interface disable request not acknowledged for Software Reset Domain 1 1b - Interconnect interface disable request acknowledged for Software Reset Domain 1
2-0 —	Reserved

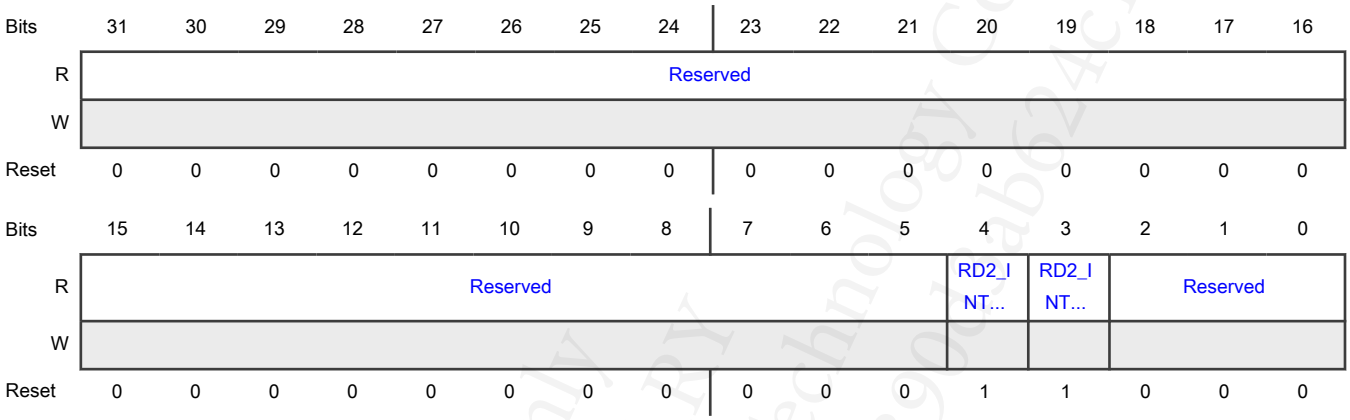
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-1c21-48a8-b251-3906ab624c1

28.13.6 Software Reset Domain 2 Status (RD2_STAT_REG)

Offset

Register	Offset
RD2_STAT_REG	88h

Diagram



Fields

Field	Function
31-5 —	Reserved
4 RD2_INTERCONNECT_INTERFACE_DISABLE_STAT	Interconnect Interface Disable Acknowledgment Status of Software Reset Domain 2 0b - Interconnect interface not disabled for Software Reset Domain 2 1b - Interconnect interface disabled for Software Reset Domain 2
3 RD2_INTERCONNECT_INTERFACE_DISABLE_REQ_ACK_STAT	Interconnect Interface Disable Request Acknowledgment Status of Software Reset Domain 2 0b - Interconnect interface disable request not acknowledged for Software Reset Domain 2 1b - Interconnect interface disable request acknowledged for Software Reset Domain 2
2-0 —	Reserved

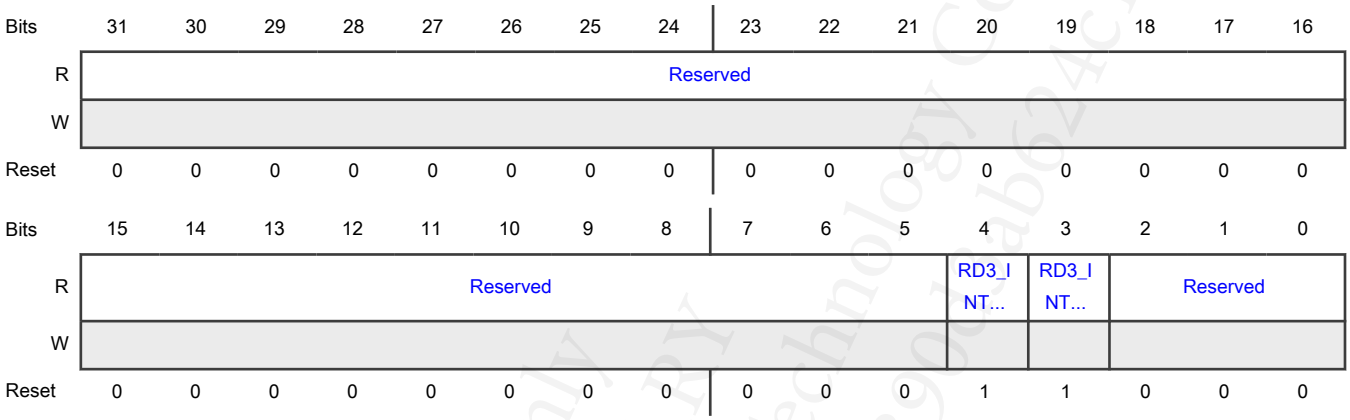
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-1c21-48a8-b251-39b6ab624c1

28.13.7 Software Reset Domain 3 Status (RD3_STAT_REG)

Offset

Register	Offset
RD3_STAT_REG	8Ch

Diagram



Fields

Field	Function
31-5 —	Reserved
4 RD3_INTERCONNECT_INTERFACE_DISABLE_STAT	Interconnect Interface Disable Acknowledgment Status of Software Reset Domain 3 0b - Interconnect interface not disabled for Software Reset Domain 3 1b - Interconnect interface disabled for Software Reset Domain 3
3 RD3_INTERCONNECT_INTERFACE_DISABLE_REQ_ACK_STAT	Interconnect Interface Disable Request Acknowledgment Status of Software Reset Domain 3 0b - Interconnect interface disable request not acknowledged for Software Reset Domain 3 1b - Interconnect interface disable request acknowledged for Software Reset Domain 3
2-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c462e5-1c21-48a8-b251-3906ab624c1

Chapter 29
Reset Generation Module (MC_RGM)

29.1 Chip-specific MC_RGM information

29.1.1 Reset sources

The MC_RGM module centralizes the various reset sources of the chip. For information on reset sources, see the "Reset sources" table in the "Reset sources" section of Reset chapter.

29.1.1.1 Destructive reset sources

In this chapter, each numbered destructive reset source, such as in the F_DR_*n* fields of the RGM_DES register, refers to the reset source with the same D[*n*] index number in the "Reset sources" table in the Reset chapter.

29.1.1.2 Functional reset sources

In this chapter, each numbered functional reset source, such as in the F_FR_*n* fields of the RGM_FES register, refers to the reset source with the same F[*n*] index number in the "Reset sources" table in the Reset chapter.

29.1.2 Peripheral numbers

For information on peripheral-number mapping, see the "PRST bits for individual partitions and cores" table in the "Software resettable domains" section of the Reset chapter.

29.1.3 Clear functional reset counter during initial application sequence

Boot ROM reads the RGM_FREC register to determine the number of functional resets. If the number of functional resets issued during Boot ROM is greater than eight, the boot process enters Fail-Safe, which is serial boot. To avoid having the Boot ROM enter Fail-Safe because of repeated functional resets from the application, clear RGM_FREC when you pass control to it.

29.1.4 Limitations on individual peripheral resets

This device supports individual peripheral resets in a limited manner. For details, see the "Software resettable domains" section of the Reset chapter.

29.2 Introduction

The Reset Generation Module (MC_RGM) centralizes the different reset sources and manages the reset sequence of the chip. It provides a register interface and the reset sequencer. There are various registers available in this module to monitor and control the chip reset sequence.

The following figure shows the block diagram of MC_RGM.

Provided under NXP Only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4888-b5e5-390a3ab624c1

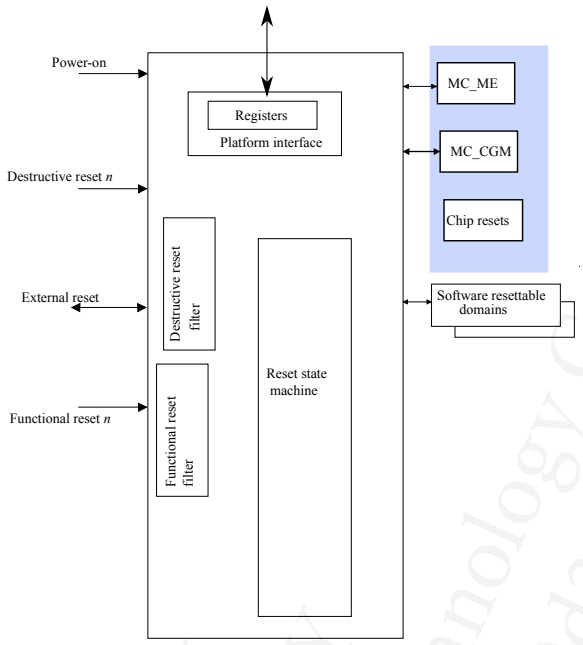


Figure 134. MC_RGM block diagram

29.3 Features

Here are the key features of MC_RGM:

- Support for destructive reset management
- Support for functional reset management
- Signaling of reset events after each reset sequence (reset status flags)
- Configurable escalation of recurring 'functional' resets to 'destructive' reset
- Configurable escalation of recurring 'destructive' resets to keep the chip in the reset state until the next power-on reset
- Support for resetting peripherals individually. For any limitations on this feature, see the chip-specific MC_RGM information
- Software controllable reset assertion
- Pad safe state control generation

29.4 Reset sources

The different reset sources are organized in three categories: power-on, destructive, and functional.

A power-on reset source is associated with an event typically related to power-up or low-voltage scenarios. When a power-on reset occurs, the full reset sequence is applied to the chip. This resets the full chip, including the MC_RGM, and the memory content must be considered to be unknown.

A destructive reset source is associated with an event related to a critical, usually hardware, error or dysfunction. When a destructive reset event occurs, the full reset sequence is applied to the chip. This resets the full chip (except for a small portion, for example FIRC, JTAG interface, and the MC_RGM itself) ensuring a safe start-up state for both digital and analog modules, and the memory content must be considered to be unknown.

A functional reset source is associated with an event related to a less-critical, usually non-hardware, error or dysfunction. When a functional reset event occurs, a partial reset sequence is applied to the chip. In this case, most digital modules are reset normally, while the state of analog modules or specific digital modules (for example, debug modules, flash modules) as well as the system memory content is preserved.

29.5 External signal description

The MC_RGM interfaces with the RESET_B pin.

The following table describes the signals that are connected to the I/O pad ring.

Table 125. MC_RGM external signals

Signal name	Reset value	Description
RESET_B	0	Active low external reset

29.6 RESET_B pin assertion and pin safe state control

The MC_RGM asserts the RESET_B pin when the device is in a reset sequence, and it remains asserted until the end of the reset sequence. During this reset sequence, most of the chip's pins are safe-stated according to the values shown in the IOMUX table. Note that the safe state values may vary according to the reset sequence type.

In addition, the MC_RGM has a feature to assert the RESET_B pin through software, without initiating a reset sequence. This is achieved by writing 1b to the ERCTRL[ERASSERT]. When this occurs, most of the chip's pins are safe-stated according to the values shown in the IOMUX table. The RESET_B assertion and pin safe-stating remain active until the end of the next reset sequence.

This feature is intended for use only in conjunction with self-test of the main reset domain. See STCU2 chapter for more details on its usage.

29.7 MC_RGM register descriptions

29.7.1 MC RGM Register Map memory map

MC_RGM base address: 4007_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Destructive Event Status Register (DES)	32	W1C	0000_0001h
8h	Functional /External Reset Status Register (FES)	32	W1C	0000_0000h
Ch	Functional Event Reset Disable Register (FERD)	32	RWONCE	0000_0000h
14h	Functional Reset Escalation Counter Register (FREC)	32	W1C	0000_0000h
18h	Functional Reset Escalation Threshold Register (FRET)	32	RW	0000_000Fh
1Ch	Destructive Reset Escalation Threshold Register (DRET)	32	RW	0000_0000h
20h	External Reset Control Register (ERCTRL)	32	RW	0000_0000h
24h	Reset During Standby Status Register (RDSS)	32	W1C	0000_0000h
40h	Peripheral Reset (PRST0_0)	32	RW	FFFF_FFFFh
48h	Peripheral Reset (PRST1_0)	32	RW	0000_001Fh
50h	Peripheral Reset (PRST2_0)	32	RW	0000_01FFh
58h	Peripheral Reset (PRST3_0)	32	RW	0000_01FFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
60h	Peripheral Reset (PRST4_0)	32	RW	0000_01FFh
68h	Peripheral Reset (PRST5_0)	32	RW	0000_01FFh
70h	Peripheral Reset (PRST6_0)	32	RW	0000_01FFh
78h	Peripheral Reset (PRST7_0)	32	RW	0000_01FFh
140h	Peripheral Reset Status Register (PSTAT0_0)	32	RO	FFFF_FFFFh
148h	Peripheral Reset Status Register (PSTAT1_0)	32	RO	0000_001Fh
150h	Peripheral Reset Status Register (PSTAT2_0)	32	RO	0000_01FFh
158h	Peripheral Reset Status Register (PSTAT3_0)	32	RO	0000_01FFh
160h	Peripheral Reset Status Register (PSTAT4_0)	32	RO	0000_01FFh
168h	Peripheral Reset Status Register (PSTAT5_0)	32	RO	0000_01FFh
170h	Peripheral Reset Status Register (PSTAT6_0)	32	RO	0000_01FFh
178h	Peripheral Reset Status Register (PSTAT7_0)	32	RO	0000_01FFh

29.7.2 Destructive Event Status Register (DES)

Offset

Register	Offset
DES	0h

Function

This register contains the status of the 'destructive' reset sources. It can be accessed in read/write in supervisor mode. It can be accessed in read-only in the user mode. Register bits are cleared on write '1'. This register is reset only on power-on reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	F_DR_31	F_DR_30	0	0	0	0	0	0	0	0	0	0	0	F_DR_18	F_DR_17	F_DR_16
W	W1C	W1C												W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	F_DR_13	F_DR_12	F_DR_11	F_DR_10	F_DR_9	F_DR_8	0	F_DR_6	0	F_DR_4	F_DR_3	0	F_DR_1	F_POR
W			W1C	W1C	W1C	W1C	W1C	W1C		W1C		W1C	W1C		W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Provided under NDA
COMPANY PROPRIETARY
Freescale Semiconductor, Inc.
Confidential and Proprietary Information
Freescale Semiconductor, Inc.

Fields

Field	Function
31 F_DR_31	Flag for 'Destructive' Reset F_DR_31 0b - 'Destructive' reset event F_DR_31 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_31 has occurred.
30 F_DR_30	Flag for 'Destructive' Reset F_DR_30 0b - 'Destructive' reset event F_DR_30 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_30 has occurred.
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 F_DR_18	Flag for 'Destructive' Reset F_DR_18 0b - 'Destructive' reset event F_DR_18 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_18 has occurred.
17 F_DR_17	Flag for 'Destructive' Reset F_DR_17 0b - 'Destructive' reset event F_DR_17 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_17 has occurred.
16 F_DR_16	Flag for 'Destructive' Reset F_DR_16 0b - 'Destructive' reset event F_DR_16 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_16 has occurred.
15 —	Reserved
14 —	Reserved
13 F_DR_13	Flag for 'Destructive' Reset F_DR_13 0b - 'Destructive' reset event F_DR_13 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_13 has occurred.
12 F_DR_12	Flag for 'Destructive' Reset F_DR_12 0b - 'Destructive' reset event F_DR_12 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_12 has occurred.
11 F_DR_11	Flag for 'Destructive' Reset F_DR_11 0b - 'Destructive' reset event F_DR_11 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_11 has occurred.
10 F_DR_10	Flag for 'Destructive' Reset F_DR_10 0b - 'Destructive' reset event F_DR_10 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_10 has occurred.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 F_DR_9	Flag for 'Destructive' Reset F_DR_9 0b - 'Destructive' reset event F_DR_9 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_9 has occurred.
8 F_DR_8	Flag for 'Destructive' Reset F_DR_8 0b - 'Destructive' reset event F_DR_8 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_8 has occurred.
7 —	Reserved
6 F_DR_6	Flag for 'Destructive' Reset F_DR_6 0b - 'Destructive' reset event F_DR_6 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_6 has occurred.
5 —	Reserved
4 F_DR_4	Flag for 'Destructive' Reset F_DR_4 0b - 'Destructive' reset event F_DR_4 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_4 has occurred.
3 F_DR_3	Flag for 'Destructive' Reset F_DR_3 0b - 'Destructive' reset event F_DR_3 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_3 has occurred.
2 —	Reserved
1 F_DR_1	Flag for 'Destructive' Reset F_DR_1 0b - 'Destructive' reset event F_DR_1 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Destructive' reset event F_DR_1 has occurred.
0 F_POR	Flag for power-on reset 0b - No power-on event has occurred since the last clear. 1b - A power-on event has occurred.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca46e0-482-48a-1251-390d3ab624c1

29.7.3 Functional /External Reset Status Register (FES)

Offset

Register	Offset
FES	8h

Function

This register contains the status of the 'functional' and external reset sources. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode. Register bits are cleared on write '1' if the triggering event has already been cleared at the source.This register is reset only on power-on reset.

NOTE

The status of this register must be ignored if the fields of Destructive Event Status Register (DES) other than DES[F_POR] are set. In this case, a destructive reset had occurred while functional reset source was active.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	F_FR_31	F_FR_30	0	0	0	0	0	0	0	0	F_FR_21	F_FR_20	0	F_FR_18	0	0
W	W1C	W1C									W1C	W1C		W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	F_FR_6	0	F_FR_4	F_FR_3	0	0	F_EXR
W										W1C		W1C	W1C			W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 F_FR_31	Flag for 'Functional' Reset F_FR_31 0b - 'Functional' reset event F_FR_31 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event F_FR_31 has occurred.
30 F_FR_30	Flag for 'Functional' Reset F_FR_30 0b - 'Functional' reset event F_FR_30 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event F_FR_30 has occurred.
29 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 F_FR_21	Flag for 'Functional' Reset F_FR_21 0b - 'Functional' reset event F_FR_21 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event F_FR_21 has occurred.
20 F_FR_20	Flag for 'Functional' Reset F_FR_20 0b - 'Functional' reset event F_FR_20 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event F_FR_20 has occurred.
19 —	Reserved
18 F_FR_18	Flag for 'Functional' Reset F_FR_18 0b - 'Functional' reset event F_FR_18 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event F_FR_18 has occurred.
17 —	Reserved
16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 F_FR_6	Flag for 'Functional' Reset F_FR_6 0b - 'Functional' reset event F_FR_6 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event F_FR_6 has occurred.
5 —	Reserved
4 F_FR_4	Flag for 'Functional' Reset F_FR_4 0b - 'Functional' reset event F_FR_4 has not occurred since either the last clear or the last power-on reset assertion. 1b - 'Functional' reset event F_FR_4 has occurred.
3	Flag for 'Functional' Reset F_FR_3 0b - 'Functional' reset event F_FR_3 has not occurred since either the last clear or the last

Table continues on the next page...

Table continued from the previous page...

Field	Function
F_FR_3	power-on reset assertion. 1b - 'Functional' reset event F_FR_3 has occurred.
2 —	Reserved
1 —	Reserved
0 F_EXR	Flag for External Reset 0b - No external reset event has occurred since either the last clear or the last power-on reset assertion. 1b - An external reset event has occurred.

29.7.4 Functional Event Reset Disable Register (FERD)

Offset

Register	Offset
FERD	Ch

Function

This register provides dedicated fields to disable functional reset sources. When any of these reset sources are disabled, the associated functional event is demoted to trigger an interrupt request. The register can be read and written to in Supervisor mode and only read in User mode. Each byte can be written to only once after a destructive or power-on reset and this register is reset only on power-on and any destructive reset.

NOTE

It is important to clear the Functional /External Reset Status Register (FES) before writing 1 to any of the fields in this register. Otherwise a interrupt request may occur.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	D_F_FR_...	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	D_EXR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 D_F_FR_31	F_FR_31 Disable Control 0b - Functional reset event F_FR_31 triggers a reset sequence. 1b - Functional reset event F_FR_31 generates an interrupt request.
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 —	Reserved
1 —	Reserved
0 D_EXR	External Reset event demote to interrupt 0b - 'External' reset event triggers a reset sequence. 1b - 'External' reset event generates an interrupt request.

29.7.5 Functional Reset Escalation Counter Register (FREC)

Offset

Register	Offset
FREC	14h

Function

This register provides the current count of functional reset escalation counter. It can be accessed in read/write, either in supervisor mode. It can be accessed in read in the user mode. This register is reset only on power-on reset or destructive reset. This register is also reset when you reconfigure the FREC field to Fh.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												FREC			
W													W1C			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-251-390d3ab624c1

Fields

Field	Function
31-4 —	Reserved
3-0 FREC	Functional' Reset Escalation Counter This bit field provides the value of functional reset escalation counter.

29.7.6 Functional Reset Escalation Threshold Register (FRET)

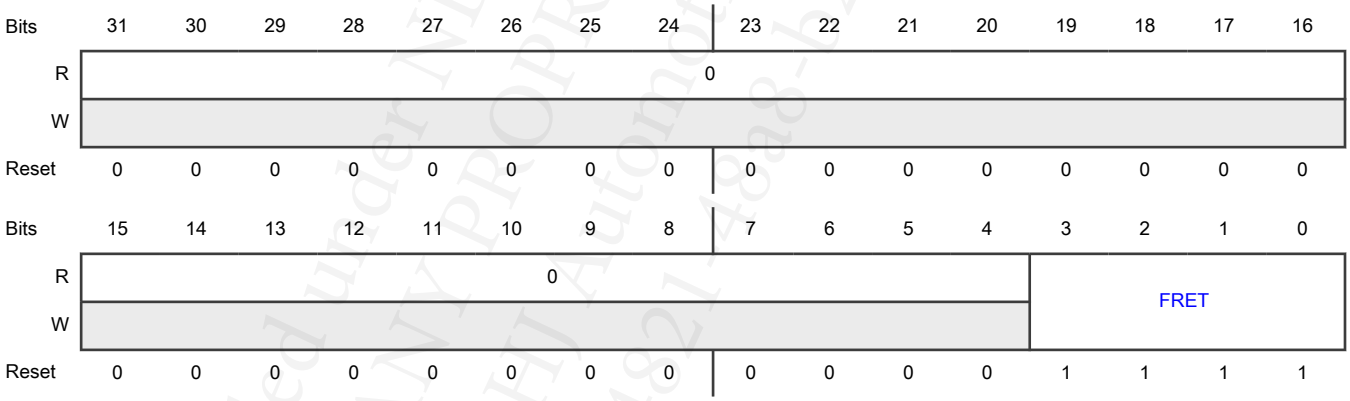
Offset

Register	Offset
FRET	18h

Function

This register sets the threshold for 'functional' reset escalation to a 'destructive' reset. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode. Writing a non-zero value to the FRET field enables the 'functional' reset escalation function. Writing any value to this register resets the 'functional' reset counter. See Functional reset escalation for details on the 'functional' reset escalation function. This register is reset only on power-on and any 'destructive' reset.

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 FRET	'Functional' Reset Escalation Threshold

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If the value of this field is 0, the 'functional' reset escalation function is disabled. Any other value is the number of 'functional' resets that causes a 'destructive' reset if the FRET register isn't written to beforehand.

29.7.7 Destructive Reset Escalation Threshold Register (DRET)

Offset

Register	Offset
DRET	1Ch

Function

This register sets the threshold for 'destructive' reset escalation to keeping the chip in the reset state until the next power-on reset triggers a new reset sequence. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode. Writing a non-zero value to the DRET field enables the 'destructive' reset escalation function. Writing any value to this register resets the 'destructive' reset counter. See Destructive reset escalation for details on the 'destructive' reset escalation function. This register is reset only on power-on reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												DRET			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 DRET	'Destructive' Reset Escalation Threshold If the value of this field is 0, the 'destructive' reset escalation function is disabled. Any other value is the number of 'destructive' resets which keeps the chip in the reset state until the next power-on reset triggers a new reset sequence if the DRET register isn't written to beforehand.

29.7.8 External Reset Control Register (ERCTRL)

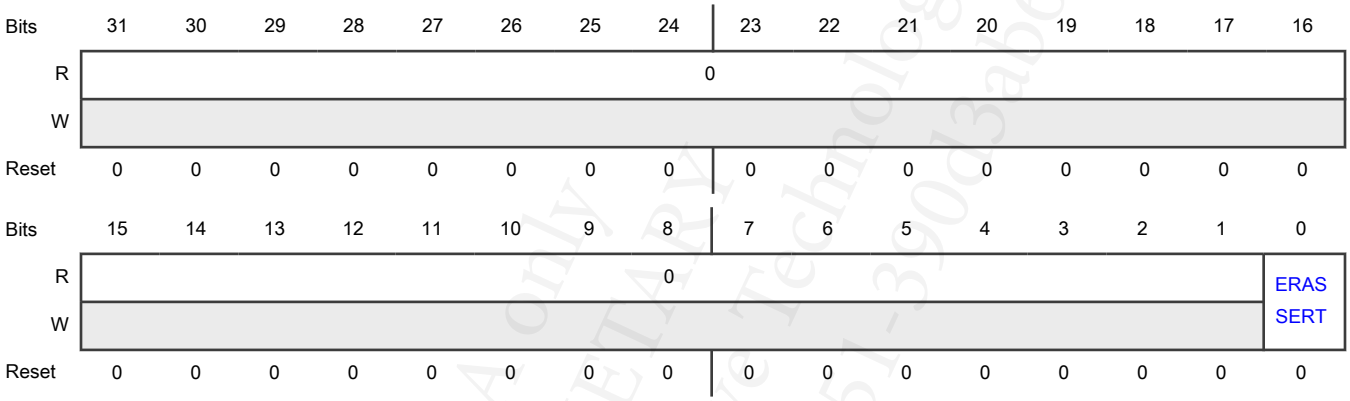
Offset

Register	Offset
ERCTRL	20h

Function

This register allows software to control the assertion of External reset pin. It can be accessed in read/write, in supervisor mode. It can be accessed in read-only in the user mode. This register is reset on all resets.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 ERASSERT	ERASSERT **NOTE** Setting ERASSERT to 1b also safe states most of the chip's pins. See the IOMUX table for each pin's safe state value. Software must use the ERASSERT bit for this purpose only as part of the main reset domain self-test entry procedure. Using it at any other time may result in unpredictable system behavior. 0b - No change 1b - External reset is asserted

Provided under NDA only
COMPANY PROPRIETARY
Beijing eSOL Automotive Technology Co., Ltd.
c8c462e5-1621-48a8-b251-39003ab6240d

29.7.9 Reset During Standby Status Register (RDSS)

Offset

Register	Offset
RDSS	24h

Function

This register provides status of whether a reset event occurred during standby mode. Register bits are cleared on write '1'. This register is reset only on power-on reset.

NOTE

On exiting a reset sequence after standby exit, the software must perform a read operation on MC_ME[PREV_MODE] and RDSS register. If any bit of the RDSS register is set, the software must ignore the status reported by MC_ME[PREV_MODE] register otherwise the status of MC_ME[PREV_MODE] register reports the device status.

If MC_ME indicates last mode as RESET, then perform a reset exit in software else a standby exit.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														0	DES_RES
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	Reserved
1 —	Reserved
0 DES_RES	DES_RES 0b - No destructive reset event occurred during standby mode. 1b - Destructive reset event occurred during standby mode.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
d246e2e5-4821-46a8-b251-590a8a6e2401

29.7.10 Peripheral Reset (PRST0_0)

Offset

Register	Offset
PRST0_0	40h

Function

This register provides individual resets for various peripherals. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31 PERIPH_31_RST	Reset Control value for peripheral PERIPH_31_RST 0b - No forced reset on PERIPH_31_RST 1b - Forced Reset on PERIPH_31_RST
30 PERIPH_30_RST	Reset Control value for peripheral PERIPH_30_RST 0b - No forced reset on PERIPH_30_RST 1b - Forced Reset on PERIPH_30_RST
29 PERIPH_29_RST	Reset Control value for peripheral PERIPH_29_RST 0b - No forced reset on PERIPH_29_RST 1b - Forced Reset on PERIPH_29_RST
28 PERIPH_28_RST	Reset Control value for peripheral PERIPH_28_RST 0b - No forced reset on PERIPH_28_RST 1b - Forced Reset on PERIPH_28_RST
27	Reset Control value for peripheral PERIPH_27_RST

Table continues on the next page...

Table continued from the previous page...

Field	Function
PERIPH_27_RST	0b - No forced reset on PERIPH_27_RST 1b - Forced Reset on PERIPH_27_RST
26 PERIPH_26_RST	Reset Control value for peripheral PERIPH_26_RST 0b - No forced reset on PERIPH_26_RST 1b - Forced Reset on PERIPH_26_RST
25 PERIPH_25_RST	Reset Control value for peripheral PERIPH_25_RST 0b - No forced reset on PERIPH_25_RST 1b - Forced Reset on PERIPH_25_RST
24 PERIPH_24_RST	Reset Control value for peripheral PERIPH_24_RST 0b - No forced reset on PERIPH_24_RST 1b - Forced Reset on PERIPH_24_RST
23 PERIPH_23_RST	Reset Control value for peripheral PERIPH_23_RST 0b - No forced reset on PERIPH_23_RST 1b - Forced Reset on PERIPH_23_RST
22 PERIPH_22_RST	Reset Control value for peripheral PERIPH_22_RST 0b - No forced reset on PERIPH_22_RST 1b - Forced Reset on PERIPH_22_RST
21 PERIPH_21_RST	Reset Control value for peripheral PERIPH_21_RST 0b - No forced reset on PERIPH_21_RST 1b - Forced Reset on PERIPH_21_RST
20 PERIPH_20_RST	Reset Control value for peripheral PERIPH_20_RST 0b - No forced reset on PERIPH_20_RST 1b - Forced Reset on PERIPH_20_RST
19 PERIPH_19_RST	Reset Control value for peripheral PERIPH_19_RST 0b - No forced reset on PERIPH_19_RST 1b - Forced Reset on PERIPH_19_RST
18 PERIPH_18_RST	Reset Control value for peripheral PERIPH_18_RST 0b - No forced reset on PERIPH_18_RST 1b - Forced Reset on PERIPH_18_RST
17 PERIPH_17_RST	Reset Control value for peripheral PERIPH_17_RST 0b - No forced reset on PERIPH_17_RST 1b - Forced Reset on PERIPH_17_RST

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 PERIPH_16_RST	Reset Control value for peripheral PERIPH_16_RST 0b - No forced reset on PERIPH_16_RST 1b - Forced Reset on PERIPH_16_RST
15 PERIPH_15_RST	Reset Control value for peripheral PERIPH_15_RST 0b - No forced reset on PERIPH_15_RST 1b - Forced Reset on PERIPH_15_RST
14 PERIPH_14_RST	Reset Control value for peripheral PERIPH_14_RST 0b - No forced reset on PERIPH_14_RST 1b - Forced Reset on PERIPH_14_RST
13 PERIPH_13_RST	Reset Control value for peripheral PERIPH_13_RST 0b - No forced reset on PERIPH_13_RST 1b - Forced Reset on PERIPH_13_RST
12 PERIPH_12_RST	Reset Control value for peripheral PERIPH_12_RST 0b - No forced reset on PERIPH_12_RST 1b - Forced Reset on PERIPH_12_RST
11 PERIPH_11_RST	Reset Control value for peripheral PERIPH_11_RST 0b - No forced reset on PERIPH_11_RST 1b - Forced Reset on PERIPH_11_RST
10 PERIPH_10_RST	Reset Control value for peripheral PERIPH_10_RST 0b - No forced reset on PERIPH_10_RST 1b - Forced Reset on PERIPH_10_RST
9 PERIPH_9_RST	Reset Control value for peripheral PERIPH_9_RST 0b - No forced reset on PERIPH_9_RST 1b - Forced Reset on PERIPH_9_RST
8 PERIPH_8_RST	Reset Control value for peripheral PERIPH_8_RST 0b - No forced reset on PERIPH_8_RST 1b - Forced Reset on PERIPH_8_RST
7 PERIPH_7_RST	Reset Control value for peripheral PERIPH_7_RST 0b - No forced reset on PERIPH_7_RST 1b - Forced Reset on PERIPH_7_RST
6	Reset Control value for peripheral PERIPH_6_RST

Table continues on the next page...

Table continued from the previous page...

Field	Function
PERIPH_6_RST	0b - No forced reset on PERIPH_6_RST 1b - Forced Reset on PERIPH_6_RST
5 PERIPH_5_RST	Reset Control value for peripheral PERIPH_5_RST 0b - No forced reset on PERIPH_5_RST 1b - Forced Reset on PERIPH_5_RST
4 PERIPH_4_RST	Reset Control value for peripheral PERIPH_4_RST 0b - No forced reset on PERIPH_4_RST 1b - Forced Reset on PERIPH_4_RST
3 PERIPH_3_RST	Reset Control value for peripheral PERIPH_3_RST 0b - No forced reset on PERIPH_3_RST 1b - Forced Reset on PERIPH_3_RST
2 PERIPH_2_RST	Reset Control value for peripheral PERIPH_2_RST 0b - No forced reset on PERIPH_2_RST 1b - Forced Reset on PERIPH_2_RST
1 PERIPH_1_RST	Reset Control value for peripheral PERIPH_1_RST 0b - No forced reset on PERIPH_1_RST 1b - Forced Reset on PERIPH_1_RST
0 PERIPH_0_RST	Reset Control value for peripheral PERIPH_0_RST 0b - No forced reset on PERIPH_0_RST 1b - Forced Reset on PERIPH_0_RST

29.7.11 Peripheral Reset (PRST1_0)

Offset

Register	Offset
PRST1_0	48h

Function

This register provides individual resets for various peripherals. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6 —	Reserved
5 —	Reserved
4 PERIPH_68_RST	Reset Control value for peripheral PERIPH_68_RST 0b - No forced reset on PERIPH_68_RST 1b - Forced Reset on PERIPH_68_RST
3 PERIPH_67_RST	Reset Control value for peripheral PERIPH_67_RST 0b - No forced reset on PERIPH_67_RST 1b - Forced Reset on PERIPH_67_RST
2 PERIPH_66_RST	Reset Control value for peripheral PERIPH_66_RST 0b - No forced reset on PERIPH_66_RST 1b - Forced Reset on PERIPH_66_RST
1 PERIPH_65_RST	Reset Control value for peripheral PERIPH_65_RST 0b - No forced reset on PERIPH_65_RST 1b - Forced Reset on PERIPH_65_RST
0 PERIPH_64_RST	Reset Control value for peripheral PERIPH_64_RST 0b - No forced reset on PERIPH_64_RST 1b - Forced Reset on PERIPH_64_RST

29.7.12 Peripheral Reset (PRST2_0)

Offset

Register	Offset
PRST2_0	50h

Function

This register provides individual resets for various peripherals. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e54-821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 PERIPH_136_RST	Reset Control value for peripheral PERIPH_136_RST 0b - No forced reset on PERIPH_136_RST 1b - Forced Reset on PERIPH_136_RST

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 PERIPH_135_RST	Reset Control value for peripheral PERIPH_135_RST 0b - No forced reset on PERIPH_135_RST 1b - Forced Reset on PERIPH_135_RST
6 PERIPH_134_RST	Reset Control value for peripheral PERIPH_134_RST 0b - No forced reset on PERIPH_134_RST 1b - Forced Reset on PERIPH_134_RST
5 PERIPH_133_RST	Reset Control value for peripheral PERIPH_133_RST 0b - No forced reset on PERIPH_133_RST 1b - Forced Reset on PERIPH_133_RST
4 PERIPH_132_RST	Reset Control value for peripheral PERIPH_132_RST 0b - No forced reset on PERIPH_132_RST 1b - Forced Reset on PERIPH_132_RST
3 PERIPH_131_RST	Reset Control value for peripheral PERIPH_131_RST 0b - No forced reset on PERIPH_131_RST 1b - Forced Reset on PERIPH_131_RST
2 PERIPH_130_RST	Reset Control value for peripheral PERIPH_130_RST 0b - No forced reset on PERIPH_130_RST 1b - Forced Reset on PERIPH_130_RST
1 PERIPH_129_RST	Reset Control value for peripheral PERIPH_129_RST 0b - No forced reset on PERIPH_129_RST 1b - Forced Reset on PERIPH_129_RST
0 PERIPH_128_RST	Reset Control value for peripheral PERIPH_128_RST 0b - No forced reset on PERIPH_128_RST 1b - Forced Reset on PERIPH_128_RST

29.7.13 Peripheral Reset (PRST3_0)

Offset

Register	Offset
PRST3_0	58h

Function

This register provides individual resets for various peripherals. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c

Table continued from the previous page...

Field	Function
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 PERIPH_200_RST	Reset Control value for peripheral PERIPH_200_RST 0b - No forced reset on PERIPH_200_RST 1b - Forced Reset on PERIPH_200_RST

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca4662e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
7 PERIPH_199_RST	Reset Control value for peripheral PERIPH_199_RST 0b - No forced reset on PERIPH_199_RST 1b - Forced Reset on PERIPH_199_RST
6 PERIPH_198_RST	Reset Control value for peripheral PERIPH_198_RST 0b - No forced reset on PERIPH_198_RST 1b - Forced Reset on PERIPH_198_RST
5 PERIPH_197_RST	Reset Control value for peripheral PERIPH_197_RST 0b - No forced reset on PERIPH_197_RST 1b - Forced Reset on PERIPH_197_RST
4 PERIPH_196_RST	Reset Control value for peripheral PERIPH_196_RST 0b - No forced reset on PERIPH_196_RST 1b - Forced Reset on PERIPH_196_RST
3 PERIPH_195_RST	Reset Control value for peripheral PERIPH_195_RST 0b - No forced reset on PERIPH_195_RST 1b - Forced Reset on PERIPH_195_RST
2 PERIPH_194_RST	Reset Control value for peripheral PERIPH_194_RST 0b - No forced reset on PERIPH_194_RST 1b - Forced Reset on PERIPH_194_RST
1 PERIPH_193_RST	Reset Control value for peripheral PERIPH_193_RST 0b - No forced reset on PERIPH_193_RST 1b - Forced Reset on PERIPH_193_RST
0 PERIPH_192_RST	Reset Control value for peripheral PERIPH_192_RST 0b - No forced reset on PERIPH_192_RST 1b - Forced Reset on PERIPH_192_RST

29.7.14 Peripheral Reset (PRST4_0)

Offset

Register	Offset
PRST4_0	60h

Function

This register provides individual resets for various peripherals. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c

Table continued from the previous page...

Field	Function
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 PERIPH_264_RST	Reset Control value for peripheral PERIPH_264_RST 0b - No forced reset on PERIPH_264_RST 1b - Forced Reset on PERIPH_264_RST

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca6e62e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
7 PERIPH_263_RST	Reset Control value for peripheral PERIPH_263_RST 0b - No forced reset on PERIPH_263_RST 1b - Forced Reset on PERIPH_263_RST
6 PERIPH_262_RST	Reset Control value for peripheral PERIPH_262_RST 0b - No forced reset on PERIPH_262_RST 1b - Forced Reset on PERIPH_262_RST
5 PERIPH_261_RST	Reset Control value for peripheral PERIPH_261_RST 0b - No forced reset on PERIPH_261_RST 1b - Forced Reset on PERIPH_261_RST
4 PERIPH_260_RST	Reset Control value for peripheral PERIPH_260_RST 0b - No forced reset on PERIPH_260_RST 1b - Forced Reset on PERIPH_260_RST
3 PERIPH_259_RST	Reset Control value for peripheral PERIPH_259_RST 0b - No forced reset on PERIPH_259_RST 1b - Forced Reset on PERIPH_259_RST
2 PERIPH_258_RST	Reset Control value for peripheral PERIPH_258_RST 0b - No forced reset on PERIPH_258_RST 1b - Forced Reset on PERIPH_258_RST
1 PERIPH_257_RST	Reset Control value for peripheral PERIPH_257_RST 0b - No forced reset on PERIPH_257_RST 1b - Forced Reset on PERIPH_257_RST
0 PERIPH_256_RST	Reset Control value for peripheral PERIPH_256_RST 0b - No forced reset on PERIPH_256_RST 1b - Forced Reset on PERIPH_256_RST

29.7.15 Peripheral Reset (PRST5_0)

Offset

Register	Offset
PRST5_0	68h

Function

This register provides individual resets for various peripherals. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 PERIPH_328_RST	Reset Control value for peripheral PERIPH_328_RST 0b - No forced reset on PERIPH_328_RST 1b - Forced Reset on PERIPH_328_RST

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca4662e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
7 PERIPH_327_RST	Reset Control value for peripheral PERIPH_327_RST 0b - No forced reset on PERIPH_327_RST 1b - Forced Reset on PERIPH_327_RST
6 PERIPH_326_RST	Reset Control value for peripheral PERIPH_326_RST 0b - No forced reset on PERIPH_326_RST 1b - Forced Reset on PERIPH_326_RST
5 PERIPH_325_RST	Reset Control value for peripheral PERIPH_325_RST 0b - No forced reset on PERIPH_325_RST 1b - Forced Reset on PERIPH_325_RST
4 PERIPH_324_RST	Reset Control value for peripheral PERIPH_324_RST 0b - No forced reset on PERIPH_324_RST 1b - Forced Reset on PERIPH_324_RST
3 PERIPH_323_RST	Reset Control value for peripheral PERIPH_323_RST 0b - No forced reset on PERIPH_323_RST 1b - Forced Reset on PERIPH_323_RST
2 PERIPH_322_RST	Reset Control value for peripheral PERIPH_322_RST 0b - No forced reset on PERIPH_322_RST 1b - Forced Reset on PERIPH_322_RST
1 PERIPH_321_RST	Reset Control value for peripheral PERIPH_321_RST 0b - No forced reset on PERIPH_321_RST 1b - Forced Reset on PERIPH_321_RST
0 PERIPH_320_RST	Reset Control value for peripheral PERIPH_320_RST 0b - No forced reset on PERIPH_320_RST 1b - Forced Reset on PERIPH_320_RST

29.7.16 Peripheral Reset (PRST6_0)

Offset

Register	Offset
PRST6_0	70h

Function

This register provides individual resets for various peripherals. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 PERIPH_392_RST	Reset Control value for peripheral PERIPH_392_RST 0b - No forced reset on PERIPH_392_RST 1b - Forced Reset on PERIPH_392_RST

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
7 PERIPH_391_RST	Reset Control value for peripheral PERIPH_391_RST 0b - No forced reset on PERIPH_391_RST 1b - Forced Reset on PERIPH_391_RST
6 PERIPH_390_RST	Reset Control value for peripheral PERIPH_390_RST 0b - No forced reset on PERIPH_390_RST 1b - Forced Reset on PERIPH_390_RST
5 PERIPH_389_RST	Reset Control value for peripheral PERIPH_389_RST 0b - No forced reset on PERIPH_389_RST 1b - Forced Reset on PERIPH_389_RST
4 PERIPH_388_RST	Reset Control value for peripheral PERIPH_388_RST 0b - No forced reset on PERIPH_388_RST 1b - Forced Reset on PERIPH_388_RST
3 PERIPH_387_RST	Reset Control value for peripheral PERIPH_387_RST 0b - No forced reset on PERIPH_387_RST 1b - Forced Reset on PERIPH_387_RST
2 PERIPH_386_RST	Reset Control value for peripheral PERIPH_386_RST 0b - No forced reset on PERIPH_386_RST 1b - Forced Reset on PERIPH_386_RST
1 PERIPH_385_RST	Reset Control value for peripheral PERIPH_385_RST 0b - No forced reset on PERIPH_385_RST 1b - Forced Reset on PERIPH_385_RST
0 PERIPH_384_RST	Reset Control value for peripheral PERIPH_384_RST 0b - No forced reset on PERIPH_384_RST 1b - Forced Reset on PERIPH_384_RST

29.7.17 Peripheral Reset (PRST7_0)

Offset

Register	Offset
PRST7_0	78h

Function

This register provides individual resets for various peripherals. It can be accessed in read/write, either in supervisor mode. It can be accessed in read-only in the user mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 PERIPH_456_RST	Reset Control value for peripheral PERIPH_456_RST 0b - No forced reset on PERIPH_456_RST 1b - Forced Reset on PERIPH_456_RST

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca4662e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
7 PERIPH_455_RST	Reset Control value for peripheral PERIPH_455_RST 0b - No forced reset on PERIPH_455_RST 1b - Forced Reset on PERIPH_455_RST
6 PERIPH_454_RST	Reset Control value for peripheral PERIPH_454_RST 0b - No forced reset on PERIPH_454_RST 1b - Forced Reset on PERIPH_454_RST
5 PERIPH_453_RST	Reset Control value for peripheral PERIPH_453_RST 0b - No forced reset on PERIPH_453_RST 1b - Forced Reset on PERIPH_453_RST
4 PERIPH_452_RST	Reset Control value for peripheral PERIPH_452_RST 0b - No forced reset on PERIPH_452_RST 1b - Forced Reset on PERIPH_452_RST
3 PERIPH_451_RST	Reset Control value for peripheral PERIPH_451_RST 0b - No forced reset on PERIPH_451_RST 1b - Forced Reset on PERIPH_451_RST
2 PERIPH_450_RST	Reset Control value for peripheral PERIPH_450_RST 0b - No forced reset on PERIPH_450_RST 1b - Forced Reset on PERIPH_450_RST
1 PERIPH_449_RST	Reset Control value for peripheral PERIPH_449_RST 0b - No forced reset on PERIPH_449_RST 1b - Forced Reset on PERIPH_449_RST
0 PERIPH_448_RST	Reset Control value for peripheral PERIPH_448_RST 0b - No forced reset on PERIPH_448_RST 1b - Forced Reset on PERIPH_448_RST

29.7.18 Peripheral Reset Status Register (PSTAT0_0)

Offset

Register	Offset
PSTAT0_0	140h

Function

This register provides the current reset status for various peripherals. It can be accessed in read-only in all modes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31 PERIPH_31_STAT	Reset Status for peripheral PERIPH_31_STAT 0b - Peripheral PERIPH_31_STAT is not in reset 1b - Peripheral PERIPH_31_STAT is in reset
30 PERIPH_30_STAT	Reset Status for peripheral PERIPH_30_STAT 0b - Peripheral PERIPH_30_STAT is not in reset 1b - Peripheral PERIPH_30_STAT is in reset
29 PERIPH_29_STAT	Reset Status for peripheral PERIPH_29_STAT 0b - Peripheral PERIPH_29_STAT is not in reset 1b - Peripheral PERIPH_29_STAT is in reset
28 PERIPH_28_STAT	Reset Status for peripheral PERIPH_28_STAT 0b - Peripheral PERIPH_28_STAT is not in reset 1b - Peripheral PERIPH_28_STAT is in reset
27 PERIPH_27_STAT	Reset Status for peripheral PERIPH_27_STAT 0b - Peripheral PERIPH_27_STAT is not in reset 1b - Peripheral PERIPH_27_STAT is in reset
26 PERIPH_26_STAT	Reset Status for peripheral PERIPH_26_STAT 0b - Peripheral PERIPH_26_STAT is not in reset 1b - Peripheral PERIPH_26_STAT is in reset
25 PERIPH_25_STAT	Reset Status for peripheral PERIPH_25_STAT 0b - Peripheral PERIPH_25_STAT is not in reset 1b - Peripheral PERIPH_25_STAT is in reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 PERIPH_24_STAT	Reset Status for peripheral PERIPH_24_STAT 0b - Peripheral PERIPH_24_STAT is not in reset 1b - Peripheral PERIPH_24_STAT is in reset
23 PERIPH_23_STAT	Reset Status for peripheral PERIPH_23_STAT 0b - Peripheral PERIPH_23_STAT is not in reset 1b - Peripheral PERIPH_23_STAT is in reset
22 PERIPH_22_STAT	Reset Status for peripheral PERIPH_22_STAT 0b - Peripheral PERIPH_22_STAT is not in reset 1b - Peripheral PERIPH_22_STAT is in reset
21 PERIPH_21_STAT	Reset Status for peripheral PERIPH_21_STAT 0b - Peripheral PERIPH_21_STAT is not in reset 1b - Peripheral PERIPH_21_STAT is in reset
20 PERIPH_20_STAT	Reset Status for peripheral PERIPH_20_STAT 0b - Peripheral PERIPH_20_STAT is not in reset 1b - Peripheral PERIPH_20_STAT is in reset
19 PERIPH_19_STAT	Reset Status for peripheral PERIPH_19_STAT 0b - Peripheral PERIPH_19_STAT is not in reset 1b - Peripheral PERIPH_19_STAT is in reset
18 PERIPH_18_STAT	Reset Status for peripheral PERIPH_18_STAT 0b - Peripheral PERIPH_18_STAT is not in reset 1b - Peripheral PERIPH_18_STAT is in reset
17 PERIPH_17_STAT	Reset Status for peripheral PERIPH_17_STAT 0b - Peripheral PERIPH_17_STAT is not in reset 1b - Peripheral PERIPH_17_STAT is in reset
16 PERIPH_16_STAT	Reset Status for peripheral PERIPH_16_STAT 0b - Peripheral PERIPH_16_STAT is not in reset 1b - Peripheral PERIPH_16_STAT is in reset
15 PERIPH_15_STAT	Reset Status for peripheral PERIPH_15_STAT 0b - Peripheral PERIPH_15_STAT is not in reset 1b - Peripheral PERIPH_15_STAT is in reset
14	Reset Status for peripheral PERIPH_14_STAT

Table continues on the next page...

Table continued from the previous page...

Field	Function
PERIPH_14_STAT	0b - Peripheral PERIPH_14_STAT is not in reset 1b - Peripheral PERIPH_14_STAT is in reset
13 PERIPH_13_STAT	Reset Status for peripheral PERIPH_13_STAT 0b - Peripheral PERIPH_13_STAT is not in reset 1b - Peripheral PERIPH_13_STAT is in reset
12 PERIPH_12_STAT	Reset Status for peripheral PERIPH_12_STAT 0b - Peripheral PERIPH_12_STAT is not in reset 1b - Peripheral PERIPH_12_STAT is in reset
11 PERIPH_11_STAT	Reset Status for peripheral PERIPH_11_STAT 0b - Peripheral PERIPH_11_STAT is not in reset 1b - Peripheral PERIPH_11_STAT is in reset
10 PERIPH_10_STAT	Reset Status for peripheral PERIPH_10_STAT 0b - Peripheral PERIPH_10_STAT is not in reset 1b - Peripheral PERIPH_10_STAT is in reset
9 PERIPH_9_STAT	Reset Status for peripheral PERIPH_9_STAT 0b - Peripheral PERIPH_9_STAT is not in reset 1b - Peripheral PERIPH_9_STAT is in reset
8 PERIPH_8_STAT	Reset Status for peripheral PERIPH_8_STAT 0b - Peripheral PERIPH_8_STAT is not in reset 1b - Peripheral PERIPH_8_STAT is in reset
7 PERIPH_7_STAT	Reset Status for peripheral PERIPH_7_STAT 0b - Peripheral PERIPH_7_STAT is not in reset 1b - Peripheral PERIPH_7_STAT is in reset
6 PERIPH_6_STAT	Reset Status for peripheral PERIPH_6_STAT 0b - Peripheral PERIPH_6_STAT is not in reset 1b - Peripheral PERIPH_6_STAT is in reset
5 PERIPH_5_STAT	Reset Status for peripheral PERIPH_5_STAT 0b - Peripheral PERIPH_5_STAT is not in reset 1b - Peripheral PERIPH_5_STAT is in reset
4 PERIPH_4_STAT	Reset Status for peripheral PERIPH_4_STAT 0b - Peripheral PERIPH_4_STAT is not in reset 1b - Peripheral PERIPH_4_STAT is in reset

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 PERIPH_3_STAT	Reset Status for peripheral PERIPH_3_STAT 0b - Peripheral PERIPH_3_STAT is not in reset 1b - Peripheral PERIPH_3_STAT is in reset
2 PERIPH_2_STAT	Reset Status for peripheral PERIPH_2_STAT 0b - Peripheral PERIPH_2_STAT is not in reset 1b - Peripheral PERIPH_2_STAT is in reset
1 PERIPH_1_STAT	Reset Status for peripheral PERIPH_1_STAT 0b - Peripheral PERIPH_1_STAT is not in reset 1b - Peripheral PERIPH_1_STAT is in reset
0 PERIPH_0_STAT	Reset Status for peripheral PERIPH_0_STAT 0b - Peripheral PERIPH_0_STAT is not in reset 1b - Peripheral PERIPH_0_STAT is in reset

29.7.19 Peripheral Reset Status Register (PSTAT1_0)

Offset

Register	Offset
PSTAT1_0	148h

Function

This register provides the current reset status for various peripherals. It can be accessed in read-only in all modes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEC Automotive Technology Co., Ltd.
c8c46265-4621-4ca8-b251-390d3ab624c1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 PERIPH_68_STAT	Reset Status for peripheral PERIPH_68_STAT 0b - Peripheral PERIPH_68_STAT is not in reset 1b - Peripheral PERIPH_68_STAT is in reset
3	Reset Status for peripheral PERIPH_67_STAT

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca43e5e2-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
PERIPH_67_STAT	0b - Peripheral PERIPH_67_STAT is not in reset 1b - Peripheral PERIPH_67_STAT is in reset
2 PERIPH_66_STAT	Reset Status for peripheral PERIPH_66_STAT 0b - Peripheral PERIPH_66_STAT is not in reset 1b - Peripheral PERIPH_66_STAT is in reset
1 PERIPH_65_STAT	Reset Status for peripheral PERIPH_65_STAT 0b - Peripheral PERIPH_65_STAT is not in reset 1b - Peripheral PERIPH_65_STAT is in reset
0 PERIPH_64_STAT	Reset Status for peripheral PERIPH_64_STAT 0b - Peripheral PERIPH_64_STAT is not in reset 1b - Peripheral PERIPH_64_STAT is in reset

29.7.20 Peripheral Reset Status Register (PSTAT2_0)

Offset

Register	Offset
PSTAT2_0	150h

Function

This register provides the current reset status for various peripherals. It can be accessed in read-only in all modes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4626a-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 PERIPH_136_STAT	Reset Status for peripheral PERIPH_136_STAT 0b - Peripheral PERIPH_136_STAT is not in reset 1b - Peripheral PERIPH_136_STAT is in reset
7 PERIPH_135_STAT	Reset Status for peripheral PERIPH_135_STAT 0b - Peripheral PERIPH_135_STAT is not in reset 1b - Peripheral PERIPH_135_STAT is in reset
6 PERIPH_134_STAT	Reset Status for peripheral PERIPH_134_STAT 0b - Peripheral PERIPH_134_STAT is not in reset 1b - Peripheral PERIPH_134_STAT is in reset
5 PERIPH_133_STAT	Reset Status for peripheral PERIPH_133_STAT 0b - Peripheral PERIPH_133_STAT is not in reset 1b - Peripheral PERIPH_133_STAT is in reset
4	Reset Status for peripheral PERIPH_132_STAT

Table continues on the next page...

Table continued from the previous page...

Field	Function
PERIPH_132_STAT	0b - Peripheral PERIPH_132_STAT is not in reset 1b - Peripheral PERIPH_132_STAT is in reset
3 PERIPH_131_STAT	Reset Status for peripheral PERIPH_131_STAT 0b - Peripheral PERIPH_131_STAT is not in reset 1b - Peripheral PERIPH_131_STAT is in reset
2 PERIPH_130_STAT	Reset Status for peripheral PERIPH_130_STAT 0b - Peripheral PERIPH_130_STAT is not in reset 1b - Peripheral PERIPH_130_STAT is in reset
1 PERIPH_129_STAT	Reset Status for peripheral PERIPH_129_STAT 0b - Peripheral PERIPH_129_STAT is not in reset 1b - Peripheral PERIPH_129_STAT is in reset
0 PERIPH_128_STAT	Reset Status for peripheral PERIPH_128_STAT 0b - Peripheral PERIPH_128_STAT is not in reset 1b - Peripheral PERIPH_128_STAT is in reset

29.7.21 Peripheral Reset Status Register (PSTAT3_0)

Offset

Register	Offset
PSTAT3_0	158h

Function

This register provides the current reset status for various peripherals. It can be accessed in read-only in all modes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 PERIPH_200_STAT	Reset Status for peripheral PERIPH_200_STAT 0b - Peripheral PERIPH_200_STAT is not in reset 1b - Peripheral PERIPH_200_STAT is in reset
7 PERIPH_199_STAT	Reset Status for peripheral PERIPH_199_STAT 0b - Peripheral PERIPH_199_STAT is not in reset 1b - Peripheral PERIPH_199_STAT is in reset
6 PERIPH_198_STAT	Reset Status for peripheral PERIPH_198_STAT 0b - Peripheral PERIPH_198_STAT is not in reset 1b - Peripheral PERIPH_198_STAT is in reset
5 PERIPH_197_STAT	Reset Status for peripheral PERIPH_197_STAT 0b - Peripheral PERIPH_197_STAT is not in reset 1b - Peripheral PERIPH_197_STAT is in reset
4	Reset Status for peripheral PERIPH_196_STAT

Table continues on the next page...

Table continued from the previous page...

Field	Function
PERIPH_196_STAT	0b - Peripheral PERIPH_196_STAT is not in reset 1b - Peripheral PERIPH_196_STAT is in reset
3 PERIPH_195_STAT	Reset Status for peripheral PERIPH_195_STAT 0b - Peripheral PERIPH_195_STAT is not in reset 1b - Peripheral PERIPH_195_STAT is in reset
2 PERIPH_194_STAT	Reset Status for peripheral PERIPH_194_STAT 0b - Peripheral PERIPH_194_STAT is not in reset 1b - Peripheral PERIPH_194_STAT is in reset
1 PERIPH_193_STAT	Reset Status for peripheral PERIPH_193_STAT 0b - Peripheral PERIPH_193_STAT is not in reset 1b - Peripheral PERIPH_193_STAT is in reset
0 PERIPH_192_STAT	Reset Status for peripheral PERIPH_192_STAT 0b - Peripheral PERIPH_192_STAT is not in reset 1b - Peripheral PERIPH_192_STAT is in reset

29.7.22 Peripheral Reset Status Register (PSTAT4_0)

Offset

Register	Offset
PSTAT4_0	160h

Function

This register provides the current reset status for various peripherals. It can be accessed in read-only in all modes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c6462e5-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 PERIPH_264_STAT	Reset Status for peripheral PERIPH_264_STAT 0b - Peripheral PERIPH_264_STAT is not in reset 1b - Peripheral PERIPH_264_STAT is in reset
7 PERIPH_263_STAT	Reset Status for peripheral PERIPH_263_STAT 0b - Peripheral PERIPH_263_STAT is not in reset 1b - Peripheral PERIPH_263_STAT is in reset
6 PERIPH_262_STAT	Reset Status for peripheral PERIPH_262_STAT 0b - Peripheral PERIPH_262_STAT is not in reset 1b - Peripheral PERIPH_262_STAT is in reset
5 PERIPH_261_STAT	Reset Status for peripheral PERIPH_261_STAT 0b - Peripheral PERIPH_261_STAT is not in reset 1b - Peripheral PERIPH_261_STAT is in reset
4	Reset Status for peripheral PERIPH_260_STAT

Table continues on the next page...

Table continued from the previous page...

Field	Function
PERIPH_260_STAT	0b - Peripheral PERIPH_260_STAT is not in reset 1b - Peripheral PERIPH_260_STAT is in reset
3 PERIPH_259_STAT	Reset Status for peripheral PERIPH_259_STAT 0b - Peripheral PERIPH_259_STAT is not in reset 1b - Peripheral PERIPH_259_STAT is in reset
2 PERIPH_258_STAT	Reset Status for peripheral PERIPH_258_STAT 0b - Peripheral PERIPH_258_STAT is not in reset 1b - Peripheral PERIPH_258_STAT is in reset
1 PERIPH_257_STAT	Reset Status for peripheral PERIPH_257_STAT 0b - Peripheral PERIPH_257_STAT is not in reset 1b - Peripheral PERIPH_257_STAT is in reset
0 PERIPH_256_STAT	Reset Status for peripheral PERIPH_256_STAT 0b - Peripheral PERIPH_256_STAT is not in reset 1b - Peripheral PERIPH_256_STAT is in reset

29.7.23 Peripheral Reset Status Register (PSTAT5_0)

Offset

Register	Offset
PSTAT5_0	168h

Function

This register provides the current reset status for various peripherals. It can be accessed in read-only in all modes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 PERIPH_328_STAT	Reset Status for peripheral PERIPH_328_STAT 0b - Peripheral PERIPH_328_STAT is not in reset 1b - Peripheral PERIPH_328_STAT is in reset
7 PERIPH_327_STAT	Reset Status for peripheral PERIPH_327_STAT 0b - Peripheral PERIPH_327_STAT is not in reset 1b - Peripheral PERIPH_327_STAT is in reset
6 PERIPH_326_STAT	Reset Status for peripheral PERIPH_326_STAT 0b - Peripheral PERIPH_326_STAT is not in reset 1b - Peripheral PERIPH_326_STAT is in reset
5 PERIPH_325_STAT	Reset Status for peripheral PERIPH_325_STAT 0b - Peripheral PERIPH_325_STAT is not in reset 1b - Peripheral PERIPH_325_STAT is in reset
4	Reset Status for peripheral PERIPH_324_STAT

Table continues on the next page...

Table continued from the previous page...

Field	Function
PERIPH_324_STAT	0b - Peripheral PERIPH_324_STAT is not in reset 1b - Peripheral PERIPH_324_STAT is in reset
3 PERIPH_323_STAT	Reset Status for peripheral PERIPH_323_STAT 0b - Peripheral PERIPH_323_STAT is not in reset 1b - Peripheral PERIPH_323_STAT is in reset
2 PERIPH_322_STAT	Reset Status for peripheral PERIPH_322_STAT 0b - Peripheral PERIPH_322_STAT is not in reset 1b - Peripheral PERIPH_322_STAT is in reset
1 PERIPH_321_STAT	Reset Status for peripheral PERIPH_321_STAT 0b - Peripheral PERIPH_321_STAT is not in reset 1b - Peripheral PERIPH_321_STAT is in reset
0 PERIPH_320_STAT	Reset Status for peripheral PERIPH_320_STAT 0b - Peripheral PERIPH_320_STAT is not in reset 1b - Peripheral PERIPH_320_STAT is in reset

29.7.24 Peripheral Reset Status Register (PSTAT6_0)

Offset

Register	Offset
PSTAT6_0	170h

Function

This register provides the current reset status for various peripherals. It can be accessed in read-only in all modes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Provided under NDA
COMPANY PROPRIETARY
Bejing CHJ Automotive Technology Co., Ltd.
c8c46e25-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 PERIPH_392_STAT	Reset Status for peripheral PERIPH_392_STAT 0b - Peripheral PERIPH_392_STAT is not in reset 1b - Peripheral PERIPH_392_STAT is in reset
7 PERIPH_391_STAT	Reset Status for peripheral PERIPH_391_STAT 0b - Peripheral PERIPH_391_STAT is not in reset 1b - Peripheral PERIPH_391_STAT is in reset
6 PERIPH_390_STAT	Reset Status for peripheral PERIPH_390_STAT 0b - Peripheral PERIPH_390_STAT is not in reset 1b - Peripheral PERIPH_390_STAT is in reset
5 PERIPH_389_STAT	Reset Status for peripheral PERIPH_389_STAT 0b - Peripheral PERIPH_389_STAT is not in reset 1b - Peripheral PERIPH_389_STAT is in reset
4	Reset Status for peripheral PERIPH_388_STAT

Table continues on the next page...

Table continued from the previous page...

Field	Function
PERIPH_388_STAT	0b - Peripheral PERIPH_388_STAT is not in reset 1b - Peripheral PERIPH_388_STAT is in reset
3 PERIPH_387_STAT	Reset Status for peripheral PERIPH_387_STAT 0b - Peripheral PERIPH_387_STAT is not in reset 1b - Peripheral PERIPH_387_STAT is in reset
2 PERIPH_386_STAT	Reset Status for peripheral PERIPH_386_STAT 0b - Peripheral PERIPH_386_STAT is not in reset 1b - Peripheral PERIPH_386_STAT is in reset
1 PERIPH_385_STAT	Reset Status for peripheral PERIPH_385_STAT 0b - Peripheral PERIPH_385_STAT is not in reset 1b - Peripheral PERIPH_385_STAT is in reset
0 PERIPH_384_STAT	Reset Status for peripheral PERIPH_384_STAT 0b - Peripheral PERIPH_384_STAT is not in reset 1b - Peripheral PERIPH_384_STAT is in reset

29.7.25 Peripheral Reset Status Register (PSTAT7_0)

Offset

Register	Offset
PSTAT7_0	178h

Function

This register provides the current reset status for various peripherals. It can be accessed in read-only in all modes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...	PERIPH_...
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e54-821f-48a8-b251-390d3ab624c1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 PERIPH_456_STAT	Reset Status for peripheral PERIPH_456_STAT 0b - Peripheral PERIPH_456_STAT is not in reset 1b - Peripheral PERIPH_456_STAT is in reset
7 PERIPH_455_STAT	Reset Status for peripheral PERIPH_455_STAT 0b - Peripheral PERIPH_455_STAT is not in reset 1b - Peripheral PERIPH_455_STAT is in reset
6 PERIPH_454_STAT	Reset Status for peripheral PERIPH_454_STAT 0b - Peripheral PERIPH_454_STAT is not in reset 1b - Peripheral PERIPH_454_STAT is in reset
5 PERIPH_453_STAT	Reset Status for peripheral PERIPH_453_STAT 0b - Peripheral PERIPH_453_STAT is not in reset 1b - Peripheral PERIPH_453_STAT is in reset
4	Reset Status for peripheral PERIPH_452_STAT

Table continues on the next page...

Table continued from the previous page...

Field	Function
PERIPH_452_STAT	0b - Peripheral PERIPH_452_STAT is not in reset 1b - Peripheral PERIPH_452_STAT is in reset
3 PERIPH_451_STAT	Reset Status for peripheral PERIPH_451_STAT 0b - Peripheral PERIPH_451_STAT is not in reset 1b - Peripheral PERIPH_451_STAT is in reset
2 PERIPH_450_STAT	Reset Status for peripheral PERIPH_450_STAT 0b - Peripheral PERIPH_450_STAT is not in reset 1b - Peripheral PERIPH_450_STAT is in reset
1 PERIPH_449_STAT	Reset Status for peripheral PERIPH_449_STAT 0b - Peripheral PERIPH_449_STAT is not in reset 1b - Peripheral PERIPH_449_STAT is in reset
0 PERIPH_448_STAT	Reset Status for peripheral PERIPH_448_STAT 0b - Peripheral PERIPH_448_STAT is not in reset 1b - Peripheral PERIPH_448_STAT is in reset

29.8 Functional description

29.8.1 Reset state machine

The main role of MC_RGM is the generation of the reset sequence that ensures that the correct parts of the chip are reset based on the reset source event.

For each reset event, immediately after it is captured by the MC_RGM, the following takes place:

1. The corresponding reset event status bit is set in the MC_RGM_DES and MC_RGM_FES registers.
2. The pins are put into their default states
3. The RESET_B pin is asserted.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-2a8-b251-390d3ab624c1

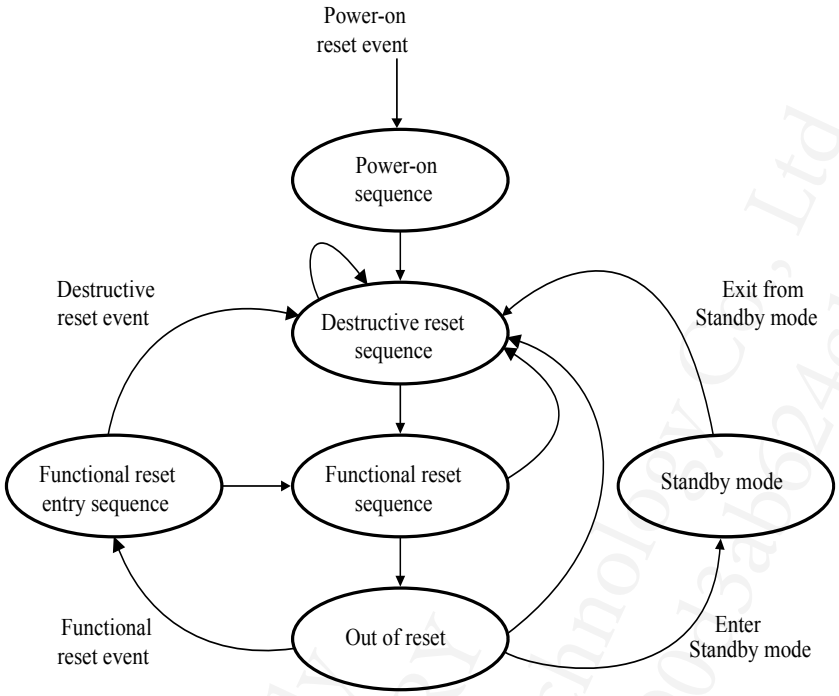


Figure 135. Reset sequence

29.8.1.1 Power-on reset sequence

A reset is always generated when the power-on reset source is asserted, and it has priority over all other reset sources. Such a power-on reset forces the reset state machine to enter the power-on sequence resulting in the assertion of all reset signals. The reset state machine starts progressing when the following two conditions are verified:

1. All the power-on reset events are cleared
2. The MC_RGM's clock source (the FIRC) has started up and stabilized

If a power-on reset event has occurred, the DES[F_POR] bit is set.

The power-on reset cannot be demoted by the software.

29.8.1.2 Destructive reset sequence

The 'Destructive reset sequence' is comprised of a number of phases, where DEST0 is the first phase and is followed by DEST1 and so forth.

This phase is entered immediately from any phase on a power-on, standby reset sequence, or enabled destructive reset event . A destructive reset counter starts immediately on entry in the DEST0 phase . The DEST0 state is exited to the DEST1 state on the rising edge of ipg_clk immediately after all of the following conditions have been established:

- the DEST0 duration time has expired
- all dest_rst inputs are deasserted

The DEST0 state is immediately exited to the power-on state after if a power-on reset event occurrs

The reset state machine exits the destructive reset sequence and enters the functional reset sequence when:

- All the destructive reset events are cleared.
- All the processes that take place during the destructive reset sequence have completed. For details on completed destructive reset sequence, see "Reset process sequence" section in the Reset chapter.

- The 'destructive' reset escalator counter has not reached the value in the DRET field of the RGM_DRET resister.

29.8.1.3 Functional reset sequence

There are two categories of functional reset sequence, the functional reset entry sequence and the functional reset exit sequence.

29.8.1.3.1 Functional reset entry sequence

The functional reset entry sequence is only entered when a functional reset event occurs during the idle phase.

If a functional reset event occurs during an ongoing reset sequence, the corresponding event status flag is set, and the RESET_B pin is asserted per the reset event's configuration. However, the reset sequence is not influenced, and it continues to progress without interruption.

No reset is asserted during the functional reset entry sequence.

The functional reset entry sequence is exited to the DEST0 on the next clock rising edge if a destructive reset event has occurred. The sequence enters the power up sequence if a POR event occurs

In all other cases, the sequence exits to the first stage of functional reset exit sequence.

29.8.1.3.2 Functional reset exit sequence

The functional reset exit sequence is entered either on exit from the destructive reset sequence or on completion of the functional reset entry sequence. The reset state machine exits this sequence and enters the idle phase on verification of the following:

- All the functional reset events are cleared.
- All the processes that take place during the destructive reset sequence have completed. For details on completed destructive reset sequence, see "Reset process sequence" section in the Reset chapter.

If a functional reset event occurs during an ongoing reset sequence, the corresponding event status flag is set, and the RESET_B pin is asserted per the reset event's configuration. However, the reset sequence is not influenced, and it continues to progress without interruption.

29.8.1.4 Idle phase

This is the final phase and is entered on exit from the functional reset exit sequence . When this phase is reached, MC_RGM releases control of the system to the platform and waits for the new reset events that can trigger a reset sequence.

29.8.2 Destructive resets

A destructive reset indicates that an event has occurred after which critical register or memory content can no longer be guaranteed.

The status flag associated with a given destructive reset event (Destructive Event Status Register (DES)) is set when the destructive reset is asserted and the power-on reset is not asserted. It is possible for multiple status bits to be set simultaneously and the software determines which reset source is the most critical for the application.

The low-voltage detector threshold ensures that when the reset corresponding to the core supply low-voltage detect is enabled, the supply is sufficient to have the destructive event correctly propagated through the digital logic. Therefore, if a given destructive reset is enabled, MC_RGM ensures that the associated reset event is correctly triggered to the full system.

An enabled destructive reset triggers a reset sequence starting from the beginning of DEST0.

29.8.3 External reset

MC_RGM manages the external reset coming from RESET_B. The detection of a falling edge on RESET_B starts the reset sequence from the beginning of the functional reset entry sequence.

The status flag associated with the external reset falling edge event (the RGM_FES[F_EXR] bit) is set when the external reset is asserted and the power-on reset is not asserted.

MC_RGM asserts the external reset if the reset sequence is triggered by one of the following:

- A power-on reset
- A destructive reset event
- A functional reset event

In this case, external reset is asserted until all conditions for the exiting functional reset sequence have been met, with the exception of the RESET_B assertion check

29.8.4 Functional resets

A functional reset indicates that an event has occurred after which it can be guaranteed that critical register and memory content is still intact.

The status flag associated with a given functional reset event (Functional /External Reset Status Register (FES)) is set when the functional reset is asserted and the power-on reset is not asserted. It is possible for multiple status bits to be set simultaneously and the software determines which reset source is the most critical for the application.

An enabled functional reset triggers a reset sequence starting from the beginning of the functional reset entry sequence.

29.8.5 External reset assertion control

The software indicates to the MC_RGM that the RESET_B is to be asserted by writing to the **ERASSERT** bit in the RGM_ERCTRL register. When this bit is set by the software, RESET_B gets asserted. Setting of this field does not impact the reset sequence in any way.

An example where the **ERASSERT** bit could be set by the software is when entering the self test sequence, during which RESET_B is to be asserted. This indicates the chip is not available in the functional mode although a reset sequence is not in progress. The deassertion of RESET_B is not controlled by the software. Instead, the RESET_B pin remains asserted until the next time the chip exits a reset sequence.

ERASSERT bit is also cleared during the reset sequence.

29.8.6 Functional reset escalation

Functional reset escalation can be used to generate a destructive reset if a number of functional resets is occurred between software writes to the RGM_FRET register. This function is enabled by writing a non-zero value to the FRET field of this register.

After the functional reset escalation is enabled, MC_RGM increases a counter on each functional reset that causes a reset sequence to be initiated (which means, entrance into FUNC0 from the IDLE phase). This counter is cleared on a write of any value to the RGM_FRET register and on any power-on or destructive reset. If the counter reaches the value in the FRET field of the RGM_FRET register, MC_RGM asserts a destructive reset.

The following figure shows the functional reset escalation counter.

Provided by IHS Markit
COMPANY CONFIDENTIAL
Beijing CIIC Automotive Technology Co., Ltd.
c8c462e5-4821-4fa8-b21c-30d8ab024c1

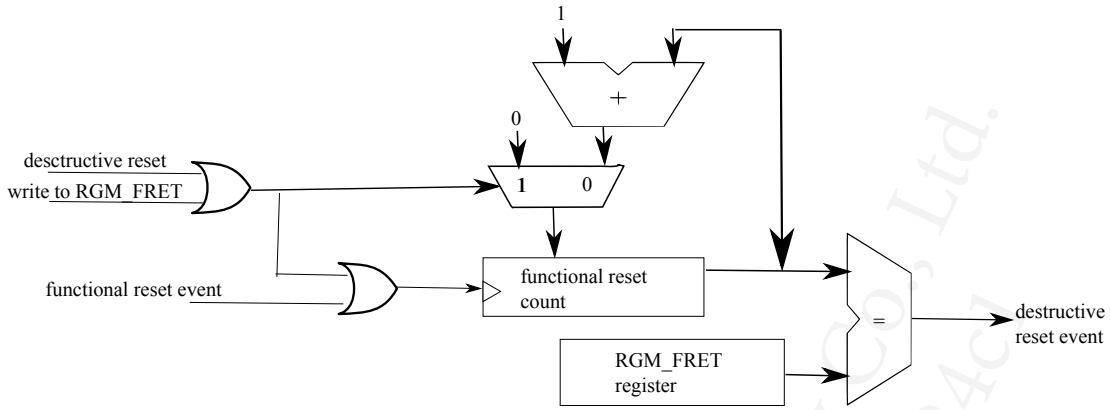


Figure 136. Functional reset escalation counter

29.8.7 Destructive reset escalation

Destructive reset escalation can be used to keep the chip in the reset state until the power-on triggers a reset sequence if a number of destructive resets are occurred between software writes to the RGM_DRET register. This function is enabled by writing a non-zero value to the DRET field of this register.

After destructive reset escalation is enabled, MC_RGM increases a counter on each destructive reset that is enabled to increment the escalator. . This causes a reset sequence to be initiated (that is, entrance into DEST0 from the idle phase) or an ongoing reset sequence to restart (that is, entrance into DEST0 from any other reset phase). This counter is cleared on a write of any value to the RGM_DRET register and on any power-on reset. If the counter reaches the value in the DRET field of the RGM_DRET register, MC_RGM enters reset DEST0 and stays there until the next power-on reset occurs .

The following figure shows the destructive reset escalation counter.

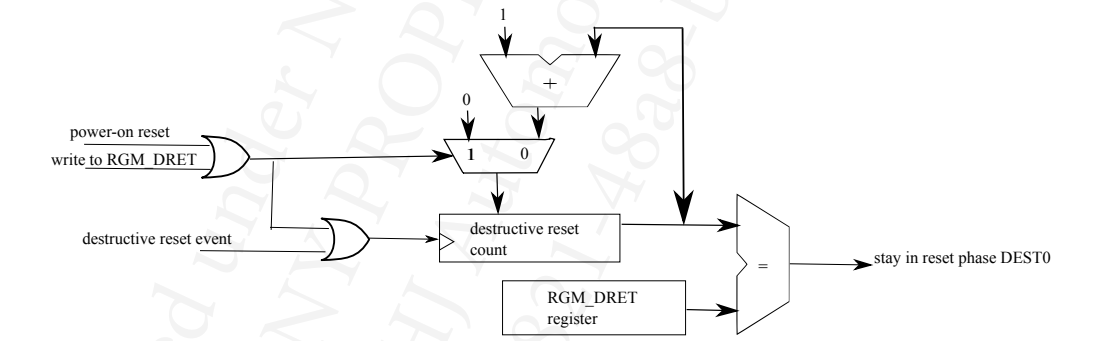


Figure 137. Destructive reset escalation counter

29.8.8 Individual peripheral resets

A peripheral can be reset individually without resetting the rest of the chip by setting the appropriate bit in the RGM_PRSTn registers. However, to prevent unexpected behavior by the chip, the following sequence must be observed by software.

1. Disable the peripheral that is to be reset through MC_ME (see the MC_ME chapter for details).
2. Wait for the mode change to complete.
3. Set the bit corresponding to the peripheral in the RGM_PRSTn registers.
4. Wait until the bit corresponding to the peripheral in the RGM_PSTATn registers isset.
5. Clear the bit corresponding to the peripheral in the RGM_PRSTn registers.

6. Wait until the bit corresponding to the peripheral in the RGM_PSTATn registers is cleared.

The peripheral can be kept in reset indefinitely by not executing steps 5 and 6. After the completion of step 6, the peripheral can be enabled again through MC_ME.

NOTE

For any limitations on this feature, see the chip-specific MC_RGM information.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 30
Boot

30.1 Introduction

This chapter describes the system boot sequence and provides details about boot options.

When the hardware reset sequence completes, the only processor core available is the Cortex-M7 core in the HSE_H subsystem, referred to as the HSE_H core. The HSE_H core begins executing firmware from the HSE_H ROM module that contains the BootROM firmware. The BootROM firmware manages the boot sequence until it passes control to other code, based on the type of boot:

- In a *non-secure boot*, firmware passes control to customer software that runs on a processor core that is outside the HSE_H subsystem.
- In a *secure boot*, firmware passes control to the HSE_H firmware running on the HSE_H core.

30.2 Features

BootROM boots both customer applications and HSE_H firmware. For a secure boot, it creates the Root of Chain of Trust. BootROM supports the following features:

- Boot from external flash memory
- Secure and Non-Secure boot modes
- Selection of boot core for customer applications
- Serial download
- Execution of Device Configuration Data (DCD)
- Configuration and start of customer-provided self-test data
- Advance of chip life cycle state
- Debug challenge/response authentication
- Boot from Standby/Low Power mode
- Boot from a backup program image

30.3 Boot modes

This chip supports two boot modes:

- Serial Boot mode (from UART, CAN, or Ethernet) is typically used for development and factory line programming. You can disable serial boot mode by blowing the DIS_SER_BOOT fuse.
- Boot from external flash memory (from QuadSPI flash, SD, or MMC) is typically used in production devices.

At reset, BootROM code for the HSE_H core configures the boot mode, including memory type, speed, and so on. It does this from either RCON pins or fuses, based on the FUSE_SEL fuse status.

Combinations of the following inputs control the boot mode:

- Boot mode pins (BMODE1 and BMODE2)
- FUSE_SEL fuse (inhibits the use of RCON and forces fuse-based boot)
- Life cycle state (blocks some boot modes that are valid only for specific life cycle states)

The boot mode pin values are latched before deassertion of the RESET_B signal, enabling selection of the required boot mode externally from the chip.

Provided under NDA only
COMPANY PROPRIETARY
Copyright (c) 2021 Automotive Technology Co., Ltd.
8216e621-21-48a8-b251-390a8ab624c

The following table shows the supported boot modes for various combinations of inputs.

Table 126. Supported boot modes

Life cycle state	FUSE_SEL	BMODE1	BMODE2	Mode	Use case
MCU_PROD or CUST_DEL or OEM_PROD or IN_FIELD	0	0	0	Serial boot + XOSC in Differential mode	Production configuration for an unprogrammed chip
	0	0	1	Serial boot + XOSC in Crystal mode or Bypass mode	Production configuration for an unprogrammed chip
	0	1	0	Boot from RCON	Development configuration
	1	0	0	Boot from fuses	Production configuration for a programmed chip
	1	0	1	Boot from fuses	Production configuration for a programmed chip
	1	1	0	Serial boot	Debug of a programmed chip
	X	1	1	Reserved	Reserved

NOTE

- The XOSC Crystal mode and XOSC Bypass mode configurations are valid only if the XOSC_VALID fuse bit is not blown. See the attached fuse map spreadsheet.
- Boot from RCON and boot from fuses correspond to boot from external flash memory. BOOT_CFG1[7:5]/ RCON[7:5] values determine the source for external flash memory.

30.3.1 Serial Boot mode

Serial Boot mode enables application code to be downloaded to SRAM. There are many possible uses for serial boot, but the two primary use cases are:

- End-of-line flash memory and fuse programming
- Recovery of chips that fail to boot correctly (nonresponsive modules)

NOTE

You can disable serial boot by blowing the DIS_SER_BOOT (BOOT_CFG2[3]) fuse.

30.3.1.1 End-of-line programming

In a new chip on the production line, the flash memory is not programmed and all fuses are not blown. To make the device usable, the flash memory must be programmed and the fuses blown with the desired configuration. Serial Boot mode enables download of the initial customer application and data to an unprogrammed device.

30.3.1.2 Non-responsive modules

Modules that fail to boot—due to software programming errors, flash memory failure, or PCB failure during production, for example—indicate that a chip is not responsive. Serial Boot mode provides a mechanism to download executable recovery software to SRAM. For example, the downloaded executable may reprogram the flash memory or may execute diagnostic code to determine the fault cause.

30.3.2 Boot from fuses and RCON

BootROM supports booting from multiple interfaces. To configure the interfaces to meet application requirements, BootROM requires a mechanism through which information can be passed to BootROM. BootROM supports two such mechanisms:

- Fuses
- RCONs

The following sections provide details of these mechanisms.

30.3.2.1 Boot from fuses

Boot from fuses is the primary method of configuring the boot process for production devices. In this method, boot code fetches boot configuration data from three fuse words. The following table lists the fuse words associated with boot from fuses.

Fuse word	Offset
BOOT_CFG1	234h
BOOT_CFG2	238h
BOOT_CFG3	23Ch

See the attached fuse map spreadsheet for fuse details. The Boot ROM Fuse Description tab in the spreadsheet provides a detailed view of the BOOT_CFG*n* fuses. The boot fuse words are described under BOOT_CFG_LOCK and must be programmed by the application. Select the boot from fuses mechanism by blowing the FUSE_SEL fuse and by pulling the BMODE1 and BMODE2 pins to the required state during reset.

NOTE

After you have finalized the correct fuse values for each fuse word associated with boot from fuses, block further changes by enabling the fuse lock for BOOT_CFG.

30.3.2.2 Boot from RCON

During application development, you may change a boot configuration multiple times before reaching the final version. After you blow the boot configuration fuses, you cannot change the configuration. If the configuration contains errors, the device may fail to boot. BootROM supports a mechanism for passing configuration parameters. The mechanism is referred to as *Boot from RCON*.

Boot from RCON enables you to change boot configurations during development without blowing fuses, because the RCON boot configuration data resides outside the chip. This boot mechanism is useful primarily during development to enable testing various boot configurations before blowing fuses, which finalizes the configuration.

BootROM supports booting from parallel RCON and serial RCON, as explained in the following sections.

30.3.2.2.1 Boot from parallel RCON

RCON uses up to 32 general-purpose I/O pins—RCON[0] to RCON[31]—that are latched on functional reset deassertion. RCON[31:0] corresponds bit by bit to values in the fuse word BOOT_CFG1[31:0]. On every reset, the chip reads the 32 I/O pads and loads their values into the SRC module's BOTT_GPR_BMR1 register for later use by BootROM.

RCON[31:0] contains configuration parameters that may change multiple times during development. The BOOT_CFG2 and BOOT_CFG3 fuse words contain configuration parameters that do not require testing of multiple configurations.

On development boards, each of the 32 I/Os requires a DIP switch and an analog isolation gate to allow normal operation after reset.

30.3.2.2.2 Boot from serial RCON

An RCON over serial EEPROM scheme, referred to as serial RCON, is available. Unlike parallel RCON, serial RCON does not require DIP switches for 32 I/O PADs and analog gates, saving significant board area.

Serial RCON requires an external serial EEPROM that communicates with the chip via I^2C. The 32 reset configuration bits are fetched from the external EEPROM (first 4 bytes, starting from offset 0h) as part of the boot process. I^2C transmits one byte of data per read and BootROM reads the first 4 bytes. BootROM expects RCON[7:0] on first read, RCON[15: 8] on second, RCON[23:16] in third, and RCON[31:24] on the fourth read. Because I^2C transmits the most significant bit first, keep the values in little-endian format.

The chip operates in master mode and the external serial EEPROM must be in slave mode. The SCL and SDA lines are configured in open-drain mode.

In master mode, I^2C drives the SCL clock from XBAR_DIV3_CLK. I^2C is initialized before the system clock is switched to PLL. The Inter Bus Clock rate (IBFD[IBC]) is programmed as 0Fh, and the effective SCL frequency is 8 MHz/68 = 117.67 kHz. See the I^2C divider and hold tables in the Inter-Integrated Circuit (I^2C) chapter for the corresponding I^2C timing definitions.

Serial EEPROM devices currently supported include:

- Atmel AT24C01C
- ON Semiconductor CAT24C01

Address pins A2 through A0 of the external serial EEPROM should be set to 0 to make the device address A0h.

30.3.2.3 RCON mode selection

The RCON[8] value determines which RCON mode is enabled:

- RCON[8] = 1 indicates that an I^2C EEPROM is connected to RCON[7:8], and 32 bits of configuration data must be read from the EEPROM (see Boot from serial RCON).
- RCON[8] = 0 indicates that all 32 parallel RCON pins are used for the boot configuration (see Boot from parallel RCON).

Serial RCON uses two pins for configuration:

- RCON[7] is the I^2C SDA signal.
- RCON[8] is the I^2C SCL signal. It also indicates to BootROM whether RCON is going to use 32 parallel I/Os or a serial I^2C device.

30.3.2.4 Selection of boot from fuses or boot from RCON

Boot from RCON is a development feature that is enabled on unprogrammed devices, so you do not need to do anything to select it. When you decide to use boot from RCON during application development, you need only choose between parallel RCON and serial RCON. See RCON mode selection for details.

After development is complete, the recommendation is to force the chip to always boot from fuses by performing the following steps:

1. Blow the BOOT_CFG*n* fuses with the proper values.
2. Blow the FUSE_SEL fuse to permanently disable boot from RCON. During end-of-line programming on the production line, you can perform this step via serial boot.
3. Pull the BMODE pins to the required state during reset. See Table 126 for BMODE pin states.

The following diagram shows the logical view of selecting the source of boot configuration information from fuses, parallel RCON, or serial RCON.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4848-b251-90d3a1624c

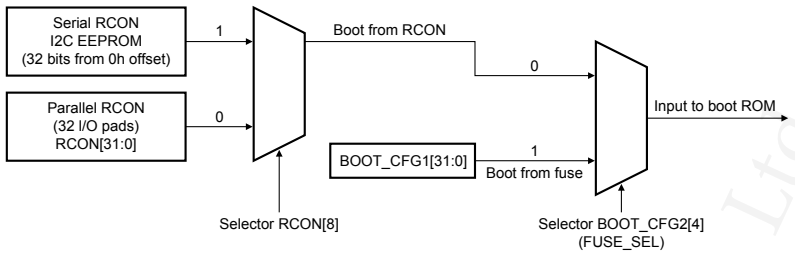


Figure 138. Source selection of boot configuration data

30.4 Program image

This section describes the set of images that you must include in a chip program image. A program image consists of:

- Image Vector Table (IVT) (see Image Vector Table (IVT))
- Device Configuration Data (DCD) (see Device Configuration Data (DCD))
- Self-Test DCD (see Self-Test)
- Application boot image (see Application boot)
- Serial boot image (see Serial boot)

30.4.1 Image Vector Table (IVT)

The IVT is the first image that BootROM reads from the boot device. The IVT contains the required data components: image entry point, pointer to Device Configuration Data (DCD), and other pointers used by BootROM during the boot process. The location of the IVT is the only fixed requirement by BootROM. The remainder of the image memory map is flexible and is traceable by the contents of the IVT. The IVT offset from the device base address and initial load region size for each boot device is defined in the tables below.

Table 127. IVT offset

Boot device type	IVT offset
SD/MMC/eMMC	1000h
QuadSPI serial flash memory	0h

BootROM expects pointers to DCD, Self-Test DCD, the application image, and the HSE_H firmware image to be 8-byte aligned for QuadSPI boot and 512-byte block aligned for µSDHC boot.

Table 128. IVT image structure

Address	Size (bytes)	Name	Comments
0h	4	IVT header	Header showing the start of the IVT
4h	4	Reserved	Reserved
8h	4	Self-Test DCD pointer	Pointer to the start of the configuration data used for BIST
Ch	4	Self-Test DCD pointer (backup)	Pointer to the start of the backup configuration data used for BIST
10h	4	DCD pointer	Pointer to the start of DCD configuration data

Table continues on the next page...

Table 128. IVT image structure (continued)

Address	Size (bytes)	Name	Comments
14h	4	DCD pointer (backup)	Pointer to the start of backup DCD configuration data
18h	4	HSE_H firmware flash memory start pointer	Pointer to the start of the HSE_H firmware in flash memory
1Ch	4	HSE_H firmware flash memory start pointer (backup)	Pointer to the start of the backup HSE_H firmware in flash memory
20h	4	Application boot code flash memory start pointer	Pointer to the start of the application boot code in flash memory
24h	4	Application boot code flash memory start pointer (backup)	Pointer to the start of the backup application boot code in flash memory
28h	4	Boot configuration word	Configuration data used to select the boot configuration
2Ch	4	Life cycle configuration word	Configuration data used for advancing life cycle
30h	4	Reserved	Reserved
34h	32	Reserved for HSE_H firmware	Defined by the HSE_H firmware specification
54h	156	Reserved	Reserved
F0h	16	Galois Message Authentication Code (GMAC)	GMAC of first 240 bytes of IVT image structure

Table 129. IVT image header

Byte 0	Byte 1	Byte 2	Byte 3
TAG = D1h	Length = 100h		Version = 60h

30.4.1.1 IVT boot configuration word

Table 130. IVT boot configuration word

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												BOOT_SEQ	SWT	BOOT_TARGET	

Table 131. IVT boot configuration word field descriptions

Field	Description
31:5 —	Reserved

Table continues on the next page...

Table 131. IVT boot configuration word field descriptions (continued)

Field	Description
3 BOOT_SEQ	Secure Boot Mode 0b - Non-secure boot. BootROM executes application image without authentication. 1b - Secure boot. HSE_H firmware executes application image after authentication.
2 SWT	Boot Target Watchdog 0b - BootROM does not configure SWT_0 1b - BootROM enables SWT_0 with default timeout before BOOT_TARGET enabled
1:0 BOOT_TARGET	Boot Target Identifies the boot target, which is the core responsible for running application code. 00b - Cortex-M7_0 01b - Cortex-A53_0 10b - Reserved 11b - Reserved

30.4.1.2 Life cycle configuration word

Table 132. Life cycle configuration word

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved														IN_FIELD	OEM_PROD

Table 133. Life cycle configuration word field descriptions

Field	Description
31:2 —	Reserved
1 IN_FIELD	In Field Advances life cycle state to IN_FIELD. This field takes precedence over the OEM_PROD field value. 0b - Life cycle state does not advance to IN_FIELD 1b - Life cycle state advances to IN_FIELD
0 OEM_PROD	OEM Production Advances life cycle state to OEM_PROD. 0b - Life cycle state does not advance to OEM_PROD 1b - Life cycle state advances to OEM_PROD

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4603c-4821-48a8-9251-390d3ab02461

30.4.2 Device Configuration Data (DCD)

On reset, all module registers have default values. These settings are typically not ideal for achieving optimal system performance. Also, some peripherals must be configured before they can be used. DCD is the configuration information contained in the DCD image that BootROM uses to configure peripherals on the device. BootROM determines the location of the DCD table from the pointer in the IVT (see Table 128). Table 134 shows the DCD image structure—a big-endian byte array of allowed DCD commands. The maximum size of the DCD image is 8192 bytes.

Table 134. DCD image structure

Address offset	Size (bytes)	Name	Comments
0h	4	DCD header	Header to signify start of DCD data
4h	DCD_Length	DCD data	DCD commands
DCD_Length + 4	16	GMAC	Cryptographic hash for DCD header and complete DCD data using IVTDCD key

Table 135. DCD header format

Byte 0	Byte 1	Byte 2	Byte 3
TAG = D2h	Length[1] (2 bytes)		Version = 60h

1. Size of the DCD data plus the size of the DCD header

30.4.2.1 DCD commands

The DCD data is comprised of three commands:

- Write Data
- Check Data
- NOP

30.4.2.1.1 Write Data command

Use the Write Data command to write a list of 1-byte, 2-byte, or 4-byte values (or bit masks) to a corresponding list of target addresses. The Write Data command structure is a big-endian byte array as shown in the table below.

Table 136. Write Data command format

Command type	Byte 0	Byte 1	Byte 2	Byte 3
Header	TAG = CCh	Length		Parameters
CMD	Address			
	Value/Mask			
CMD	Address			
	Value/Mask			
...	...			
CMD	Address			
	Value/Mask			

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-488-b251-390d3ab624c1

Table 137. Write Data command byte descriptions

Command byte(s)	Description
Length	Size of entire Write Data command structure including header.
Parameters	A single byte divided into fields as shown in Table 138.
Address	Target address where data will be written. As shown in Table 136, you can specify one or more Address and Value/Mask pairs. The same bytes and flag parameters apply to all locations in the command.
Value/Mask	Data value or bit mask to be written to Address. As shown in Table 136, you can specify one or more Address and Value/Mask pairs. The same bytes and flag parameters apply to all locations in the command.

Table 138. Write Data command Parameters byte

7	6	5	4	3	2	1	0
			Data Set	Data Mask	Bytes		

Table 139. Write Data command Parameters byte field descriptions

Field	Description
7:5 —	Reserved
4 Data Set	Status bit showing whether bits at target were successfully overwritten. 0 - Bits at target address not overwritten 1 - Bits at the target address overwritten
3 Data Mask	Enables overwriting of bits at target address. 0 - All bits may be overwritten 1 - Only specific bits may be overwritten
2:0 Bytes	Width of target location(s) in bytes. 001 - 1 byte 010 - 2 bytes 100 - 4 bytes

When successful, this command writes to each target address in accordance with the Data Mask and Data Set values as indicated in the following table.

Table 140. Effect of Write Data command Data Mask and Data Set values

Data Mask	Data Set	Action	Interpretation
0	X	*address = val_mask	Write value
1	0	*address &= ~val_mask	Clear bit mask
1	1	*address |= val_mask	Set bit mask

BootROM performs the following checks when processing the list of write Address/Value pairs:

1. If any of the target addresses does not have the same alignment as the data width specified in the Bytes field of the Parameters byte, none of the values are written.
2. If any of the values is larger, or any of the bit masks is wider, than permitted by the data width specified in the Bytes field of the Parameters byte, none of the values are written.
3. If any of the target addresses do not lie within an allowed region, none of the values are written. For allowed target address ranges, see the notes below.

NOTE

- You cannot use the Write Data command to write SRAM before SRAM initialization, because this will cause an ECC error due to the 64-bit ECC protection.
- BootROM does not allow HSE_H address space or HSE_H Cortex-M7 system control register space to be modified via DCD commands.

30.4.2.1.2 Check Data command

Use the Check Data command to test for a specific 1-, 2-, or 4-byte bit mask from a source address. The Check Data command structure is a big-endian byte array as shown in the following table.

Table 141. Check Data command structure

Command type	Byte 0	Byte 1	Byte 2	Byte 3
Header	TAG = CFh (1 byte)	Length (2 bytes)		Parameters (1 byte)
CMD	Address (4 bytes)			
	Mask (4 bytes)			
	Count (4 bytes)			

Table 142. Check Data command byte descriptions

Command byte(s)	Description
Length	Size of the entire Check Data command structure, including the header. Used to determine the optional Count.
Parameters	A single byte divided into fields as shown in Table 143.
Address	Source address to test. The values in the same Parameters byte apply to all the Address/ Mask pairs in the command structure.
Mask	Bit mask to test. The values in the same Parameters byte apply to all the Address/Mask pairs in the command structure.
Count	Optional poll count. If no value is specified this command polls indefinitely until the exit condition is met. For a finite value, this command polls Count times. If Count = 0, this command behaves as a NOP command.

Table 143. Check Data command Parameters byte

7	6	5	4	3	2	1	0
			Data Set	Data Mask	Bytes		

Table 144. Check Data command Parameters byte field descriptions

Field	Description
7:5 —	Reserved
4 Data Set	Enables bit checking at target address. 0 - Bits at target address are ignored 1 - Bits at target address are checked
3 Data Mask	Enables partial/complete checking of bits when the Data Set field is set to 1. 0 - All bits in mask are checked 1 - Any bit in mask may be checked at target address
2:0 Bytes	Size of target locations in bytes. 001 - 1 byte 010 - 2 bytes 100 - 4 bytes

The Check Data command polls until either the exit condition is satisfied, or the poll count is reached. If the exit condition is not satisfied until the poll count is reached, it is simply skipped. The exit condition is determined by the Data Mask and Data Set values as shown in the following table.

Table 145. Effect of Check Data command Data Mask and Data Set values

Data Mask	Data Set	Action	Interpretation
0	0	*address & mask == 0	All bits in mask clear.
0	1	*address & mask == mask	All bits in mask set.
1	0	*address & mask != mask	At least one bit in mask clear.
1	1	*address & mask != 0	At least one bit in mask set.

Reasons Check Data command fields are not read:

- If the Address does not have the same alignment as the data width specified in the Bytes field of the Parameters byte, the value is not read.
- If the Mask is wider than permitted by the data width specified in the Bytes field of the Parameters byte, the value is not read.

30.4.2.1.3 NOP command

This command has no effect. The NOP command structure is a big-endian 4-byte array as shown in the table below.

Table 146. NOP command structure

Byte 0	Byte 1	Byte 2	Byte 3
TAG = C0h	Length = 04h		Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c625-42a-4a8-8251-590d3ab624c1

Table 147. NOP command byte descriptions

Command byte(s)	Description
Length	Size of the NOP command in bytes, fixed to 4. Any value in this field is ignored.
Reserved	This byte is ignored.

30.4.3 Self-Test

BootROM provides the capability to optionally execute self-test on reset.

Self-test is executed when all of the following conditions are true:

- Device is coming out of POR and self-test completion status is not set.
- Self-Test Image Vector is programmed in IVT.

30.4.3.1 Self-Test boot operation

BootROM executes self-test only when booting from POR and self-test completion status (ST_DONE) is not set in MC_RGM. The following sequence shows how the BootROM executes the self-test.

1. Verify the Self-Test DCD Image header.
2. Copy the Self-Test DCD Image to internal memory.
3. Authenticate image if (LC==(OEM_PROD|IN_FIELD) && IVT_AUTH==1) && EXPORT CONTROL NOT ENABLED.
4. Execute self-test commands.

NOTE

The last self-test command must be to start self-test by setting RUNSW[RUNSW] = 1 in STCU2.

30.4.3.2 Self-Test DCD image structure

The Self-Test DCD image structure has the same format as the DCD image structure (see Device Configuration Data (DCD)). BootROM determines the location of the Self-Test DCD image via the Self-Test DCD pointer in the IVT. The maximum size of the image is 8192 bytes (see Device Configuration Data (DCD) for details on available commands).

Table 148. Self-Test header

Byte 0	Byte 1	Byte 2	Byte 3
TAG = D3h	Length[1]		Version = 60h

1. Size of the entire Self-Test DCD Image structure, that is, the size of the Self-Test DCD data and the DCD header.

30.4.4 Application boot

The Application Boot Code image must comply with the structure shown in the following table for non-secure boot (BOOT_SEQ == 0).

Table 149. Non-secure Application Boot Code image structure (BOOT_SEQ == 0)

Address offset	Size (bytes)	Name	Comments
0h	4	Image header	Marks start of application image.
4h	4	RAM start pointer	Pointer to the first RAM address to which BootROM

Table continues on the next page...

Table 149. Non-secure Application Boot Code image structure (BOOT_SEQ == 0) (continued)

Address offset	Size (bytes)	Name	Comments
			must load application boot code.
8h	4	RAM entry pointer	Pointer to out of RESET start of boot target core. For Cortex-M7, it corresponds to VTOR. For A53, it corresponds to start of code execution. This pointer should be within the section of SRAM where the application image is downloaded.
Ch	4	Code length	Length of code section of the image.
10h	48	Reserved	Reserved
40h	Code_len	Code	Code can be any size up to the maximum size of system SRAM.

The header used by the Application Boot Code image is shown below.

Table 150. Application boot image header

Byte 0	Byte 1	Byte 2	Byte 3
TAG = D5h	Reserved[1]		Version = 60h

1. The Reserved field has no effect. It is set to all zeroes.

NOTE

For application boot via the µSDHC interface, when BOOT_SEQ == 0, the RAM start pointer for the application should not point between 34008000h to 34078000h. This address range is used by BootROM for internal operation during boot via the µSDHC interface. BootROM also uses 4 KB of SRAM memory starting at 343FF000h for ADMA descriptors in case of µSDHC boot. The Application boot image header should not point to this location in case of µSDHC boot.

30.4.5 Serial boot

The image structure for non-secure serial boot is shown below.

Table 151. Image structure for non-secure serial boot

Address offset	Bytes	Name	Comments
0h	4	Image header	Marks start of serial boot code image (see Table 153).
4h	4	RAM start pointer	Pointer to first RAM address to which BootROM must load serial boot code.

Table continues on the next page...

Table 151. Image structure for non-secure serial boot (continued)

Address offset	Bytes	Name	Comments
8h	4	RAM Entry pointer	Pointer to out of RESET start of boot target core. For CCortex-M7, it corresponds to VTOR, and for Cortex-A53 to start of code execution.
Ch	4	Code length word	Length of code section of the image (see Table 154).
10h	48	Reserved	Reserved
40h	Code_len	Code	Code can be any size up to the maximum size of SRAM.

The image structure for secure serial boot is shown below.

Table 152. Image structure secure serial boot

Address offset	Bytes	Name	Comments
0h	4	Image header	Marks start of serial boot code image (see Table 153).
4h	4	RAM start pointer	Pointer to first RAM address to which BootROM must load serial boot code.
8h	4	RAM Entry pointer	Pointer to out of RESET start of boot target core. For Cortex-M7, it corresponds to VTOR, and for Cortex-A53 to start of code execution.
Ch	4	Code length word	Length of code section of the image (see Table 154).
10h	4	Auth mode	Authentication mode. Reserved for non-secure boot.
14h	4	NSKPUB Key Selector	NSK public key selector
18h	40	Reserved	Reserved
40h	4096	NSKPUB	NXP Public keys (16) for RSA authentication.
1040h	256	CSKPUB	Customer Public key for RSA authentication.
1140h	256	CSK Signature	RSA signature for CSKPUB authentication.
1240h	Code length	Code	Code can be any size up to the maximum size of SRAM.
1240h + Code length	256	Public key Signature	Public key signature for image authentication.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.
c8c462e5-4821-48a8-b231-390d3ab6a4

NOTE

CSKPUB and CSK Signature fields are reserved if the serial boot application is signed by NXP.

The image header shown below is common to both the non-secure and secure serial boot image structures.

Table 153. Non-secure and secure boot image header

Byte 3	Byte 2	Byte 1	Byte 0
Version = 60h	Reserved	Reserved	TAG = D8h

The Code length word is described in the table below.

Table 154. Code length word

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NO_ECHO	CODE_SIZE														
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CODE_SIZE															

Table 155. Code length word field description

Field	Description
31 NO_ECHO	Do not echo incoming serial data from next serial phase onward.
30:0 CODE_SIZE	Size of the serial boot code to be downloaded.

30.5 Boot register descriptions

During execution, Boot ROM accesses memory-mapped registers to:

- Retrieve configuration details
- Maintain execution details required for proper execution

Some of the registers are read-only and provide input to Boot ROM. Other registers are read/write and contain status information required for proper functioning. The following sections contain the register memory map and detailed register descriptions that include reset phase and field descriptions.

30.5.1 Boot memory map

Boot_CC base address: 4007_C900h

Offset	Register	Width (In bits)	Access	Reset value
0h	Boot Configuration 1 (BOOT_GPR_BMR1)	32	RO	0000_0000h
4h	Boot Configuration 2 (BOOT_GPR_BMR2)	32	RO	0000_0000h
38h	SRC POR Control (SRC_POR_CTRL_REG)	32	RW	0000_0000h

30.5.2 Boot Configuration 1 (BOOT_GPR_BMR1)

Offset

Register	Offset
BOOT_GPR_BMR1	0h

Function

Contains boot configuration data latched from fuses or GPIOs based on RCON/FUSE_SEL fuse setting. The data is latched at PHASE3 deassertion in the chip. The reset value for a chip that is not configured is 0000_0000h. Otherwise, the reset value is dependent on the values latched.

RCON uses 32 IO signals labelled RCON[0] to RCON[31]. See the attached IOMUX spreadsheet for details.

NOTE

RCON pads should be supplied with a pull on the board.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BOOT_CFG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BOOT_CFG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BOOT_CFG	Boot Configuration Boot configuration data from BOOT_CFG fuses or RCON pads, depending on the FUSE_SEL fuse state.

30.5.3 Boot Configuration 2 (BOOT_GPR_BMR2)

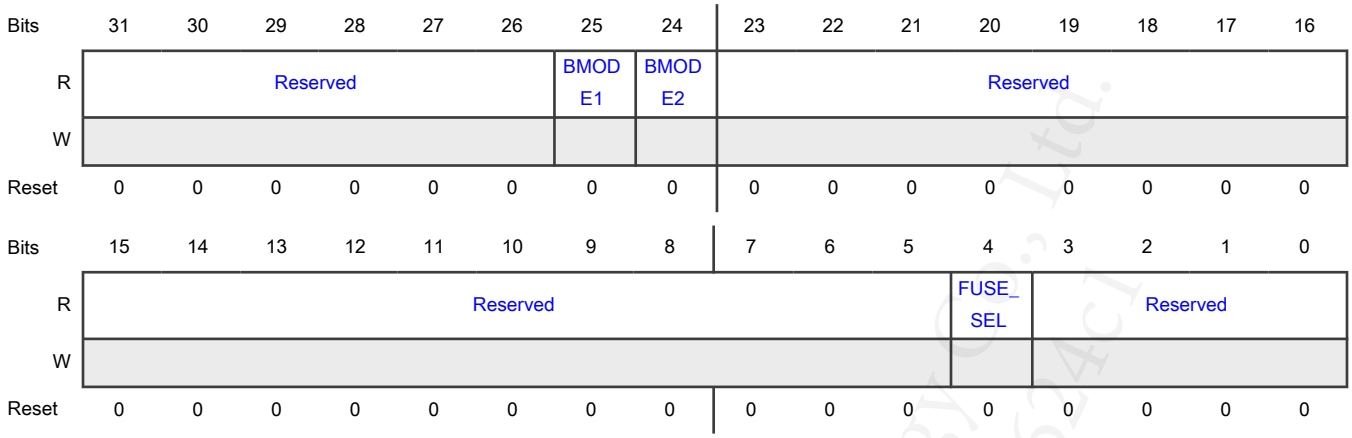
Offset

Register	Offset
BOOT_GPR_BMR2	4h

Function

Boot Mode pins configuration and FUSE_SEL setting latched from fuses or GPIOs on PHASE3 deassertion in the chip.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 BMODE1	Boot Mode Pin 1 BMODE1 pin value latched at phase3 deassertion.
24 BMODE2	Boot Mode Pin 2 BMODE2 pin value latched at phase3 deassertion.
23-5 —	Reserved
4 FUSE_SEL	Select Boot from Fuses FUSE_SEL value from fuses. This fuse indicates whether the boot configuration is to be fetched from RCON pads or BOOT_CFG fuses.
3-0 —	Reserved

30.5.4 SRC POR Control (SRC_POR_CTRL_REG)

Offset

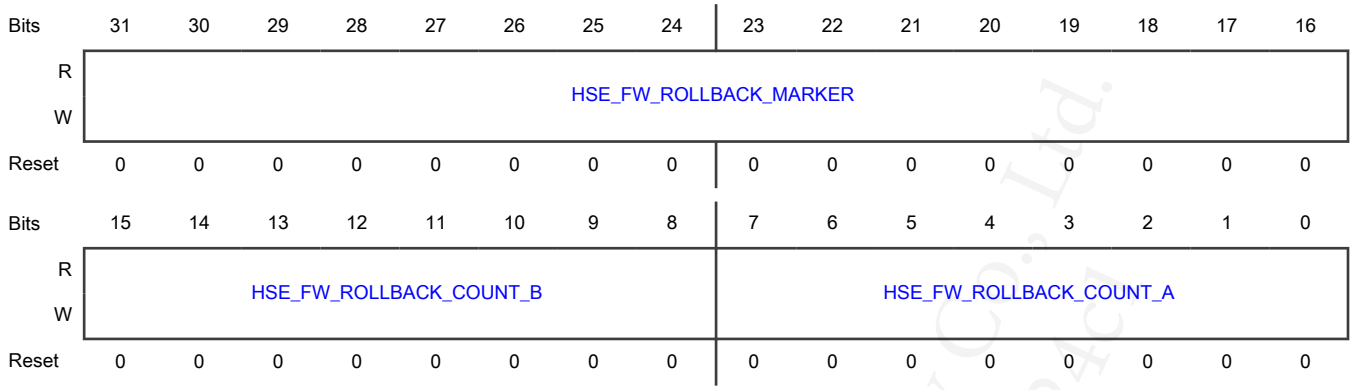
Register	Offset
SRC_POR_CTRL_REG	38h

Function

Boot ROM uses this register to implement the HSE_H roll back mechanism.

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESHJ Automotive Technology Co., Ltd.
8c462e54-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-16 HSE_FW_ROLLBACK_MARKER	HSE_H Firmware Rollback Marker HSE_H firmware writes the HSE_H rollback marker. boot ROM defines only BADAh as a valid marker.
15-8 HSE_FW_ROLLBACK_COUNT_B	HSE Firmware Rollback Count – Backup Image Represents the number of times that boot ROM passed control to the HSE_H firmware backup image, and the HSE_H firmware failed to write a valid HSE_FW_ROLLBACK_MARKER value.
7-0 HSE_FW_ROLLBACK_COUNT_A	HSE Firmware Rollback Count – Primary Image Represents the number of times that boot ROM passed control to the HSE_H firmware primary image, and HSE_H firmware failed to write a valid HSE_FW_ROLLBACK_MARKER value.

30.6 Boot flow

HSE_H BootROM boots HSE_H firmware and/or application code based on the BOOT_SEQ value. Following are basic details about boot flow.

1. Depending on self-test pointers and other BootROM checks on the Self-Test DCD, boot flow can complete in either three phases or one phase. See Boot flow phases for more detail.
2. BootROM supports booting of application code and and HSE_H firmware images, depending upon the BOOT_SEQ value and whether the image pointers in the IVT are valid.
3. BootROM supports the concept of backup images for:
   - Self-Test DCD
   - DCD
   - Application code
   - HSE_H firmware

Beijing CTI Automotive Technology Co., Ltd.
c8c4623-42-48a-b23-390d3ab624c4

For each type of program image, BootROM first tries to execute the primary image if the IVT points to a valid location. Only 0h is considered an invalid location. If any of the following are encountered, BootROM tries to execute the backup image pointed to by the IVT, wherever applicable:

- Primary image points to 0h.
- Primary image header does not conform to the required image header format.
- Authentication fails.

4. BootROM allows rollback of the HSE_H firmware from the backup image when a failed update of HSE_H firmware occurs in its primary slot. The failure scenario is as follows:
   a. The updated primary HSE_H firmware image passes all header checks.
   b. The updated image successfully completes the BootROM authentication process.
   c. After getting control, the updated firmware issues a RESET or generates a SWT RESET due to a hang scenario before reaching the point of execution where it can boot application code and provide services.

   BootROM can identify these scenarios.

   a. After a sequence of 8 failed attempts for the primary HSE_H firmware image, BootROM tries booting from the HSE_H firmware backup image.
   b. When a sequence of 8 failures of the backup image occurs, BootROM enters Serial Boot.

   BootROM uses an SRC GPR register (POR_CTRL_REG1) to keep track of the number of trials for the primary image and the backup image. A marker, HSE_FW_ROLLBACK_MARKER (BADAh), must be written by HSE_H firmware at the point of its execution where it has passed control to application code and is ready to provide services. BootROM looks for this marker to determine whether booting from the HSE_H firmware primary image or the backup was successful.

See Boot register descriptions for more details about the POR_CTRL_REG1 register.

30.6.1 Boot flow phases

When booting from external flash memory, BootROM execution occurs in three phases.

Table 156. Boot flow phases when booting from external flash memory

Step	Phase	Description
1	Initialize 1	BootROM execution on HSE_H core.
2	Self-Test	LBIST/MBIST execution by STCU2 (if configured).
3	Initialize 2	BootROM execution on HSE_H core after STCU2-induced reset.

The Initialize 2 phase initializes the device after reset when self-test completes, and continues the boot process. If the Self-Test DCD vector is not programmed in the IVT, the boot flow skips step 1 and step 2, and starts from step 3, the Initialize 2 phase.

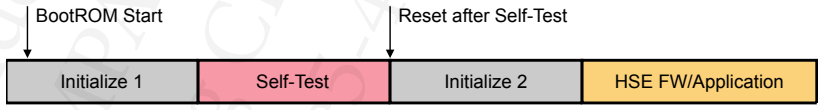


Figure 139. Boot flow phases

30.6.1.1 Initialize 1 boot flow phase

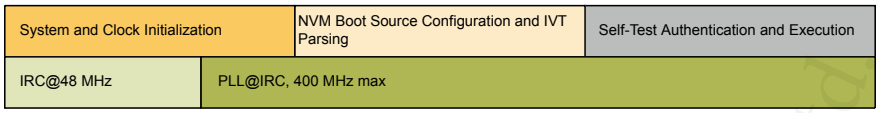


Figure 140. Initialize 1 boot flow phase

The steps below define the Initialize 1 boot flow phase:

1. Initialize the stack pointer to memory pointed to by the stack area as described in the memory map.
2. If coming out of Standby mode, enter boot from Standby mode (see Standby boot).
3. If OCOTP detects a double-bit ECC (DED) error for HSE_H or the SYS fuse box, or the MTR status is failure, issue a destructive reset.
4. Identify the boot interface.
5. Enable PLL@FIRC and reconfigure the clock to run at 400 MHz.
6. Initialize CAAM and PKC.
7. If the number of functional resets ≥ 8, enter Serial Boot mode.
8. Configure QuadSPI/SD/MMC/eMMC based on fuse information.
    a. If interface is QuadSPI:
        - Read customer-provided flash memory reconfiguration parameters.
        - Reconfigure QuadSPI to run at the required speed, pads, and other configuration.
9. Copy the IVT from QuadSPI/SD/MMC/eMMC.
10. Authenticate the IVT, if (LC==OEM_PROD OR IN_FIELD) AND (IVT_AUTH==1) AND (EXPORT CONTROL NOT ENABLED).
11. Check the IVT and issue a functional reset if any of these conditions apply:
    - Header is invalid.
    - Boot target is incorrect.
    - Both the primary and backup pointers for application code and HSE_H firmware are NULL.
12. If self-test has not been executed, copy and authenticate (depending on step 10) the Self-Test DCD data pointed to by the primary pointer if the pointer is not NULL. If any failure occurs—authentication, invalid header, or wrong DCD, for example—do not execute the remaining Self-Test DCDs under the primary pointer and instead begin self-test configuration operations from the Self-Test DCD under the backup pointer.

    If authentication fails for the Self-Test DCD data pointed to by the backup pointer, issue a functional reset.

    In case of other failures—invalid header or wrong DCD, for example—skip DCD execution for subsequent DCDs from backup and allow BootROM to proceed to subsequent application boot steps.

NOTE

It is recommended that a Write Data DCD command be added just before the DCD command to trigger STCU2. The Write Data DCD command can write a Self-Test DCD configuration completion signature marker. Application code can later read this location and verify whether the Self-Test DCD configuration completion signature marker is available. Application code can use the ST_DONE status along with the signature marker to determine the outcome of the Self-Test phase.

30.6.1.2 Self-Test

Self-test can execute Memory BIST (MBIST) and/or Logic BIST (LBIST). The tests to be performed are defined in the Self-Test DCD image.

- Self-test is run only after POR, and if the Self-Test Completion status flag is not set to 1 in MC_RGM.
- Self-test is optional and cannot be executed if the primary and back up pointers are NULL.
- Self-test execution begins after the Self-Test DCD triggers STCU2.
- STCU2 performs a reset at the end of self-test.

30.6.1.3 Initialize 2 boot flow phase

The following figure illustrates the Initialize 2 boot flow phase.

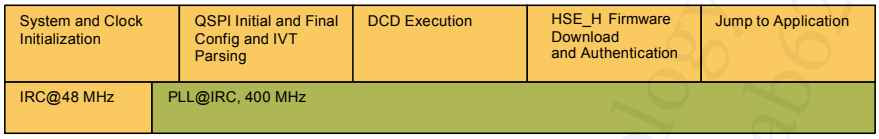


Figure 141. Initialize 2 boot flow phase

The steps below define the Initialize 2 boot flow phase:

1. Initialize the stack pointer to the memory pointed to by the stack area as described in the memory map.
2. If coming out of Standby mode, enter boot from Standby mode. (See Standby boot).
3. If an OCOTP double-bit ECC (DED) error is latched for HSE_H or the SYS fuse box, or the MTR status is failure, issue a destructive reset.
4. Identify the boot interface.
5. Enable PLL@FIRC and reconfigure the clock to run at 400 MHz.
6. Initialize CAAM and PKC.
7. If the number of functional resets ≥ 8, enter Serial Boot mode.
8. Configure QuadSPI/SD/MMC/eMMC based on fuse information.
    a. If interface is QuadSPI:
        i. Read customer-provided flash memory reconfiguration parameters.
        ii. Reconfigure QuadSPI to run at required speed, pads, and other configuration.
9. Copy the IVT from QuadSPI/SD/MMC/eMMC.
10. Authenticate the IVT, if (LC==OEM_PROD OR IN_FIELD) AND (IVT_AUTH==1) AND (EXPORT CONTROL NOT ENABLED).
11. Check the IVT and issue a functional reset if any of these conditions apply:
    - Header is invalid.
    - Boot target is incorrect.
    - Both the primary and backup pointers for application code and HSE_H firmware are NULL.
12. Copy and authenticate (depending on the above step) the DCD data if the DCD pointer is not NULL. In case of any failure—authentication, invalid header, or wrong DCD, for example—do not execute the remaining DCDs under the primary pointer and instead begin DCD operations for the DCD data under the backup pointer.

    If authentication fails for the DCD data pointed to by the DCD backup pointer, issue a reset.

In case of other failures—invalid header or wrong DCD, for example—skip DCD execution for subsequent DCDs from backup and allow BootROM to proceed to subsequent application boot steps.

13. Set up the Debug challenge/response.
14. If the IVT Boot Configuration Word [BOOT_SEQ] == 0, parse the IVT and locate the Primary Application Boot Code Structure.
    a. If the Primary Application header is valid, download the application to target memory.
    b. If the Primary Application header is invalid, verify that the backup application header is valid. If it is, download the backup application to target memory.
    c. If both the Primary Application header and the backup application header are invalid, skip downloading the application.
15. Parse the IVT and locate the Primary HSE_H Firmware Structure.
16. If the value of HSE_ROLLBACK_COUNT_A in the SRC module's POR_CTRL_REG1[7:0] ≤ 8:
    a. If the value of HSE_ROLLBACK_MARKER in the SRC module's POR_CTRL_REG1[31:16] == BADAh:
        i. Clear HSE_ROLLBACK_COUNT_A in the SRC module's POR_CTRL_REG1[7:0] to 0.
        ii. Clear HSE_ROLLBACK_MARKER in the SRC module's POR_CTRL_REG1[7:0] to 0.
    b. If the Primary HSE_H Firmware header is valid:
        i. Decrypt the HSE_H firmware, authenticate, and copy to target memory.
        ii. If authentication is successful:
            i. If (BOOT_SEQ == 0) AND (Application downloaded successfully in step 14), enable the Application core.
            ii. Enable the HSE_H firmware.
            iii. Increment the value of HSE_ROLLBACK_COUNT_A in SRC module's POR_CTRL_REG1[7:0] by 1.
        iii. Else, jump to step 17.
    c. Else, jump to step 17.
17. If the value of HSE_ROLLBACK_COUNT_B in the SRC module's POR_CTRL_REG1[15:8] ≤ 8:
    a. If the value of HSE_ROLLBACK_MARKER in the SRC module's POR_CTRL_REG1[31:16] == BADAh:
        i. Clear the HSE_ROLLBACK_COUNT_B value in the SRC module's POR_CTRL_REG1[15:8] to 0.
        ii. Clear the HSE_ROLLBACK_MARKER value in the SRC module's POR_CTRL_REG1[31:16] to 0.
    b. If the HSE_H firmware backup image header is valid:
        i. Decrypt the HSE_H firmware, authenticate, and copy to target memory.
            i. If authentication is successful:
                i. If (BOOT_SEQ == 0) AND (Application downloaded successfully in step 14), enable the Application target core.
                ii. Enable HSE_H firmware.
                iii. Increment the value of HSE_ROLLBACK_COUNT_B in the SRC module's POR_CTRL_REG1[15:8] by 1.
            ii. Else:
                i. If (BOOTSEQ == 0) and (Application downloaded successfully in step 14):
                    i. Enable the Application target core.

ii. Put the HSE_H Cortex-M7 core into an infinite wait loop.

ii. Else, issue a functional reset.

c. Else:

i. If (BOOTSEQ == 0) AND (Application downloaded successfully in step 14):

i. Enable the Application target core.

ii. Put the HSE_H Cortex-M7 core into an infinite wait loop.

ii. Else, issue a functional reset.

NOTE

When BOOT_SEQ == 0, BootROM downloads application code to SRAM as described in Phase 2 of the boot flow. Although BootROM expects the length of all program images to be 8-byte aligned, for the application download, it initializes an additional 8 bytes of memory just after the last location of the copied image. This is to avoid pre-fetch errors when the application executes.

30.6.2 Boot flow diagram

The following figure shows the high level boot flow of BootROM firmware.

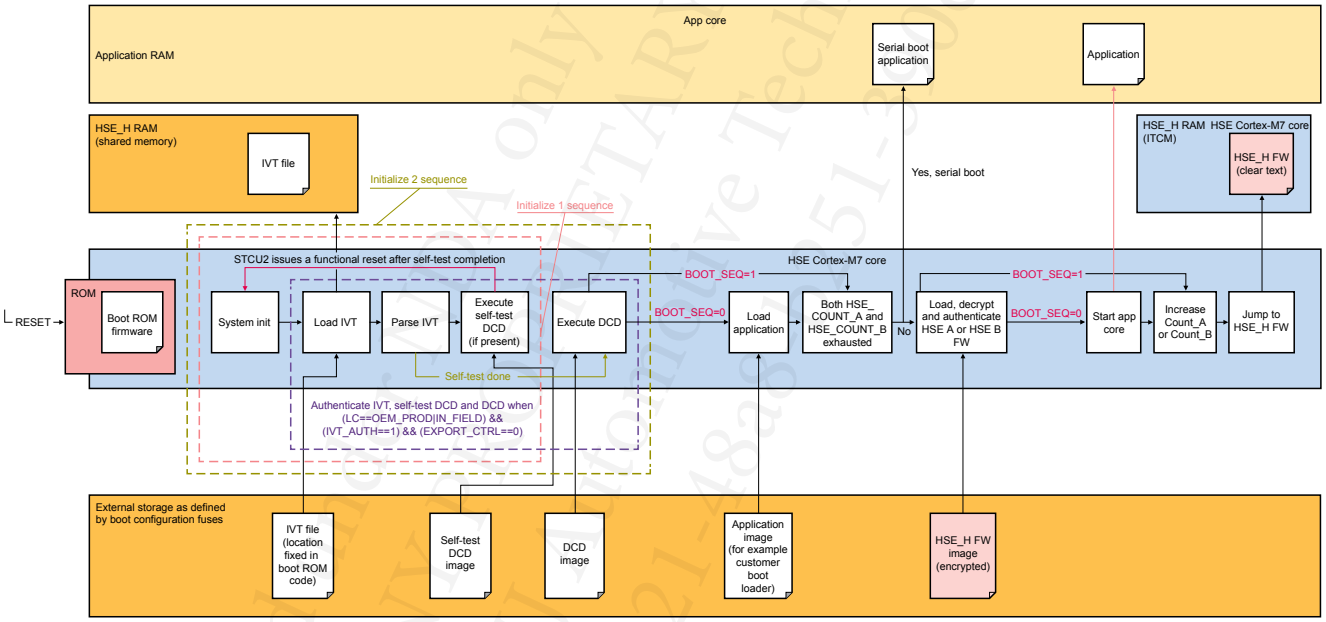


Figure 142. High level boot flow

30.7 Boot settings and fuse map

For additional details about boot settings and the fuse map, see the Boot Settings and Fuse Map spreadsheets.

30.8 BootROM impact on module registers' reset values

BootROM execution might change the reset values of some modules' registers when control passes to the application. For list of these modules, their affected registers, and the registers' values, see the Boot Settings attachment.

NOTE

- BootROM supports many configuration options and permissible combinations. Interfaces used by BootROM, e.g., QSPI or other boot sources, may not be in outright operational state on entry into the booted application. Such interfaces needs to be reconfigured properly before application use.
- If application needs to modify the clock configuration or change interface settings at the start of execution, it should first disable the interface. If application intended to keep using any interface after clock change then it should ensure that interface is reconfigured properly before further use.

30.9 BootROM clock configuration

BootROM begins execution with FIRC as the default system clock source. For better performance, BootROM tries to configure CORE PLL-DFS over FIRC. After CORE PLL-DFS is sucessfully locked, BootROM switches the XBAR clock (source to booting core) to CORE_DFS1_CLOCK, which is targeted to be 400 MHz. In case of failure to lock PLL-DFS, BootROM continues booting with FIRC as the system clock.

BootROM also has a provision to boot without locking the core PLL. This feature is controlled by BOOT_CFG1[31]/RCON[31]. A value of 1b forces BootROM to skip PLL-DFS programming when booting from QuadSPI or µSDHC. BOOT_CFG1[31]/RCON[31] does not have any impact on PLL-DFS programming when booting from any serial interface—for example, BootROM always configures the core PLL when performing a serial download.

When BOOT_SEQ == 0, before BootROM passes control to application code, the system clock is switched back to FIRC and CORE PLL-DFS is switched off.

When BOOT_SEQ == 1, BootROM passes control to the HSE_H firmware with the system clock as CORE PLL-DFS.

See the respective interface sections for the clocking scheme of boot interfaces.

30.10 Image authentication and decryption

BootROM can load multiple images from non-volatile storage space (QuadSPI or SD/MMC/eMMC) to boot on:

- HSE_H Cortex-M7 core (for HSE_H firmware)
- Cortex-M7/Cortex-A53 core (for application boot code)

Application boot can be either non-secure or secure, depending on the life cycle state and the BOOT_SEQ field value in the IVT Boot Configuration word. The following images are authenticated by BootROM when performing a secure boot:

- IVT
- DCD
- Self-Test DCD
- Serial application image
- HSE_H firmware image

30.11 Serial boot

Serial Boot mode is entered via the BMODE input pins. Serial download can also be initiated if the Functional Reset Counter (FREC register in the MC_RGM module) reaches a value ≥8. This mode is available in all life cycle states except:

- OUT_OF_FAB
- FA

In Serial Boot mode, BootROM programs the HSE_H SWT for a 60-second timeout, then continuously polls for activity on any of the available interfaces:

- Ethernet
- CAN

- UART

If no activity is detected, the timer expires and the core is reset. BootROM sequentially checks for activity on all available interfaces and selects the first serial interface that it identifies as active as the download interface. The following figure describes the selection process.

NOTE

In Serial Download mode, BootROM disables HSE_H SWT if the life cycle state is:

- MCU_PROD
- CUST_DEL

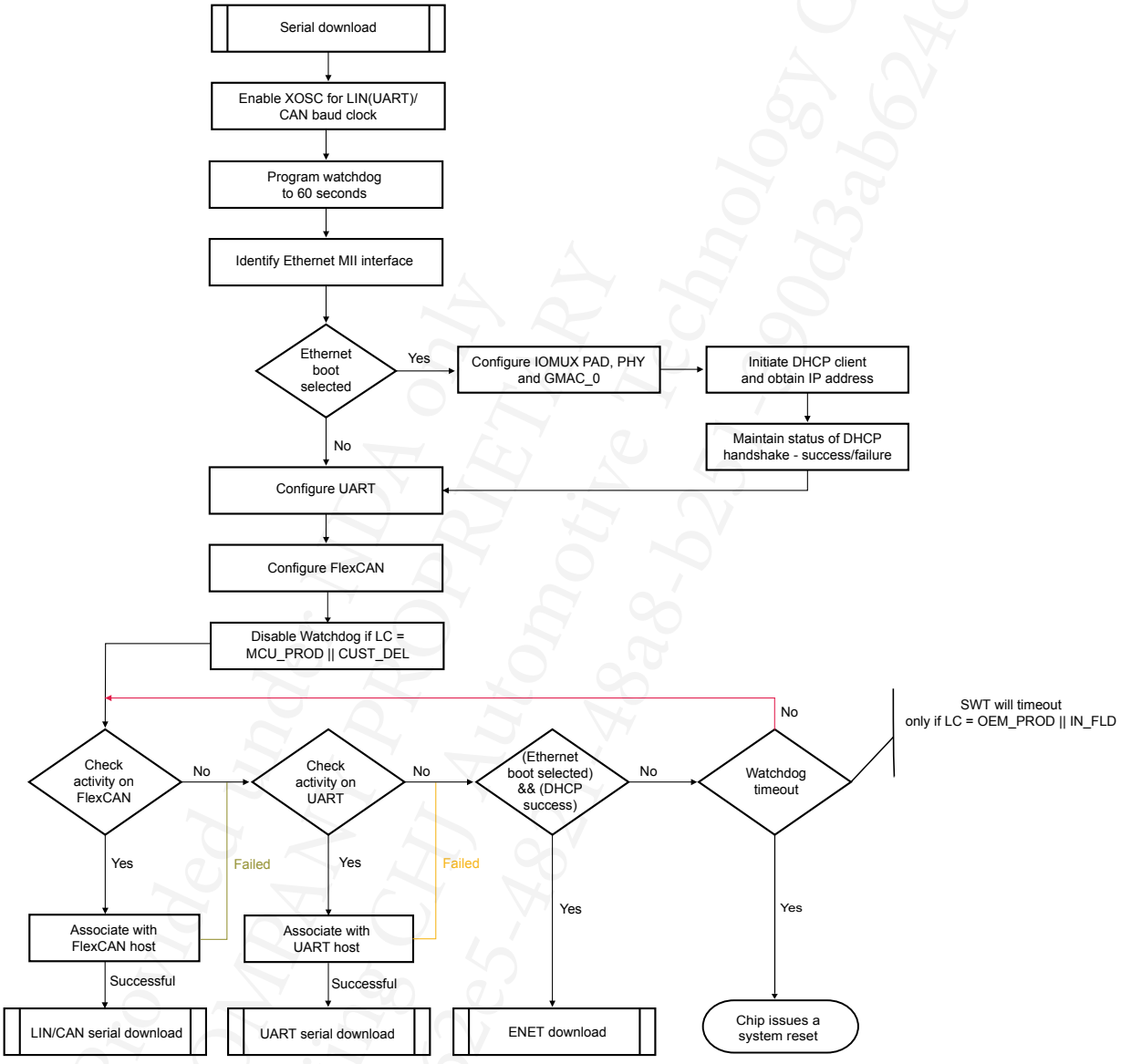


Figure 143. BootROM serial download sequence

30.11.1 XOSC configuration

During serial boot, BootROM uses XOSC as the clock source for serial interfaces.

- LIN_CLK and CAN_CLK are derived from an external clock source connected over XOSC.

- The GMAC_0 clock is derived from the Peripheral PLL (PERIPH_PLL) locked over XOSC.

The XOSC interface supports multiple operating modes and requires specific configuration for providing a stable clock in these modes. BootROM must be provided with the configuration details for enabling correct configuration, as per clock source connected on ECU, over the XOSC interface.

30.11.2 Serial download protocol

30.11.2.1 Serial download protocol – UART and CAN

The following figure shows the download protocol followed by BootROM during serial download over FlexCAN and UART.

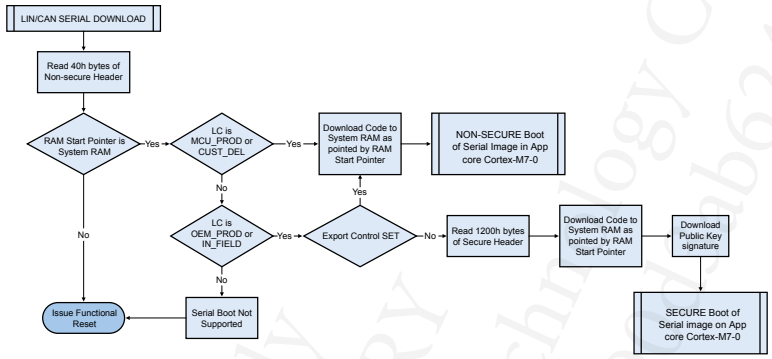


Figure 144. FlexCAN and UART download protocol

The serial download protocol is divided into various phases as described below. The phases marked "Secure boot", are skipped during a non-secure boot.

Table 157. Serial download protocol phases

No.	Phase	Description	Size (bytes)	Entry
1	Initialization	BootROM prepares the device to perform a serial download by performing the following steps: 1. Configure clock. 2. Configure the module. 3. Configure watchdog to 60 seconds. 4. Disable SWT if the life cycle state is MCU_PROD or CUST_DEL.	—	—
2	Association	BootROM polls sequentially for activity on the transmission channel until the marker FEED_FACE_CAFE_BEEFh is received. Any incorrect data received is discarded.	8h	Always
3	Non-secure header	BootROM reads the first 40h bytes over the channel and extracts the RAM start pointer, entry pointer, and the code size from the image.	40h	Always

Table continues on the next page...

Table 157. Serial download protocol phases (continued)

No.	Phase	Description	Size (bytes)	Entry
		BootROM also extracts the NO_ECHO field from the most significant bit of the Code length word. **NOTE** • BootROM allows serial download of data only to system RAM. The maximum size of the serial image is limited to the size of the system RAM on the chip. • The image download address must be 64-bit aligned.		
4	Secure header	BootROM reads the next 1200h bytes of the image containing the NSK Public keys, CSK Public key, and the CSK signature.	1200h	Secure boot
5	Code download	BootROM downloads the serial code of specified length over the channel and stores it at the specified location.	Code length	Always
6	Public signature	BootROM reads the Public key signature of the image used for authentication of the image.	100h	Secure boot
7	Execution	After successful download of entire image, the Cortex-M7_0 core is enabled with the reset vector pointing to the Entry pointer. Before enabling the core, the watchdog SWT_0 is enabled with its default configuration. **NOTE** It is the application's responsibility to service the watchdog appropriately.	—	—

When performing a secure boot, after the image is downloaded, BootROM first performs authentication of the image per the Authentication mode selected.

Received packets can be either echoed or not. The NO_ECHO field in the Code length word determines the configuration. By default, all packets are echoed. If the NO_ECHO field is set to 1:

- For a secure boot, starting from the Secure header phase, CAN packets are not echoed.
- For a non-secure boot, starting from the Code download phase, CAN packets are not echoed.

30.11.2.2 Serial download protocol – Ethernet

The serial download protocol over GMAC interface performs the following steps:

1. Identifies MII Interface connected over GMAC_0.
2. Configures PHY and GMAC_0 for Ethernet packet exchange.
3. DHCP client launch:
    a. Chip performs a DHCP request to get an IP address (IPV4).

b. DHCP exchange also retrieves the address of a TFTP server. BootROM expects the TFTP server address to be the same as the DHCP server address.

4. If the DHCP operation was successful, perform a TFTP data exchange:
   a. Fetch SerHeader.img using TFTP protocol from the TFTP Server.
   b. Fetch SerApp.img using TFTP protocol from the TFTP Server.
   c. Fetch SerSign.img using TFTP protocol from the TFTP Server.
5. The image is loaded into SRAM and executed. If the chip is configured for Secure mode, the downloaded image is authenticated before it is executed.

30.11.3 FlexCAN

BootROM supports serial download from the FlexCAN_0 module. Communication between the chip and the host (transmitting utility) is done by exchange of data through CAN packets. BootROM supports transfer of data only in non-FD mode.

30.11.3.1 FlexCAN I/O multiplexer configuration

The I/O pads for FlexCAN communication are configured as shown in the following table.

Table 158. FlexCAN I/O

Signal	Pad name
CAN RX	PAD[43]
CAN TX	PAD[44]

30.11.3.2 Clock configuration

The baud clock for the FlexCAN module is derived from the MC_CGM_0 clock Mux 7 selector, as shown below. BootROM uses XOSC as the clock source for CAN_PE_CLK during boot. If XOSC fails to initialize, FIRC is selected as the MUX source.

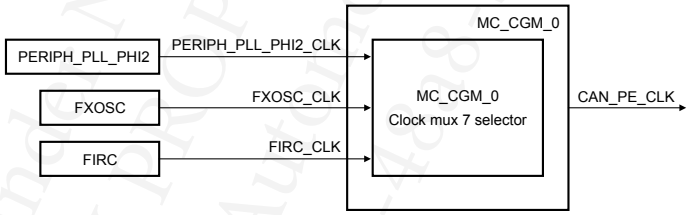


Figure 145. FlexCAN clock configuration

30.11.3.3 Baud rates

The baud rate is determined by two factors: the timing parameters programmed for the module and the baud clock.

The FlexCAN baud clock is derived from XOSC as mentioned in Clock configuration. The timing parameters are programmed to achieve a common bit rate for both clock sources.

For XOSC, for varying frequencies of the crystal, the bit rate achieved is directly proportional to the baud clock.

Table 159. FlexCAN baud rates

Baud clock frequency	Bit rate
20 MHz (XOSC)	0.5 MBd
40 MHz (XOSC)	1 MBd

30.11.3.4 FlexCAN boot operation

The host transfers the application image as per the serial download protocol (see Serial download protocol). The message IDs used by the chip (MCU) for transfer of frames over the FlexCAN interface are defined in the following table. The message IDs are segregated based on the phase of communication.

Table 160. FlexCAN mailbox IDs

Serial download phase	Rx MB ID	Tx MB ID
Association phase	11h	01h
Non-secure header phase	12h	02h
Secure header phase	13h	03h
Code download phase	14h	04h
Public signature phase	15h	05h

NOTE

The payload size of the message transmitted is restricted to 8 bytes.

30.11.4 UART

BootROM supports boot from the LINFlexD_0 module, configured in UART mode. The communication between the MCU and the host (transmitting utility) is done by exchange of data through UART packets. LINFlexD_0 is configured to communicate in 8E1 configuration.

30.11.4.1 UART IOMUX configuration

The IOMUX pads for UART communication are configured as shown in the following table.

Table 161. UART external I/O

Signal	PAD name
RX	PAD[42]
TX	PAD[41]

30.11.4.2 Clock configuration

The clock for the LINFlexD module is derived from the MC_CGM_0 clock Mux 8 selector. BootROM uses XOSC as a clock source for LIN_BAUD_CLK during boot. If XOSC fails to initialize, FIRC is selected as the MUX source.

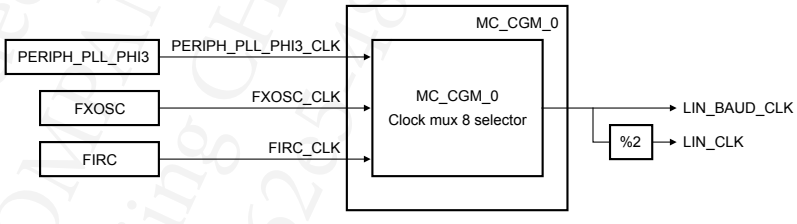


Figure 146. UART clock configuration

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e82-1682-48a8-b251-30d3ab624c

30.11.4.3 UART baud rates

The baud rate is determined by two factors: the timing parameters programmed for the module and the baud clock. The LINFlexD baud clock is derived from either XOSC or FIRC, as shown in Clock configuration. The timing parameters are programmed to achieve a common bit rate for both clock sources.

For XOSC, for varying frequencies of the crystal, the bit rate achieved is directly proportional to the baud clock.

Table 162. LINFlexD baud rates

Baud clock frequency	LINFlexD baud rate
20 MHz (XOSC)	24 kBd
40 MHz (XOSC)	48 kBd
48 MHz (FIRC)	48 kBd

30.11.5 Ethernet boot

BootROM supports serial download over an Ethernet interface. Ethernet boot currently supports RGMII interface.

30.11.5.1 IOMUX configuration

For details on the IOMUX configuration for Ethernet boot, see the BootROM_IOMUX_Settings tab in Boot_Settings_Sheet attached to this document.

30.11.5.2 Functional description

At a high level, Ethernet serial download follows the steps shown below:

1. Ethernet MII interface detection and configuration.
2. PHY identification and configuration.
3. GMAC_0 speed and clock configuration.
4. GMAC_0 controller configuration.
5. MAC address configuration.
6. DHCP client launch.
7. The chip performs a DHCP request to get an IP address (IPV4).
8. The DHCP exchange also retrieves the address of a TFTP server. BootROM expects the TFTP server address to be the same as the DHCP server address.
9. The file is then fetched using TFTP protocol from the TFTP server.
10. The file is loaded into SRAM and executed. If the chip is configured for Secure mode, the code is authenticated before it is executed.

30.11.5.2.1 MII interface detection

For an unprogrammed device, MMI Interface selection is made by a second configuration applied to the BMODE1 and BMODE2 pins.

1. The first configuration is applied before deassertion of the RESET_B signal. It selects the boot mode.
2. The second configuration is applied after RESET_B is deasserted. It selects the PHY according to the table below.

The first configuration is sampled by chip hardware. The second configuration is sampled by BootROM in Serial Boot mode (BMODE combination of 00 and 01) only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIT Automotive Technology Co., Ltd.
c8f6225-4b12-48a8-b251-39013b624c1

Table 163. MII interface detection via BMODE

Before reset		After reset		Mode	Recommended hardware configuration	
BMODE1	BMODE2	BMODE1	BMODE2		BMODE1	BMODE2
0	0	0	0	Serial boot with XOSC in Differential mode No Ethernet PHY	Tie to GND (0 V)	Tie to GND (0 V)
0	0	0	1	Serial boot with XOSC in Differential mode SGMII Ethernet PHY	Tie to GND (0 V)	Tie to RESET_B
0	0	1	0	Serial boot with XOSC in Differential mode RGMII Ethernet PHY	Tie to RESET_B	Tie to GND (0 V)
0	0	1	1	Serial boot with XOSC in Differential mode RMII Ethernet PHY	Tie to RESET_B	Tie to RESET_B
0	1	0	0	Serial boot with XOSC in Crystal mode or Bypass mode No Ethernet PHY	Tie to GND (0 V)	Tie to invert of RESET_B
0	1	0	1	Serial boot with XOSC in Crystal mode or Bypass mode SGMII Ethernet PHY	Tie to GND (0 V)	Tie to HIGH (3.3 V)
0	1	1	0	Serial boot with XOSC in Crystal mode or Bypass mode RGMII Ethernet PHY	Tie to RESET_B	Tie to invert of RESET_B
0	1	1	1	Serial boot with XOSC in Crystal mode or Bypass mode RMII Ethernet PHY	Tie to RESET_B	Tie to HIGH (3.3 V)

For serial boot via fail-safe path, selection of PHY is made via the HSE_GPR_BMR1[1:0] field according to the table below. HSE_GPR_BMR1[1:0] is loaded either via BOOT_CFG[1:0] or RCON[1:0], based upon the FUSE_SEL fuse value.

Table 164. MII interface detection through BOOT_CFG1/RCON

BOOT_CFG[0]	BOOT_CFG[1]	Ethernet PHY
0	0	RMII
0	1	SGMII

Table continues on the next page...

Table 164. MII interface detection through BOOT_CFG1/RCON (continued)

BOOT_CFG[0]	BOOT_CFG[1]	Ethernet PHY
1	0	RGMII
1	1	No PHY

30.11.5.2.2 PHY identification and configuration

BootROM reads the PHYID1 and PHYID2 PHY registers via the MDIO interface to identify the connected PHY. BootROM maintains a list of RGMII PHYs against which these PHYIDn registers are compared. BootROM expects PHYs to start Autonegotiation after every reset and wait for autonegotiation to complete within 5 seconds.

After autonegotiation completes, BootROM reads the link status, negotiated speed, and duplex mode. It then configures GMAC_0 accordingly.

BootROM forces PHY configurations for 1000 Mbps and full duplex mode operation if one of these events occurs:

- Autonegotiation fails within 5 seconds.
- The connected PHY is not in the list of supported PHYs.
- The link status is not OK.

GMAC_0 is also configured for 1000 Mbps and Full Duplex mode. You must ensure there is a stable connection.

BootROM communicates via the MDIO interface for configuring the PHY and uses 1 as the PHY address. It is recommended that the default address of PHY, connected to GMAC_0, be 1.

Currently, BootROM supports identification of the PHYs listed below for status reads after autonegotiation completes:

Table 165. Supported PHYs

PHY	PHYID1	PHYID2
RTL8211F	001Ch	C916h

30.11.5.2.3 GMAC_0 speed and clock configuration

BootROM supports GMAC_0 speeds of 1000 Mbit/s, 100 Mbit/s, and 10 Mbit/s.

BootROM locks the PLL over XOSC for generating the GMAC_0 TX clock. PERIPH_PLL is locked for generating 125 MHz at the PHI5 output. BootROM selects CLK_PERIPH_PLL_PHI5 as the source for MC_CGM_0_MUX_10, which controls the GMAC_0 TX clock.

Depending on the negotiated or configured GMAC_0 speed of 1000 Mbit/s, 100 Mbit/s, or 10 Mbit/s, MC_CGM_0_MUX_10_DIV_0 is programmed for generating the relevant clock of 125 MHz, 25 MHz, or 2.5 MHz. The GMAC_0 RX clock is controlled via MC_CGM_0_MUX_11. It is sourced from GMAC_0_EXT_RX_CLK. For more details, see Boot_Settings_Sheet attached to this document.

30.11.5.2.4 GMAC_0 controller configuration

BootROM configures the GMAC_0 controller for transmission and reception of data with PHY, connected via MII interface. GMAC_0 uses internal DMA channels for transferring data between MAC and application (system RAM). BootROM uses DMA channel 0 for this purpose and uses the first 256 bytes of system RAM (beginning at 34000000h) for preparing the DMA descriptor. Application code must not target this memory location as the download address for Ethernet boot.

30.11.5.2.5 MAC address configuration

This chip provides six bytes of fuse area for programming the device MAC address for GMAC_0 (see MAC0_ADDR[47:0] in the attached fuse map sheet). BootROM reads the MAC address from fuses and uses it as the GMAC_0 source address for packet exchange. In unprogrammed devices, where the MAC address is not programmed and fuses for the MAC Address read 0h,

BootROM creates a MAC Address using the device unique id, which is programmed in fuses. For the first three bytes of the MAC address, the designated OUI number 00:04:9F is used.

Table 166. Unprogrammed device MAC Address 1

MAC[0]	00h
MAC[1]	04h
MAC[2]	9Fh

The next three bytes are prepared with the scheme shown below.

Table 167. Unprogrammed device MAC Address 2

	7	6	5	4	3	2	1	0
MAC[3]	UNIQUE_ID[47:43]					Reserved	UNIQUE_ID[1:0]	
MAC[4]	UNIQUE_ID[55:48]							
MAC[5]	UNIQUE_ID[63:56]							

MAC[0] is sent as the first byte of the MAC address and MAC[5] is sent as the last.

30.11.5.2.6 DHCP handshake

The basic steps that occur when a DHCP client requests an IP address from a DHCP server are shown in the following figure.

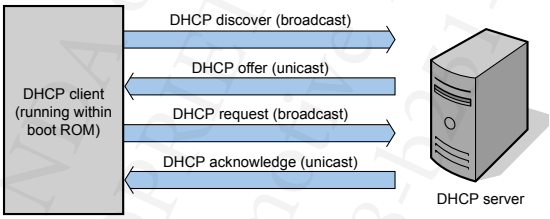


Figure 147. DHCP handshake

1. The client sends a DHCP DISCOVER broadcast message to locate a DHCP server.
2. A DHCP server offers configuration parameters (such as an IP address and a TFTP server IP address).
3. The client returns a formal request for the first offered IP address to the DHCP server in a DHCPREQUEST broadcast message.
4. The DHCP server confirms that the IP address has been allocated to the client by returning a DHCPACK unicast message to the client.
5. The MAC address can be set during the boot after reading from the MAC_ADDR eFuses (if present). Otherwise boot firmware creates a unique MAC address using the NXP OUI and part of the device unique ID stored in fuses.

30.11.5.2.7 TFTP data transfer flow

BootROM implements a TFTP client that supports three types of TFTP messages:

- RRQ (Read request)
- DATA (Data)
- ACK (Acknowledgment)

The following figure illustrates the TFTP handshake process.

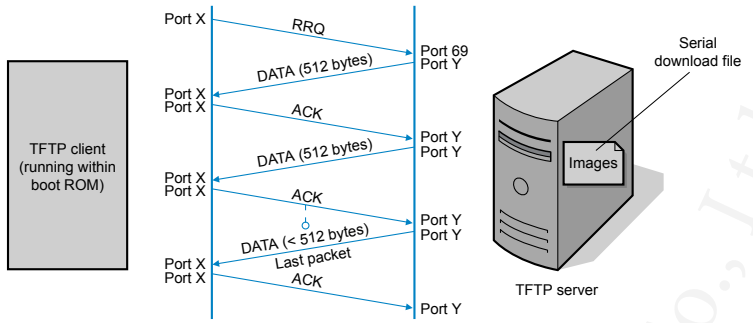


Figure 148. TFTP data transfer flow

The service uses the UDP port 69. Because TFTP uses UDP, it must supply its own session support. Each file transferred via TFTP constitutes an independent exchange.

1. The initiating client (BootROM in this case) sends an RRQ packet, containing the filename and the transfer mode.
2. The TFTP server responds with a DATA packet if it received an RRQ message (this also indicates to the client the ports being used).
3. The sending host then sends numbered DATA packets to the destination host after receiving each ACK message. All but the last message contains a full-sized block of data. The destination host replies with numbered ACK packets for each received DATA packet. This forms a simple ARQ protocol, providing retransmission when a packet is lost.
4. The final DATA packet must contain less than a full-sized block of data (including possibly zero bytes) to indicate that it is the last block of the transfer.

NOTE

- TFTP has no authentication or encryption mechanisms.
- If the chip is configured for Secure mode, the image fetched via TFTP is authenticated before it is executed on the application core.

The following table lists the files downloaded by BootROM from the TFTP server:

Table 168. Files downloaded from TFTP server by BootROM

Name	Description
SerHeader.img	• When doing non-secure serial boot, this file contains the first 64 bytes of the serial image structure. See Serial boot for details. • When doing secure serial boot, this file contains the first 1240h bytes of the serial image structure. See Serial boot for details. **NOTE** BootROM uses the first 8 KB of SRAM memory for its internal operation during serial boot over Ethernet. Care must be taken to ensure that the first 8 KB of SRAM is not used for the serial application.
SerApp.img	Contains the serial boot code.
SerSign.img	BootROM downloads this file only for secure serial boot. This file contains the public key signature. See Serial boot for details.

30.11.5.3 Flow diagram

The following diagram illustrates the Ethernet serial download boot flow.

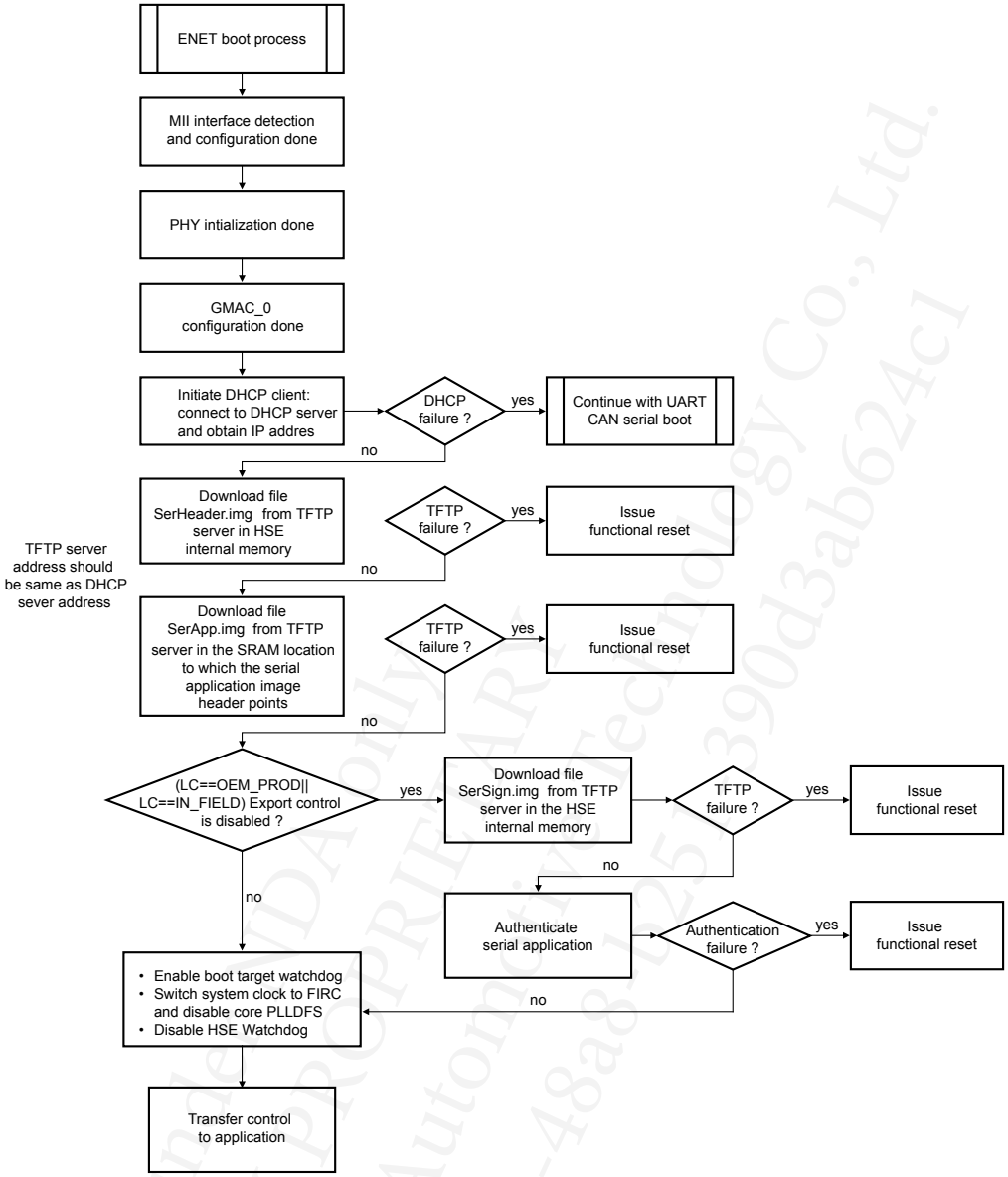


Figure 149. Ethernet serial download boot flow

30.12 Boot from external flash memory

BootROM supports boot from external flash memory devices over the following interfaces:

- QuadSPI
- SD/MMC/eMMC via µSDHC interface

Boot device selection is controlled by Boot_CFG1[7:5] eFuses as shown in the following table.

Table 169. Boot device selection

BOOT_CFG[7:5]	Boot device
000	QuadSPI flash memory

Table continues on the next page...

Table 169. Boot device selection (continued)

BOOT_CFG[7:5]	Boot device
001	Reserved
010	SD
011	MMC/eMMC

30.12.1 QuadSPI boot

BootROM supports booting from the QuadSPI_0 interface:

- Quad and Octal flash memories in 1-bit SDR mode
- HyperFlash flash memories in 8-bit mode

This configuration is only required for initial reads to identify the flash memory. You can reconfigure the interface for optimal performance, based on flash memory configuration parameters provided by customers.

30.12.1.1 Functional description

Boot ROM supports boot from a variety of flash memories, providing flexibility for choosing the configuration parameters for which the controller must be programmed during boot. The configuration parameters can be based on product requirements. For better performance, the QuadSPI controller supports high-speed operations in DDR and SDR modes, which requires specific configurations to achieve correct data sampling at such a high rate. The QuadSPI controller supports reading data over an AHB interface or an IP interface—however, boot ROM supports only read via an AHB interface. Boot ROM does not support any write operations to QuadSPI flash memory.

Boot ROM performs QuadSPI controller configuration in two phases, as described in the following table.

Table 170. QuadSPI controller configuration phases

No.	Name	Description
1	QuadSPI initial configuration	Configure the QuadSPI controller at a frequency of 40 MHz in 1-bit mode for Quad and Octal flash memories, and 8-bit mode for HyperFlash flash memories. BootROM uses BOOT_CFG/RCON based values, table 170, for configuring QSPI controller for applying successful initial configuration.Read the flash memory reconfiguration data as shown in Table 173 located at offset 200h of the flash memory.
2	QuadSPI final configuration	Prepare the clock source per required frequency in reconfiguration data. Reconfigure the QuadSPI controller based on details provided in the QuadSPI reconfiguration data.

You must prepare the QuadSPI reconfiguration data per the flash memory requirement, which is finalized during development. Details of these configurations can be passed either via RCON, during development, or via fuses for the projected use case, when the FUSE_SEL fuse is blown (production use case). For a detailed description, see the boot ROM Fuse description tab in the attached Fuse map spreadsheet. When QuadSPI reconfiguration data is not provided, the QuadSPI final configuration phase is skipped.

Table 171. QuadSPI fuse description

No.	Fuse word offset	Fuse bit position	Fuse name	Settings	Description
1	234h	RCON/ BOOT_CFG1[7:5]	Boot interface	000	Selects QuadSPI as the boot interface.

Table continues on the next page...

Table 171. QuadSPI fuse description (continued)

No.	Fuse word offset	Fuse bit position	Fuse name	Settings	Description
2	234h	RCON/ BOOT_CFG1[4:2]	QuadSPI mode	000: Quad flash memory 001: HyperFlash flash memory 011: Octal flash memory	Selects type of flash memory.
3	234h	RCON/ BOOT_CFG1[9]	QuadSPI port selection	0: Port A 1: Port B	Selects the QuadSPI port on which external flash memory is connected.
4	234h	RCON/ BOOT_CFG1[18:16]	QuadSPI POR delay	000: 300 µsec 001: 600 µsec 010: 1000 µsec 011: 2000 µsec 100: 5000 µsec 101: 50000 µsec 110: 100000 µsec 111: 300000 µsec	A delay is required before the first access is issued to flash memory after flash memory reset.
5	234h	RCON/ BOOT_CFG1[14:11]	QuadSPI CAS	0000-1111: QuadSPI_SFACR[CAS] value	Column address space. Only valid for HyperFlash mode. See the SFACR[CAS] register field description in the QuadSPI chapter.
6	234h	RCON/ BOOT_CFG1[30:29]	DQS selection	00: Reserved 01: Pad loopback 10: Reserved 11: External DQS	Selects DQS clock for sampling read data. • For Quad and Octal flash memories, a setting of 10b is recommended. • For HyperFlash memory a setting of 11b is recommended. See the MCR[DQS_FB_SEL] register field description in the QuadSPI chapter.
7	234h	RCON/ BOOT_CFG1[23:22]	TDH: Time Hold Delay	00: Data aligned at PosEdge of	Serial flash memory data in hold time.

Table continues on the next page...

Table 171. QuadSPI fuse description (continued)

No.	Fuse word offset	Fuse bit position	Fuse name	Settings	Description
				internal reference clock 01: Data aligned with 2X serial flash memory	See the FLSHCR[TDH] register field description in the QuadSPI chapter.
8	234h	RCON/ BOOT_CFG1[25]	FSDLY: Full Speed Delay	0: Same DQS 1: Half cycle early DQS	Full speed delay selection. See the SMPR[FSDLY] register field description in the QuadSPI chapter.
9	234h	RCON/ BOOT_CFG1[24]	FSPHS: Full Speed Phase	0: Select sampling at non-inverted clock 1: Select sampling at inverted clock	Full speed phase selection. See the SMPR[FSPHS] register field description in the QuadSPI chapter.
10	234h	RCON/ BOOT_CFG1[28:26]	DLLFSMPF	N can vary from '0' – '7'	Selects the nth tap provided by the slave delay-chain. See the SMPR[DLLFSMPFB] register field description in the QuadSPI chapter.
11	234h	RCON/ BOOT_CFG1[19]	CKN: Differential clock selector	0: Differential clock not required 1: Differential clock required	Enables clock on differential pad.
12	234h	RCON/ BOOT_CFG1[10]	CK2: CK2 clock selector	0: CK2 clock not required 1: CK2 clock required	Enables 90-degree phase-shifted clock.
13	238h	BOOT_CFG2[12]	SLAVE DELAY EN	1: SLV_DLY_OFFSET and SLV_DLY_COARSE fuses configure slave delay 0: Boot ROM configured value	Enables DLLCRA[SLV_DLY_COARSE] and DLLCRA[SLV_DLY_OFFSET].

Table continues on the next page...

Table 171. QuadSPI fuse description (continued)

No.	Fuse word offset	Fuse bit position	Fuse name	Settings	Description
				configures slave delay	
14	238h	BOOT_CFG2[16:13]	SLV_DLY_COARSE	10000b – 1111b	Sets number of delay elements in each delay tap. See the DLLCRA[SLV_DLY_COARSE] or DLLCRB[SLV_DLY_COARSE] descriptions in the QuadSPI chapter.
15	238h	BOOT_CFG2[20:17]	SLV_DLY_OFFSET	10000b – 1111b	Selects the number of T/16th offset delay elements, tap, in incoming DQS. See the DLLCRA[SLV_DLY_OFFSET] or DLLCRB[SLV_DLY_OFFSET] descriptions in the QuadSPI Chapter.

NOTE

Because information in BOOT_CFG1, at 234h offset, has bit-by-bit correspondence to RCON[31:0], the FUSE_BIT_POSITION column in the table also corresponds to the RCON description.

30.12.1.2 QuadSPI IOMUX configuration

Table 172 shows the QuadSPI IOMUX pin configuration used in BootROM.

BootROM provides an option of overriding some default QuadSPI IOMUX configuration settings implemented in BootROM firmware, and contained in the attached BootROM Settings Sheet, via a pad override scheme. If BOOT_CFG3[9] is enabled, BootROM applies the following IOMUX configuration based on fuse values in BOOT_CFG3. This feature is applicable for both PORT A and PORT B.

Table 172. QuadSPI IOMUX pin configuration

No.	OCOTP fuse word offset	Fuse bit position	Definition	Description
1	23Ch	BOOT_CFG3[9]	PAD_OVERRIDE_EN	Enable pad override for QuadSPI. 0: Disabled 1: Enabled
2	23Ch	BOOT_CFG3[7:5]	QSPI_SRE	Pad slew rate value applies to all QuadSPI pads.

Table continues on the next page...

Table 172. QuadSPI IOMUX pin configuration (continued)

No.	OCOTP fuse word offset	Fuse bit position	Definition	Description
3	23Ch	BOOT_CFG3[4]	QSPI_PUS_CS	Pullup select for all CS pads for chosen QuadSPI interface. 0: Pulldown 1: Pullup
4	23Ch	BOOT_CFG3[3]	QSPI_PUS_DATA[7:0]	Pullup select for all DATA pads for chosen QuadSPI interface. 0: Pulldown 1: Pullup
5	23Ch	BOOT_CFG3[2]	QSPI_PUS_DQS	Pullup select for DQS pad for chosen QuadSPI. 0: Pulldown 1: Pullup
6	23Ch	BOOT_CFG3[1]	RCVR	Value applies to all QuadSPI pads.
7	23Ch	BOOT_CFG3[0]	QSPI_PUS_CK_CK2	Pullup select for all clock pads for chosen QuadSPI. 0: Pulldown 1: Pullup

30.12.1.3 QuadSPI clock configuration

QuadSPI drives the clock from CLK_MUX_12 selector in MC_CGM_0. CLK_MUX_12 selectors can select QSPI_CLK from either IRC or PERIPH_PLL_DFS1, as shown in the following figure.

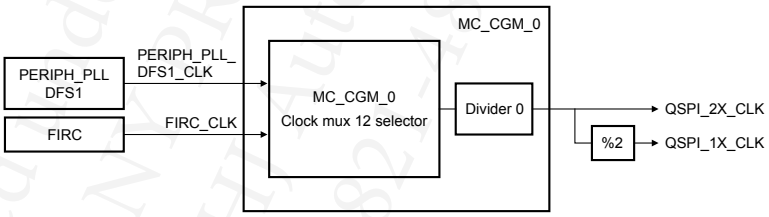


Figure 150. QuadSPI clock selector

In the initial configuration phase, the BootROM clocking scheme for QuadSPI operations depends on the PLL lock status. BootROM tries to lock the PERIPH PLL and PERIPH DFS1 for generating QSPI_1X_CLK at 40 MHz. If PLL-DFS is locked successfully at 40 MHz, then the clock is switched to PLL. Otherwise, the QuadSPI clock is derived from FIRC, resulting in QSPI_1X_CLK being at FIRC/2.

In the final configuration phase, QuadSPI read operation clock requirements are derived by the clock that you provide in the QuadSPI reconfiguration data. In case of failure to generate the required clock, BootROM skips the configuration provided and continues read operations with the default configuration and the QuadSPI clock derived from FIRC.

30.12.1.4 QuadSPI configuration parameters

The following table lists QuadSPI configuration parameters in order, and the required details to be programmed in flash memory at offset 200h.

Table 173. QuadSPI configuration parameters

Name	Offset	Size in bytes	Description
Header	0h	4	Should be 5A5A5A5Ah.
MCR	4h	4	MCR values for targeted operation.
FLSHCR	8h	4	FLSHCR values for targeted operation.
BFGENCR	Ch	4	BFGENCR values for targeted operation.
DLLCR	10h	4	Configuration to be programmed in DLLCRA or DLLCRB register depending on whether flash memory is connected to Port A or B. Only the DLLCR[30:4] bits are used—bit 31 and bits 0–3 are masked out.
PARITYCR	14h	4	Values to be programmed in case flash memory is enabled for parity.
SFACR	18h	4	Holds value to be configured in SFACR register.
SMPR	1Ch	4	Holds value to be configured in SMPR register for Phase 2 configurations.
DLCR	20h	4	Data Learning Control Register configuration required as per desired Phase 2 configurations.
SFLASH_1_SIZE	24h	4	Flash 1 top address, applicable to either Port A or B. Only one port is supported during boot. Port selection depends on the value in BOOT_CFG[9].
SFLASH_2_SIZE	28h	4	Flash 2 top address, applicable to either Port A or B. Only one port is supported during boot. Port selection depends on the value in BOOT_CFG[9].
DLPR	2Ch	4	Data Learning Pattern Register, as applicable for required Phase 2 configurations by user.
SFAR	30h	4	Serial flash memory address to be configured in Phase 2 configuration.
IPCR	34h	4	Content to be programmed in IP Configuration register.
TBDR	38h	4	Data to be written in TBDR register.
DLL_BYPASS_EN	3Ch	1	Based upon DLL_BYPASS_EN configuration, BootROM configures DLL in bypass mode for Phase 2 configurations. You should choose only one DLL configuration, out of the total three, at a time unless stated otherwise. 0: Disable 1: Enable
Reserved	3Dh	1	Reserved

Table continues on the next page...

Table 173. QuadSPI configuration parameters (continued)

Name	Offset	Size in bytes	Description
DLL_AUTO_UPD_EN	3Eh	1	Based upon DLL_AUTO_UPD_EN configuration, BootROM configures DLL in auto update mode for Phase 2 configurations. User should choose only one DLL configuration, out of total three, at a time unless stated otherwise. 0: Disable 1: Enable **NOTE** When this mode is selected, the value of DLL_BYPASS_EN must be 1.
IPCR_TRIGGER_EN	3Fh	1	Writes IPCR field to IPCR register only if IPCR_TRIGGER_EN is 1.
SFLASH_CLK_FREQ	40h	1	User-provided frequency (in MHz) for Phase 2 QuadSPI configuration. This frequency corresponds to the QSPI_1X_CLK clock.
Reserved	41h	1	Reserved
Reserved	42h	1	Reserved
Reserved	43h	1	Reserved
COMMAND_SEQ	44h	320	User-provided LUT configuration to be used for read operations over the AHB interface. The LUT should be programmed as per requirements of the flash memory connected and the mode of operation selected, including clock, DDR, SDR, 1-bit, 4-bit, or 8-bit operation. The LUT sequence to be invoked during a read is controlled by the configuration provided in BFGENCR.
FLASH_WRITE_DATA	184h	120	An array of 10 structures. Each structure contains details of a command and related parameters to be sent to flash memory after Phase 1 and before Phase 2.

For better performance, you may need to reconfigure flash memory using default settings, after the initial configuration phase is over. To do this, boot ROM supports a list of 10 commands (CMDs). Each of these commands and associated data are encoded in 12 bytes. See Table 174 and Table 175 that provide details related to:

- FLASH_WRITE_DATA structure
- Data to be configured
- Status register address for external flash memory configuration

Using the command configuration structure in Table 174:

- You can pass CMD-related information to boot ROM that is used for configuration.
- BootROM parses the FLASH_WRITE_DATA structure array until the CMD field becomes 0.

This structure is only meant to write to flash memory configuration registers, and not perform operations in the flash memory array. Also, you must perform all flash memory configurations through FLASH_WRITE_DATA commands and in 1-bit SPI mode.

Table 174. Word 0 (Bytes 0-3): Command Configuration

31	30–24	23–18	17–16	15–8	7–0
Address valid	Configuration data size	Valid address bits	Pad	Reserved	Command Opcode (from Flash Datasheet)
0 – Do not send address bytes 1 – send address bytes	00 – 0 byte 01 – 1 byte 02 – 2 byte 03 – 3 byte 04 – 4 byte Others values for this field are reserved and must not be used	Number of address bits to be sent (for ex., 6'd24 => 24 address bits required)	00 – 1 bit (SPI) 01 – 2 bit (DUAL) 10 – 4 bit (QUAD) 11 – 8 bit (OCTAL)		
e.g. for Macronix Flash, WREN command doesn't need any address to be sent, but WRCR2 needs 4 bytes of address			As only 1-bit SPI commands are supported, set this to 00	Set this field to 0x00	e.g. for Macronix Flash, WREN command has opcode 0x06, so set this field to 0x06

Table 175. Word 1 (Byte 7-4): External Flash Configuration/Status Register Address

31 - 20	19-0
Reserved, set them to 0x000	Set to the address of the register as per Flash datasheet
	e.g. for Macronix Flash, address of Configuration Register 2 (to set DOPI bit) is 0x00000000, hence set this field to 0x00000

Word 2(Byte 11-8): Data to be sent to Flash along with the command

- 0 to 4 bytes of data
- Store it in Little Endian form in the image

After every reset, BootROM requires connected flash memory to be in a configuration mode in which the flash memory is able to respond to a SPI read command.

- If the application alters the flash memory configuration such that the flash memory requires a reset for it to switch to its default SPI mode, then the application must ensure that flash memory is reset on device reset.
- If the application alters the flash memory configuration such that flash memory does not revert to SPI mode even after a reset, then the application must either ensure that flash memory enters SPI mode before every reset, or must use CMDs that do not require such a mode change.

30.12.1.5 QuadSPI boot flow diagram

The following figure illustrates the QuadSPI boot flow.

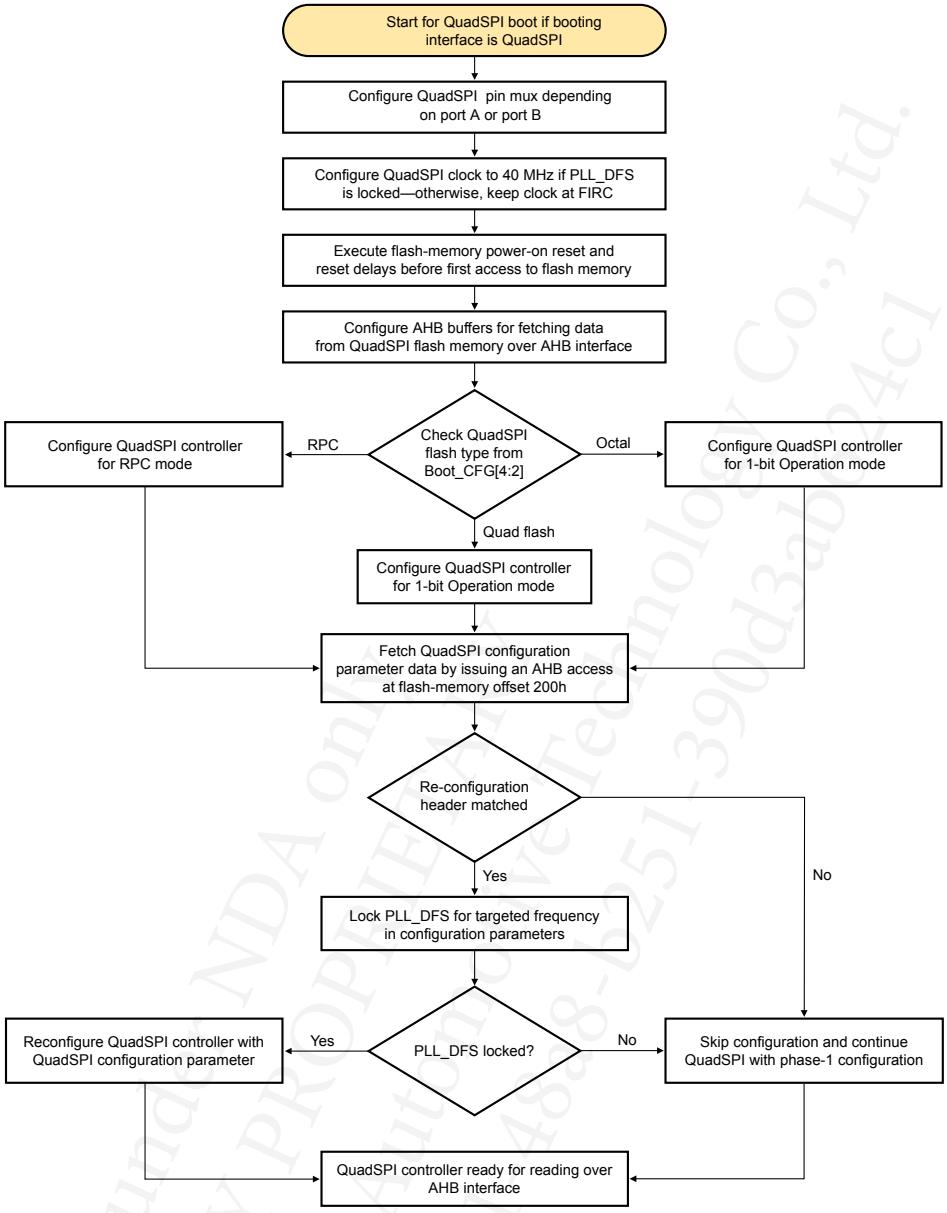


Figure 151. QuadSPI boot flow

30.12.2 µSDHC boot

BootROM supports booting from the SDHC interface. The SDHC controller initial configuration is based on information provided via eFuse or RCON GPIO pins. BootROM supports boot from SD, MMC, and eMMC cards in different data rate modes. BOOT_CFG1[7:5] controls the selection of SD card for boot over the µSDHC interface. For eFuse information on SD/MMC card support, see the Fuse Map in the attached spreadsheet.

30.12.2.1 µSDHC IOMUX configuration

BootROM provides an option of overriding some of the default µSDHC IOMUX configuration settings in BootROM firmware, shown in the attached BootROM settings spreadsheet, via a pad override scheme. If BOOT_CFG3[9] is enabled, BootROM applies the following IOMUX configuration, based on fuse values in BOOT_CFG3.

Table 176. µSDHC IOMUX configuration

No.	OCOTP FUSE word offset	Fuse bit position	Definition	Description
1	23Ch	BOOT_CFG3[9]	PAD_OVERRIDE_EN	Enable pad override for SDHC. 0: Disabled 1: Enabled
2	23Ch	BOOT_CFG3[8]	SDHC_PUS_CMD	Pull up select for CMD I/O PAD. 0: Pull down 1: Pull up
3	23Ch	BOOT_CFG3[7:5]	SDHC_SRE	Pad slew rate value applies to all SDHC pads.
4	23Ch	BOOT_CFG3[4]	SDHC_PUS_RST	Pullup select for all RST pads 0: Pulldown 1: Pullup
5	23Ch	BOOT_CFG3[3]	SDHC_PUS_DATA[7:0]	Pullup select for all SDHC DATA pads interface. 0: Pulldown 1: Pullup
6	23Ch	BOOT_CFG3[2]	SDHC_PUS_DQS	Pullup select for SDHC DQS pad. 0: Pulldown 1: Pullup
7	23Ch	BOOT_CFG3[1]	RCVR	Value applies to all SDHC pads.
8	23Ch	BOOT_CFG3[0]	SDHC_PUS_CLK	Pullup select for clock pad. 0: Pulldown 1: Pullup

30.12.2.2 µSDHC clock configuration

SDHC_CLK is derived from clock MUX 14 in CGM_0. SDHC_CLK can be derived from either FIRC or PERIPHRAL_PLL_DFS3. As per the clocking scheme, BootROM tries to lock the Peripheral PLL-DFS and, in case of success, switches the SDHC_CLK from FIRC to PLL.

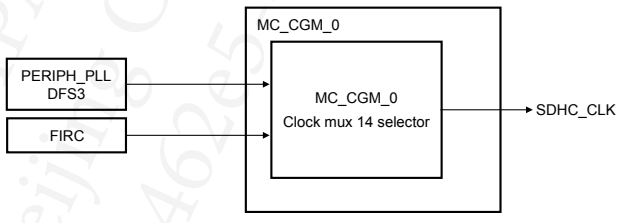


Figure 152. µSDHC clock configuration

30.12.2.3 µSDHC supported data rates

Only 3.3 V high speed and default speed data rate operations are supported when booting from SD cards. BootROM does not support switching to 1.8 V for SDR modes. During Card Identification mode, the clock is kept below 400 KHz. All SDSC, SDHC, and SDXC cards support these data rates. The application can switch to SDR modes after control is passed to it.

Table 177. SD supported boot data rates

Speed modes	Maximum baud rate	Clock speed	Signal voltage
Identification mode	No data transfer	~380 kHz	3.3 V
Default speed	12.5 MB/s	25 MHz	3.3 V
High speed	25 MB/s	50 MHz	3.3 V

BOOT_CFG1[19] controls the selection of Default Speed or High Speed mode when the SD card is used over the µSDHC interface.

For MMC/eMMC cards, BootROM supports the speed modes listed in the following table.

Table 178. Data rates supported for MMC card boot

Speed modes	Maximum baud rate	Clock speed	Bus width
High speed	25 MB/s	25 MHz	1, 4, or 8 bits
Normal speed	50 MB/s	50 MHz	1, 4, or 8 bits
Normal DDR	100 MB/s	50 MHz	4 or 8 bits

BOOT_CFG1[22:19] controls the selection of speed modes when an MMC/eMMC card is used over the µSDHC interface

30.12.2.4 µSDHC boot flow diagrams

30.12.2.4.1 µSDHC boot flow – startup

The following figure illustrates the startup portion of the µSDHC boot flow.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a6-b25...590d8ab6b4c1

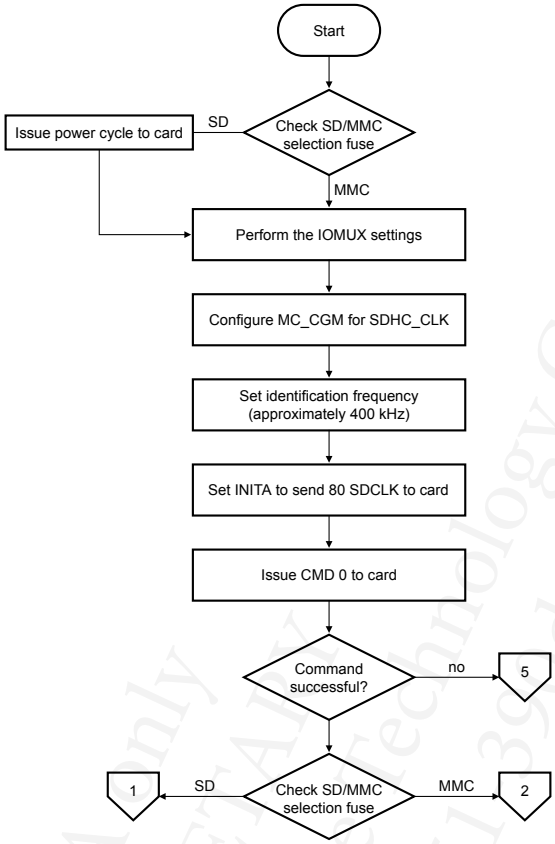


Figure 153. µSDHC boot flow – startup

30.12.2.4.2 µSDHC boot flow – voltage validation

The following figure illustrates the voltage validation portion of the µSDHC boot flow.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a2-b251-39d3ab624c1

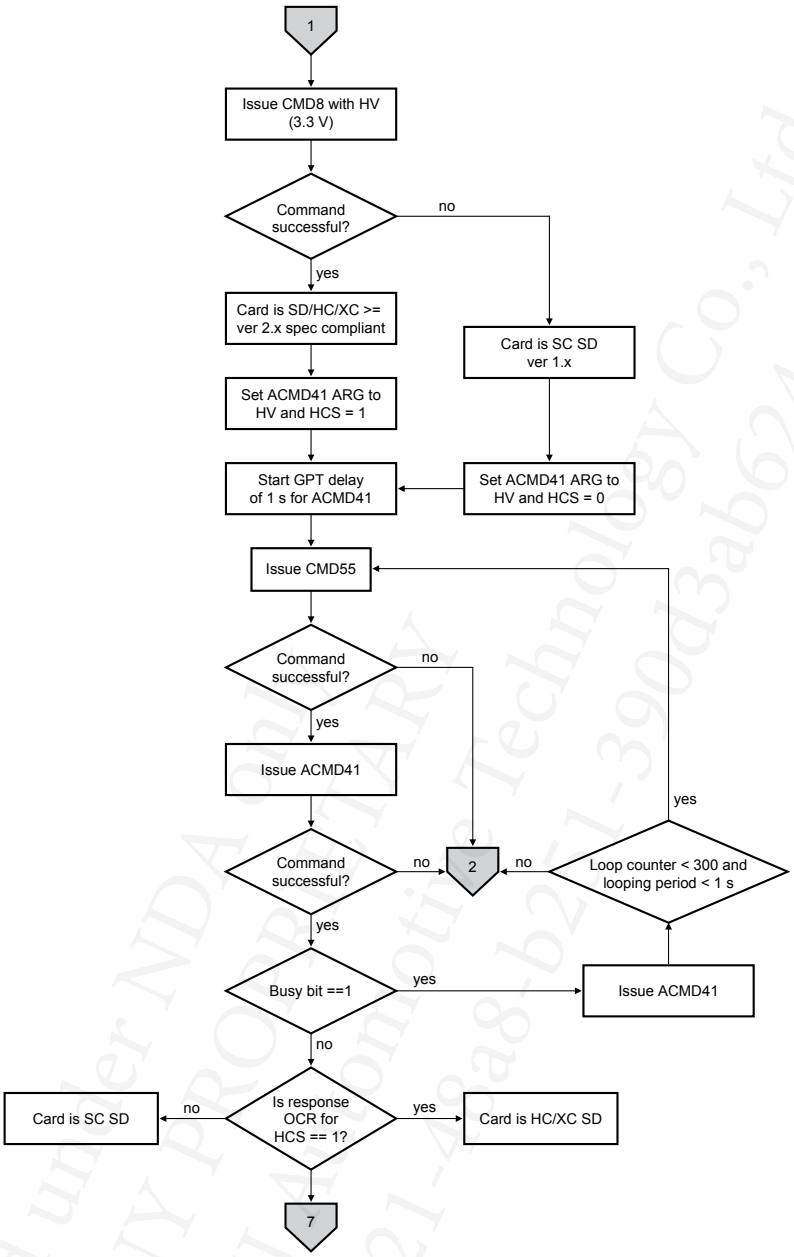


Figure 154. µSDHC boot flow – voltage validation

30.12.2.4.3 µSDHC boot flow – chip initialization

The following figure illustrates the chip initialization portion of the µSDHC boot flow.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-4821-4a8-b251-390d3ab624c1

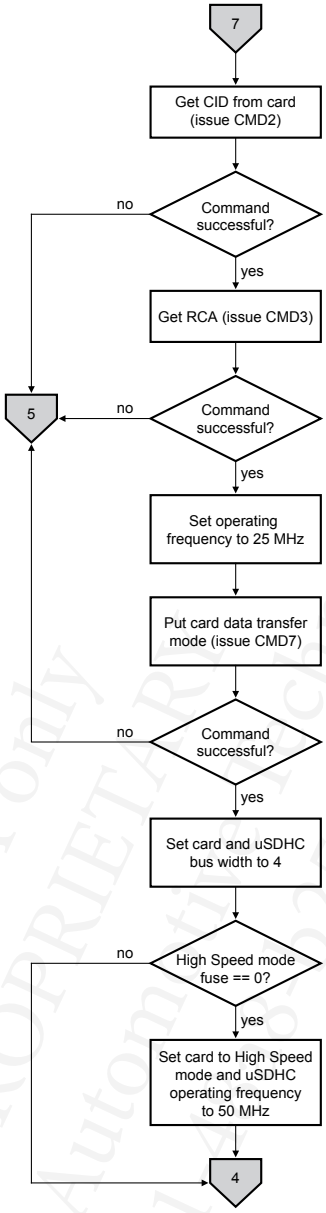


Figure 155. µSDHC boot flow – chip initialization

30.12.2.4.4 µSDHC boot flow – data read

The following figure illustrates the data read portion of the µSDHC boot flow.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e_4821_8251-390d3ab624c1

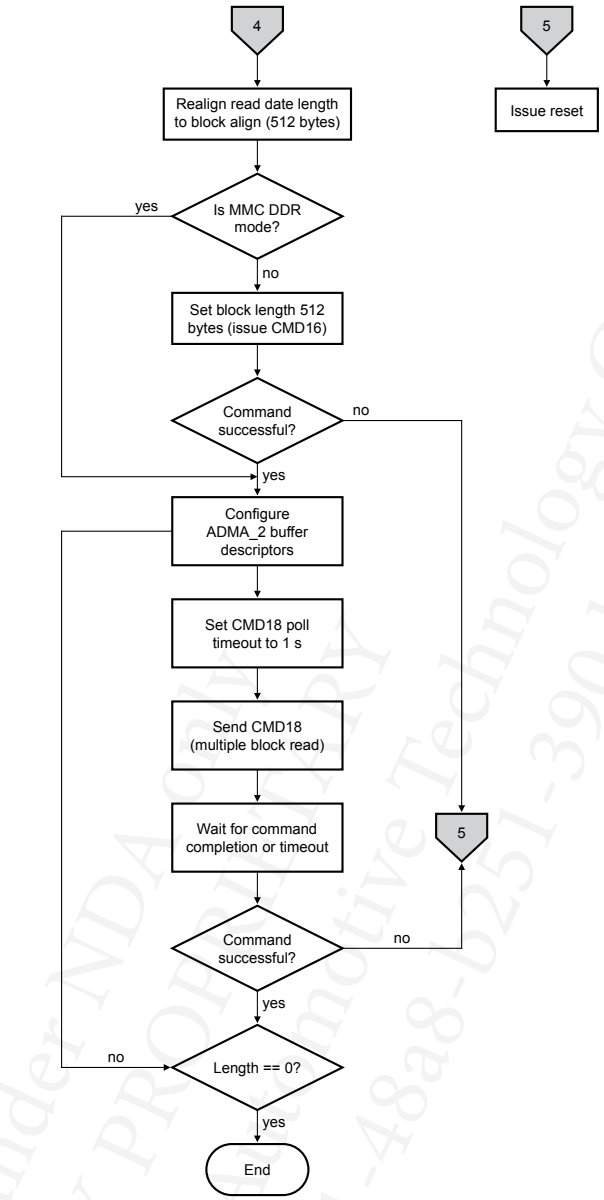


Figure 156. µSDHC boot flow – data read

30.12.2.4.5 µSDHC MMC boot device initialization

The following figure illustrates µSDHC MMC boot device initialization.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing ESWIN Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

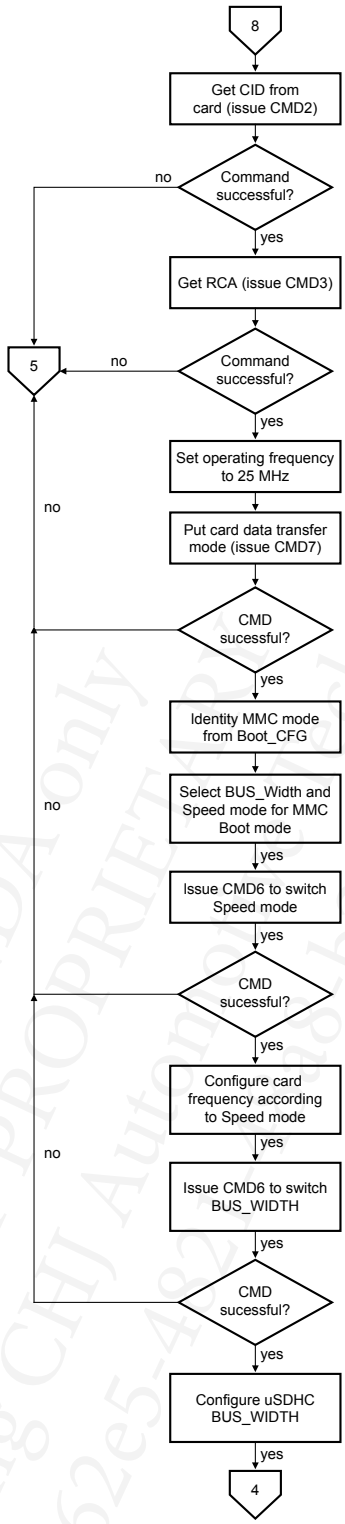


Figure 157. µSDHC MMC boot device initialization

30.12.2.4.6 µSDHC MMC boot voltage validation

The following figure illustrates µSDHC MMC boot voltage validation.

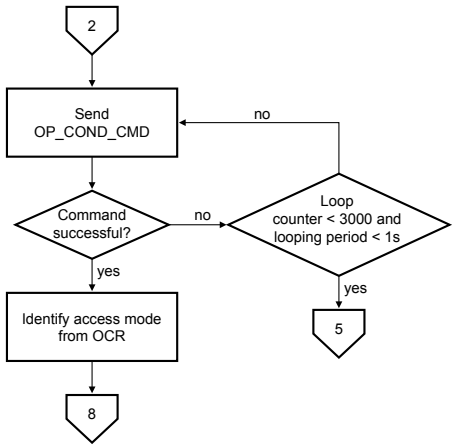


Figure 158. µSDHC MMC boot voltage validation

30.13 Fail-Safe

Errors may be encountered during BootROM execution—software programming errors during development, flash memory failure, PCB failure during production, timeout due to failure of hardware modules, authentication failure during secure boot, exceptions during BootROM execution, and so on. BootROM issues a functional reset in all error conditions.

When BootROM issues repeated functional resets due to a persistent error condition, it is unable to pass control to the application. To manage this scenario, BootROM checks the number of functional resets issued. If ≥ 8, BootROM enters Serial Boot mode to download application code via supported serial interfaces. This mechanism is called Fail-Safe. See Figure 142 for condition RESETs >= 8.

NOTE

The application should check for reset sources and the number of functional resets in the starting instructions and make a local copy for further reference. It should then clear these registers in MC_RGM. Otherwise, persistent resets caused by the application (not from BootROM), will force the BootROM firmware to enter Fail-Safe even though the application can be successfully booted.

30.14 Boot target

The *boot target* is the core responsible for running application code. You can choose to start execution of your application on either the Cortex-M7_0 or Cortex-A53_0 core. For safety applications, the boot target must be Cortex-M7_0. The boot target is enabled only in the default configuration. For lockstep Cortex-A53 configuration, the boot target must be Cortex-M7_0, and the application can then start the Cortex-A53_0 in lockstep mode. Application code running on the boot target can start on any other core(s) in any sequence. To set the target application core, provide the settings in the IVT Boot configuration word (see IVT boot configuration word). An invalid value triggers a functional reset.

30.15 Boot target watchdog

The *Boot target watchdog*, SWT_0, is the watchdog instance responsible for the application code running on the boot target. You can enable the watchdog via setting a field in the IVT Boot configuration word (see IVT boot configuration word for details). When BOOT_SEQ = 0 (in the Boot configuration word) and the watchdog is configured to be enabled in the IVT Boot configuration word, BootROM enables the boot target watchdog before enabling the boot target, with its reset configuration of FIRC at 7601632 cycles. When BOOT_SEQ = 1, HSE_H firmware configures the boot target watchdog. When the watchdog is enabled, the customer application must service it in a timely manner. Otherwise a functional reset occurs.

30.16 Standby boot

Exit from Standby mode triggers execution of BootROM. Depending on the wakeup source, BootROM boots from either the Standby IVT or performs a full boot, as after any reset.

The Standby IVT has the same format as the normal IVT (see Image Vector Table (IVT)), but its location is fixed at start of standby RAM (24000000h). When booting via the Standby IVT, BootROM only supports:

- DCD operations (from primary and backup)
- Application boot (from primary and backup)

The Standby IVT does not support Self-Test or HSE_H firmware. It only supports (IVT boot configuration word field) BOOT_SEQ = 0, which indicates a non-secure boot is to be performed. If BOOT_SEQ = 1, BootROM issues a reset.

When booting from the Standby IVT, BootROM does not support fetching of data from either QuadSPI or an SD/MMC/eMMC card.

Before entering Standby mode, the application should program:

- The IVT (at 24000000h)
- DCD (if required)
- Application on standby RAM, whose contents remain valid during Standby mode

All pointers in the IVT must point to addresses within standby RAM. BootROM does not perform any authentication on the Standby IVT, DCD, or application when booting via the Standby IVT.

When a full boot is requested, BootROM performs normal boot operation, as after any reset, as detailed in this chapter. BootROM does not check the Standby IVT at the start of standby RAM. For more details on wakeup sources,see the Wakeup Unit (WKPU) chapter.

The decision to perform a full boot or boot from the Standby IVT, is based on the wakeup source that triggers the wakeup from standby. If the wakeup source is configured to perform a full boot, BootROM performs a full boot. Otherwise, it boots from the Standby IVT.

If more than one wakeup source is latched in wakeup status, and one of the wakeup sources is configured for full boot, then a full boot is performed.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-3f0d3ab624

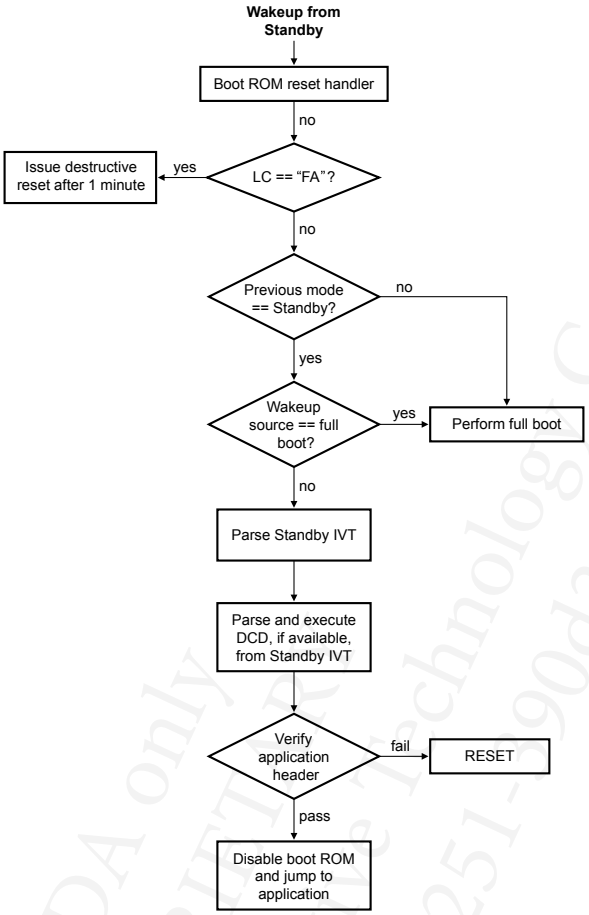


Figure 159. Standby boot flow diagram

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-590d3ab624c1

Chapter 31
Power Management

31.1 Introduction

This chapter describes the power management architecture. It also provides details for the necessary chip voltages and how to control them.

31.1.1 Power management functional blocks

Table 179. Power management functional blocks

Module	Description
MC_ME	Mode Entry module Manages: • On/off control of blocks and subsystems • Standby mode entry requests • Software reset requests.
MC_PCU	Power Control Unit Manages: • Standby mode entry after MC_ME request • Standby mode exit.
MC_RGM	Reset Generation Module Manages full-chip and post-wake-up reset sequences.
PMC	On-Chip Power Management Controller • Controls Non-Critical Supply Presence Detectors (NCSPDs). • Manages power-on reset mechanism.
PMIC	Off-Chip Power Management IC The off-chip power management circuit may be a single IC or a combination of components, and may contain associated safety and low-power mode infrastructure.
WKPU	Wakeup Unit Manages external NMIs and Standby mode wake-up sources.

31.2 Power management features

The power management provides the following features:

- Interfaces to an external SBC (PMIC), providing all chip voltages
- Provides hardware-controlled entry into Run mode
- Enables software-controlled entry into Standby mode
- Enables software-controlled disabling of partitions to reduce power consumption during Run mode
- Provides wake-up event management for exit from Standby mode

31.3 Power-related I/Os

Table 180. Power-related I/O

Name	I/O type	Logic (0.8 V) pullup/pulldown	Muxable	Description
PMIC_STBY_MODE_B	Output	—	No	Active-low Standby mode indicator, causes PMIC to disable all switchable power supplies except the Standby mode core supply and 3.3 V required to power the POR and standby pads. The selection of which supply remains ON in standby mode depends on the programming done to the PMIC registers.
PMIC_VDD_OK	Input	—	No	Output from PMIC indicating the core supply voltage is in the specified operating range, to enable Standby mode exit process to proceed.
PMIC_SENSE	Output	—	No	Sense for core supply (0.8 V)
NMI	Input	—	No	External non-maskable interrupts
WKUP*n*	Input	—	Yes	External wake-ups that can cause an exit from Standby mode (where '*n*' = 0 to 22)
POR_B	Input	Pulldown	No	Startup reset pin indicating supplies are within their specified operating range

NOTE

See the attached IOMUX spreadsheet for details on power-related I/Os and supply pins.

The package pins/balls and corresponding chip pads that supply each analog block in the list below are independent (for example, they are not shared and not adjacent to each other):

- PMC
- FIRC
- FXOSC

This does not preclude that the supplies may be shared at the board and/or PMIC level.

31.4 Operating modes

The power management architecture supports three modes of operation:

- Reset mode
  - Entire chip is powered.
  - Reset mode is the initial mode of the chip. Run mode is entered after Reset mode.
- Run mode
  - Entire chip is powered.
  - Power consumption is managed by clock gating unused processing blocks and peripherals.
  - This mode is always entered on exit from reset and wake-up from Standby mode.
- Standby mode
  - Only a portion of the chip is powered.

— This mode is supported in chips requiring ultra-low power. This mode is required when the chip enters a very low-power state caused by PMIC programming that shuts off supplies. It also enters the mode by following the protocol for Standby entry and exit scenarios.

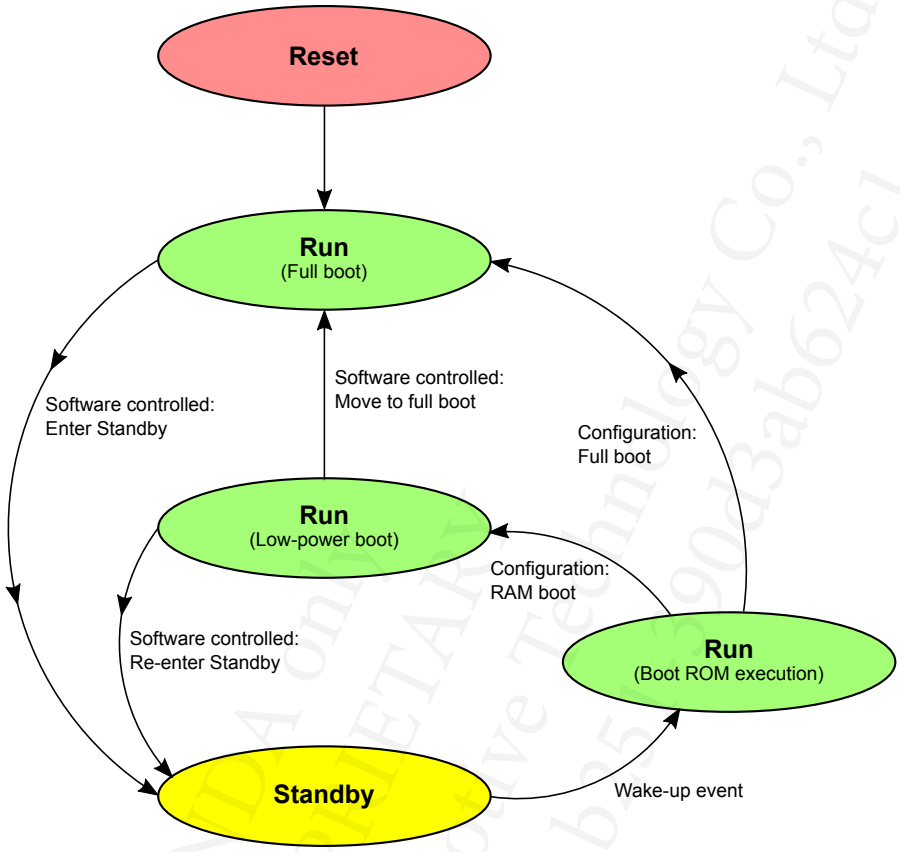


Figure 160. MC_ME supported modes

31.4.1 Reset mode

Reset mode has the following attributes:

- The chip is in a safe state.
- All pins are in their default safe states.
- All computing and communication processes are terminated.
- The chip is fully powered, if not previously fully turned on, on Reset mode entry.
- The chip transitions to its default state prior to enabling Reset mode exit.

The chip always enters Run mode after Reset mode exit (see the "Reset" chapter for related details).

31.4.2 Run mode

Run mode has the following attributes:

- Run mode is the only mode during which the chip executes computational and communications functions.
- The chip is fully powered.
- The boot core is the only core enabled on Run mode entry.

- Software can enable/disable cores and subsystems as needed by the application, based on functional and power requirements.
- Software can configure the device pins needed by the application.
- Software can set up and trigger BIST execution as needed by the application:
  - Main Reset domain:
    - Device pins are placed in a safe state.
    - No computational or communication activities are possible.
    - After BIST completes, Reset mode is entered automatically.
  - Software Reset domain: Some computational and communication activities can run when BIST is running.

31.4.3 Standby mode

Standby mode is the lowest power consumption mode of the chip. Only a minimal portion of chip logic is powered during Standby mode.

Operations permitted during Standby mode are:

- Execution shutdown sequence
- Storage of data needed after Standby mode and store data during standby mode
- Generation of wake-up events
- Detection and response to reset and wake-up events
- Execution of the startup sequence.

31.4.3.1 Standby mode modules

Table 181. Standby mode modules

Module	Description
MC_PCU	Manages power-up/power-down sequencing.
MC_RGM	Manages resets occurring during Standby mode and the reset sequences associated with Standby mode entry/exit.
PMC	Generates a POR on any of the following during Standby mode: • POR_B assertion • CSPD_STANDBY_0P8 indicates loss of voltage on VDD_STBY
RTC	Optionally generates a periodic wake-up event.
SIRC	Provides the clock used by the RTC.
OCOTP	Retains the fuse values (avoids the need to read fuses after each Standby mode exit).
LC_CHECK	Detects any corruption of life cycle information provided by OCOTP.
STDBY_RAM	Stores data that must be intact after Standby mode exit.

NOTE

All supplies switched off in Standby mode must have low impedance to ground. The supplies must discharge to less than 100 mV before making the PMIC_VDD_OK pin low. The requirement is to keep impedance less than 50 Ω.

All wake-up sources, including wake-up pins, are functional and powered during Standby mode.

31.4.3.2 High-level Standby mode power-down/power-up flow diagram

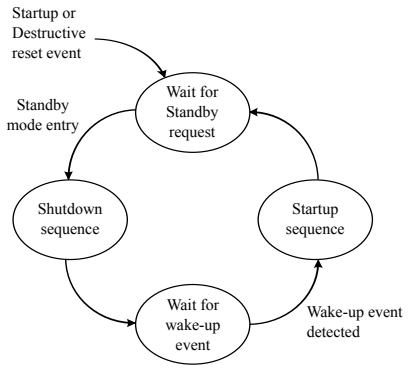


Figure 161. High level Standby mode power up/down sequence flow diagram

31.5 Operating mode transitions

31.5.1 Reset mode to Run mode

The transition from Reset mode to Run mode occurs automatically on any reset sequence exit.

See the "Reset" chapter for details on the reset sequence portion of this mode transition.

31.5.2 Configuring clocks

See the "Clocking" chapter for clock configuration details.

31.5.3 Turning on cores and partitions

See the "Mode Entry Module (MC_ME)" and "Reset" chapters for details.

31.5.4 Run mode to Reset mode

Numerous sources can trigger a transition to Reset mode.

See the "Reset" chapter for details on the reset sequence portion of this mode transition, and for a list of sources that can trigger a reset.

31.5.5 Run mode to Standby mode

During the transition from Run mode to Standby mode, software uses the I^2C or SPI module to communicate that the transition of the chip to Standby mode is in progress. Communicating the transition ensures that the PMIC is aware that the chip is entering Standby mode and prevents undesired Standby mode entry sequence triggers.

31.5.5.1 Standby mode entry flow

After the chip notifies the PMIC that Standby mode entry is pending by writing to the Standby Time Window register in the PMIC, the PMIC ignores all further communication bus activity until one of the following events occurs:

- The Standby entry sequence has completed, followed by the Standby exit sequence having completed.
- The PMIC's Standby mode watchdog timer has timed out.
- The reset sequence completes successfully after a reset event is indicated by deassertion of the RESET_B pin.

Processed by IDA on NXP COMPANY PROPRIETARY Beijing CU Automotive Technology Co., Ltd. ca46e521-21a8-4a8-b251-390d3ab624c1

31.5.5.1.1 Standby mode entry flow diagram

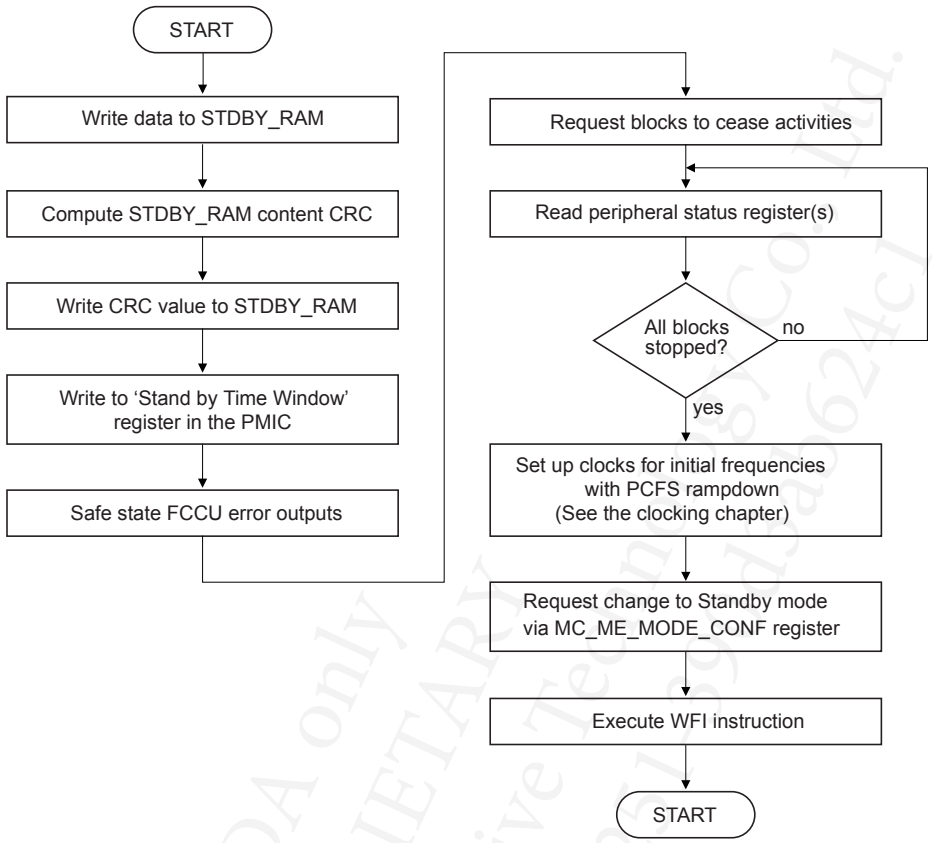


Figure 162. Standby mode entry flow diagram

See Standby mode to Run mode for Standby mode exit flow.

31.5.6 Standby mode to Run mode

Only a wake-up event triggers a transition from Standby mode to Run mode. WKPU manages all wake-up events.

If a wake-up event occurs while a transition from Run mode to Standby mode is in progress, the current transition completes. Only after the transition has completed does the wake-up event trigger a transition back to Run mode.

31.5.6.1 Standby mode exit flow

MC_PCU ignores wake-up and functional reset events that occur during shutdown sequencing. The chip retains the events, then MC_PCU evaluates them. This evaluation occurs one SIRC clock cycle after PMIC confirmation that the run power domain has successfully shut down.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJ Automotive Technology Co., Ltd.
c8c462e5-8248-b2...

31.5.6.1.1 Standby mode exit flow diagram

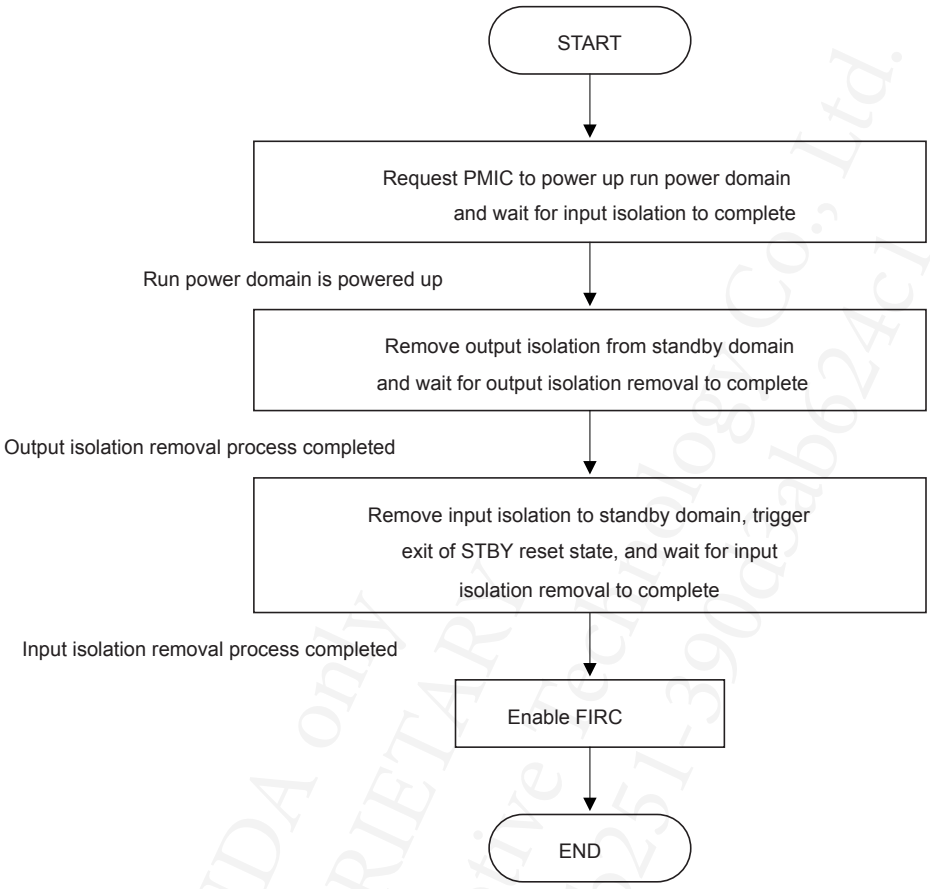


Figure 163. Standby mode exit flow diagram

See the "Reset" chapter for details on the reset sequence portion of this mode transition.

31.5.7 Standby to Reset mode

There are numerous reset sources that can trigger a transition to Reset mode.

A reset source available in Standby mode can trigger a transition to Reset mode.

The actions listed below occur automatically when a power-on reset event occurs at any time during Standby mode or during the transition to/from Standby mode:

- Immediate abort of any ongoing Standby mode transition step
- Deassertion of the STANDBY_MODE_B signal to the PMIC
- Transition to the startup sequence by MC_RGM.

See the "Reset" chapter for a list of reset sources and related details.

31.5.8 Wake-up pin pullup/pulldown control during Run and Standby modes

SIUL2, WKPU, and S32G_STBY_GPR modules manage the pull control of the WKUP*n* pins.

To configure pullup/pulldown of WKUP*n* pins:

1. Write zero to WKUP_PUS[WKUP_PU_OVERRIDE] of S32G_STBY_GPR during Run mode, so that SIUL2 has full control over the pad pullup/pulldown for each WKUP*n* pad (see Table 182).

2. Write one to WKUP_PUS[WKUP_PU_OVERRIDE] of S32G_STBY_GPR before entering Standby mode, to give WKPU control. It can remain one if the application is only spending a short time in Run mode between Standby mode periods (for example, short wake-up events) (see Table 183).

Table 182. Wake-up pin pullup/pulldown during Run mode (WKUP_PUS[WKUP_PU_OVERRIDE] = 0)

Desired pull	SIUL2_MSCR*n*[PUE]	WKPU_WIPUER[IPUE[*p*]]	SIUL2_MSCR*m*[PUS]	S32G_STBY_GPR WKUP_PUS[WKUP_PUS]
None	0	x	x	x
Pullup	1	x	1	x
Pulldown	1	x	0	x

Table 183. Wake-up pin pullup/pulldown during Standby mode[1] (WKUP_PUS[WKUP_PU_OVERRIDE] = 1)

Desired pull	SIUL2 MSCR*n*[PUE]	WKPU WIPUER[IPUE[*p*]]	SIUL2 MSCR*m*[PUS]	S32G_STBY_GPR WKUP_PUS[WKUP_PUS]
None	x	0	x	x
Pullup	x	1	x	1
Pulldown	x	1	x	0

1. Configure before entering Standby mode.

NOTE

Enabling pullups on the wake-up pins during Standby mode increases overall Standby mode power consumption.

NOTE

It is not valid to configure the wake-up pad from SIUL2 and write zero to WKUP_PUS[WKUP_PU_OVERRIDE] of the S32G_STBY_GPR before entering the Standby mode.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab024c1

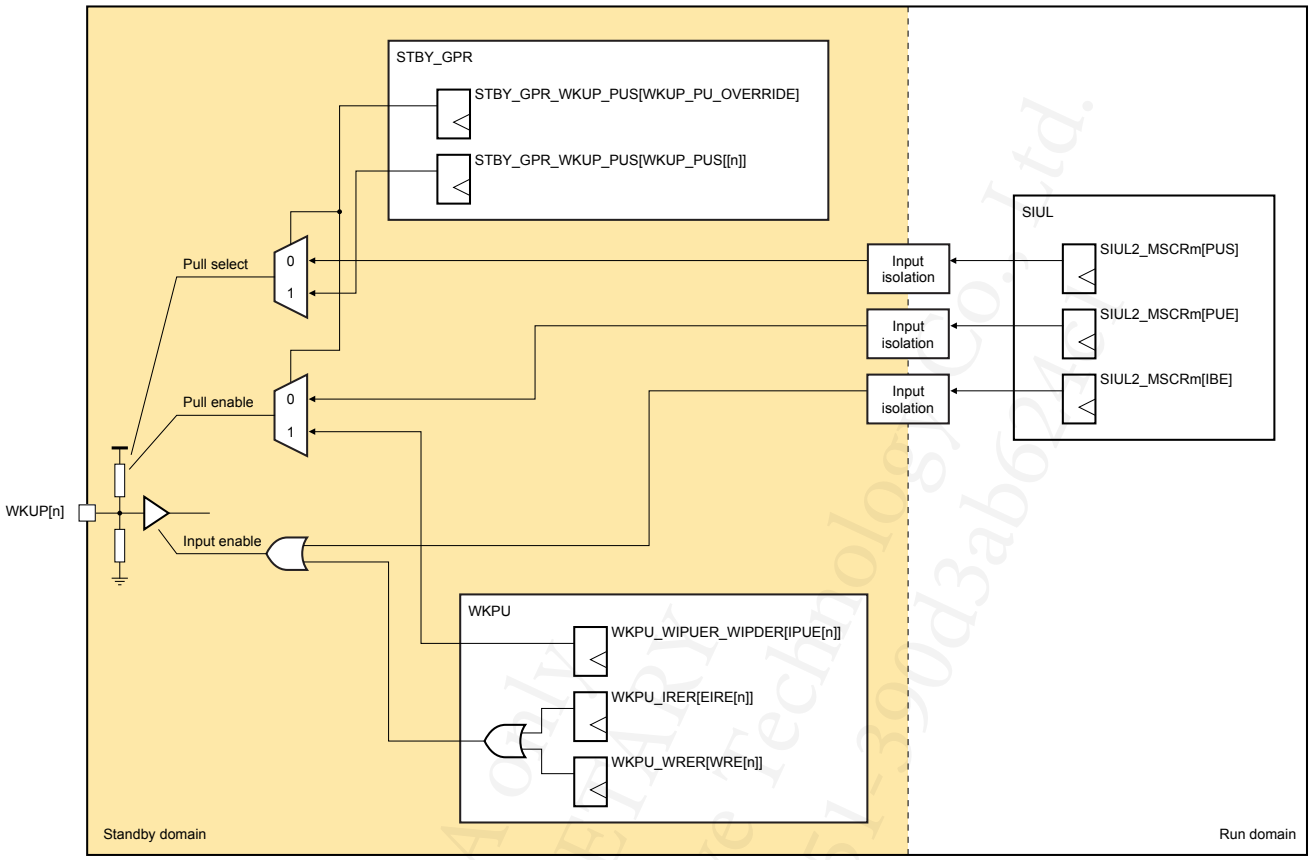


Figure 164. Wake-up pin pullup/pulldown control

31.6 Operating mode control blocks

31.6.1 MC_ME

MC_ME manages the hardware-driven portions of the mode transition processes as follows (see the "Mode Entry Module (MC_ME)" chapter details):

- Provides control and status registers to manage enabling/disabling of partition and core clocks.
- Provides control and status registers to manage enabling/disabling of partition output isolation.
- Provides a set of core address registers providing cores with the jump addresses to application code.
- Provides a mechanism for changing the chip mode from Run mode to Standby mode or Reset mode via software triggering.

MC_ME signals MC_RGM to enter the start of its Standby mode reset sequence. This signal is asserted immediately after software successfully requests a mode change to Standby mode.

31.6.2 MC_PCU

MC_PCU manages shutdown sequencing on entry to Standby mode and startup sequencing on exit from Standby mode (see the "Power Control Unit (MC_PCU)" chapter for details).

31.6.3 MC_RGM

MC_RGM manages power domain resets during Run mode, of both the shutdown sequence and startup sequence (see the "Reset Generation Module (MC_RGM)" chapter).

After entering Standby mode reset sequence, MC_RGM provides a signal to MC_PCU that triggers MC_PCU to start the shutdown sequence.

MC_RGM provides a signal to MC_PCU that triggers MC_PCU to begin startup sequencing. This signal is asserted immediately after MC_RGM detects any functional reset event.

31.6.4 RTC

RTC is a real-time counter that runs on the SIRC clock. You can configure it to generate a periodic interrupt or Standby mode wake-up event (see the "Real Time Clock (RTC)" chapter for details).

31.6.5 WKPU

WKPU provides a mechanism for managing the Standby mode wake-up sources:

- Up to 31 external wake-up pins (WKUP*n* signals in IOMUX spreadsheet)
- A single wake-up from RTC

You can also configure the wake-up sources to trigger an interrupt during Run mode.

WKPU also manages NMI pins. NMI pins do not function as a Standby mode wake-up source.

When an NMI event occurs, software determines the NMI pin that caused the event, then whether an NMI event overflow has occurred on that pin, by reading WKPU_NSR. Software clears each status flag by writing 1 to the corresponding field.

The wake-up outputs of the WKPU connect to the wake-up input of the MC_PCU.

After exiting the Standby mode wake-up sequence, software determines what caused the wake-up by reading WKPU_WISR. Similarly, after a WKPU interrupt event, software determines what caused the interrupt by reading the same register.

Software can clear each individual status flag by writing 1 to the corresponding field. Reading the fields in WKPU_WISR determines the cause of the wake-up:

- Flag 31 equals one indicates an RTC counter rollover.
- If any of the other flags equal one, the cause was an external wake-up event on the corresponding wake-up pin.

After exiting the Standby mode wake-up sequence, software can determine whether to load the low-power boot image or the full boot image by reading WKPU_WBMSR.

Figure 165 shows the wake-up and interrupt mapping of WKPU. The blocks labeled "glitch filter" represent fields in WKPU_WIFER or WKPU_NCR. The blocks labeled "edge det" represent a field in WKPU_WIFEER (riding edge), WKPU_WIFEEF (falling edge), or WKPU_NCR.

Provided by IHS Markit
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482f-48a8-b241-90d3ab624c1

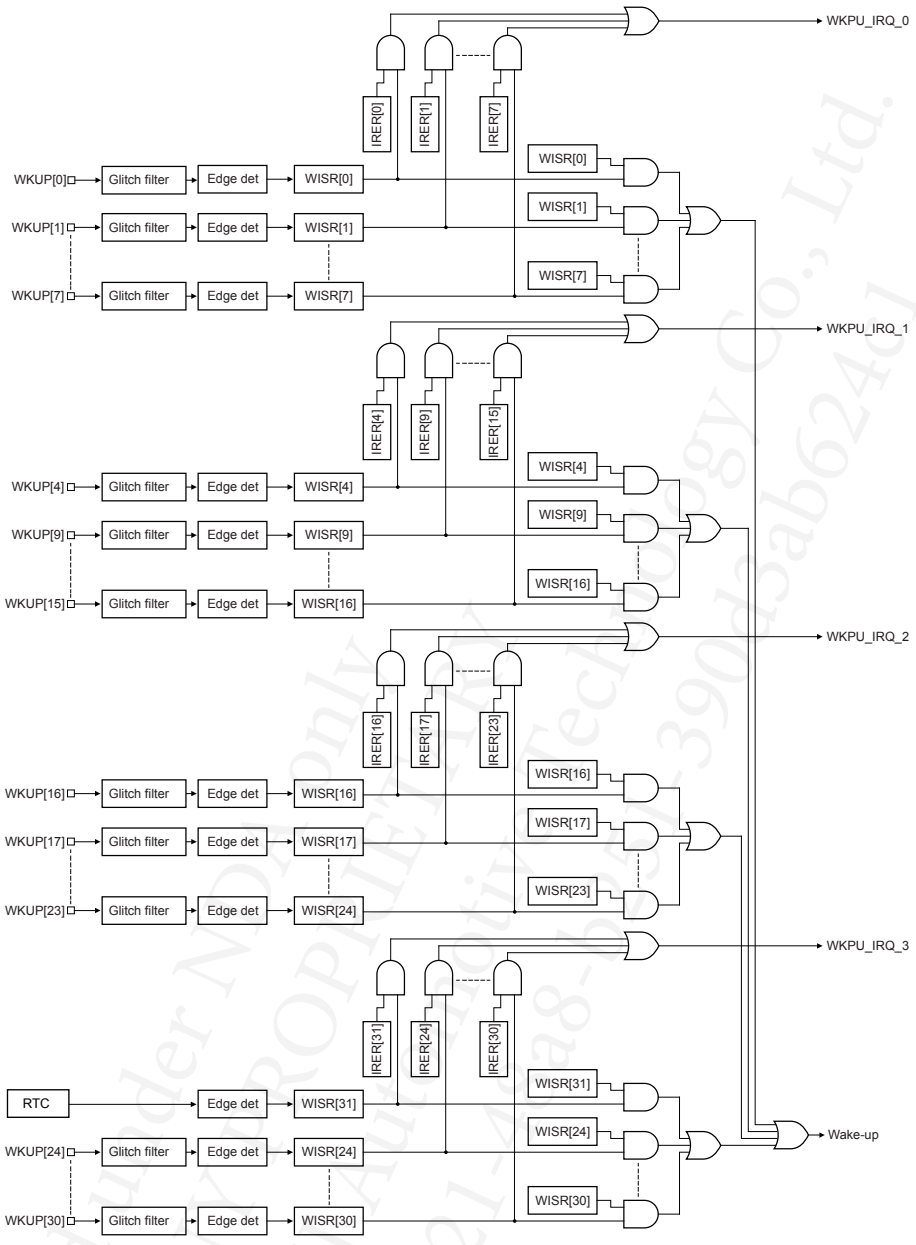


Figure 165. WKPU wake-up and interrupt mapping overview

31.6.5.1 NMI pins

Each NMI pin (NMI[*n*]) maps to NMI *n* in the NMI status and configuration registers in WKPU.

You can configure each NMI input via the corresponding WKPU_NCR[NREE] and WKPU_NCR[NFEE] to trigger an NMI on:

- Any rising edge, but not any falling edge
- Any falling edge, but not any rising edge
- Any edge
- No edge (for example, NMI pin disabled)

The default after a full chip reset is no edge.

You cannot configure NMI inputs to cause a wake-up from Standby mode.

There is a glitch filter between each NMI pad input buffer and its corresponding WKPU input. You can enable and disable the filter via the corresponding WKPU_NCR[NFE] field. The default condition of the filter is disabled after a full chip reset.

See the "Interrupts" section of the WKPU chapter for further details of the processor assignment of each NMI input.

31.6.5.2 Wake-up pins

Each wake-up pin (WKUP[*n*]) maps to wake-up *n* in the WKPU's status and configuration registers.

Each wake-up input is configurable via the corresponding WKPU_WRER, WKPU_WIREER, and WKPU_WIFEER register fields to trigger a wake-up from Standby mode on:

- Any rising edge but not any falling edge
- Any falling edge but not any rising edge
- Any edge
- No edge (for example, wake-up input is disabled)

The default after a full chip reset is no edge.

There is a glitch filter between each wake-up pad input buffer and its corresponding WKPU input. You can enable or disable the filter via the corresponding WKPU_WIFER field. A full chip reset sets the filter to its default condition with the filter disabled.

31.6.5.3 Wake from RTC

RTC wake-up maps to wake-up 31 in the WKPU's wake-up status and configuration registers.

RTC wake-up must be enabled in both WKPU and RTC. You can disable it in either WKPU or RTC.

The RTC wake-up event is triggered on assertion of RTC wake-up output.

31.6.6 Wake-up pin glitch filter

The wake-up input glitch filter filters glitches on the wake-up pins without the need of any clock.

The characteristics of the wake-up pin glitch filter are:

- Maximum pulse width rejected = 20 ns
- Minimum pulse width passed = 400 ns

31.7 Power management and power domains

The power management architecture supports two types of chips:

- Single-core logic power domain
- Two-core power domains (Run mode and Standby mode)

In both cases, there are multiple non-core power domains to support pads and analog modules.

See the IOMUX spreadsheet attached to this reference manual for a list of power domains.

31.7.1 Precision voltage monitoring

The purpose of the precision voltage detectors (LVDs and HVDs) is to ensure that each supply voltage remains within its minimum and maximum operating voltage levels. If any supply voltage level is outside this range, the corresponding precision voltage detector indicates a failure. The consequence of the failure is a power-on reset sequence executed on the chip.

There are no on-chip precision voltage monitors. All required precision voltage monitoring is done off-chip. Each precision voltage monitor should be on the same sense point as the regulator for the corresponding power supply.

There is a dedicated PMIC_SENSE analog output pad for providing the internal cold-point voltage level of the VDD power domain.

The PMIC protects against VDD overvoltage in the case of a broken connection between the VDD sense point and the PMIC (for example, the PMIC_SENSE ball disconnects from the board). The PMIC then indicates the detected failure via POR_B assertion or other error indication.

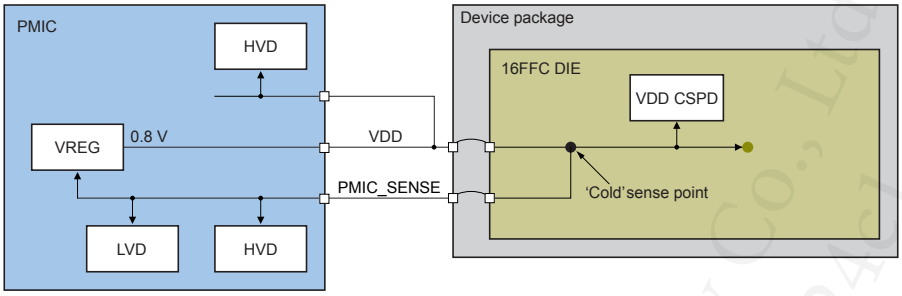


Figure 166. Example of protection against broken PMIC_SENSE connection

NOTE

The additional sense point at the VDD pin on the board is monitored in case PMIC_SENSE connection is broken.

Any extended overvoltage condition detected by PMIC causes PMIC to shut down the corresponding regulator.

Table 184. Minimum off-chip LVDs and HVDs

Supply	Type	Sense
0.8 V	HVD	PMIC_SENSE
	LVD	
1.8 V	HVD	Output of 1.8 V regulator
	LVD	
3.3 V	HVD	Output of 3.3 V regulator
	LVD	
1.1 V to 1.5 V	LVD	Output of 1.1 V to 1.5 V regulator
1.2 V	LVD	Output of 1.2 V regulator

NOTE

There is no need to monitor supplies that are not required for a chip.

When one of the off-chip precision voltage monitors detects an overvoltage or undervoltage condition, the POR_B input asserts.

The VDD_STBY domain may not be monitored by the PMIC when supplied by its low-power regulator. This minimizes overall power consumption during Standby mode.

31.7.2 Supply Presence Detectors (SPD)

SPDs ensure that all the physical connection paths from the voltage regulators to the on-chip voltage supply rails (for example, board connection from PMIC outputs to chip pins, or package connections from pins to the die power supply pads, including substrate and bond wires) are intact.

Depending on the type of SPD (NCSPD or CSPD), if a power rail connection to the supply is broken, the corresponding SPD indicates a failure. The consequences of the failure are:

- An on-chip power-on reset sequence executes for CSPDs.
- If enabled in the PMC NCSPD_CTL register, an on-chip destructive reset sequence executes for NCSPDs.

Each SPD signals a disconnect event based on its detection of an extremely low voltage on a particular power domain. Such an event automatically triggers when the supply of this target power domain is disconnected, or when it falls below a voltage level that is significantly lower than the precision LVD level of the PMIC. When any of the CSPDs/NCSPDs detect a disconnected core supply, the corresponding CSPD/NCSPD input of the PMC asserts. See the "Reset" chapter for further details.

The figure below shows the basic SPD operation.

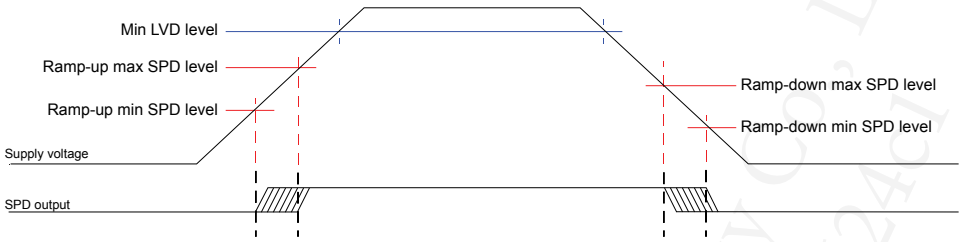


Figure 167. Basic SPD operation

31.7.2.1 NCSPD configuration

The following table shows the NCSPD mapping in the NCSPD_CTL, NCSPD_STAT, and NCSPDEF registers of the PMC.

NOTE

See the chip's data sheet for voltage supply descriptions.

Table 185. PMC NCSPD mapping

PMC NCSPD register fields	NCSPD index	Module	Supply
NCSPDEF[NCSPD_EVENT_CAPTURE0] NCSPD_CTL[NCSPD_CTL0] NCSPD_STAT[NCSPD_STAT0]	NCSPD0	FXOSC	VDD_FXOSC
NCSPDEF[NCSPD_EVENT_CAPTURE1] NCSPD_CTL[NCSPD_CTL1] NCSPD_STAT[NCSPD_STAT1]	NCSPD1	ADC0	VDD_ADC
NCSPDEF[NCSPD_EVENT_CAPTURE2] NCSPD_CTL[NCSPD_CTL2] NCSPD_STAT[NCSPD_STAT2]	NCSPD2	ADC1	VDD_ADC
NCSPDEF[NCSPD_EVENT_CAPTURE3] NCSPD_CTL[NCSPD_CTL3] NCSPD_STAT[NCSPD_STAT3]	NCSPD3	TMU	VDD_TMU
NCSPDEF[NCSPD_EVENT_CAPTURE4] NCSPD_CTL[NCSPD_CTL4] NCSPD_STAT[NCSPD_STAT4]	NCSPD4	FBXC	VDD_EFUSE

Table continues on the next page...

Table 185. PMC NCSPD mapping (continued)

PMC NCSPD register fields	NCSPD index	Module	Supply
NCSPDEF[NCSPD_EVENT_CAPTURE5] NCSPD_CTL[NCSPD_CTL5] NCSPD_STAT[NCSPD_STAT5]	NCSPD5	ACCEL_PLL, PERIPH_PLL and CORE_PLL	VDD_HV_PLL
—	NCSPD6	—	Not mapped to any monitor for this chip. Reserved for future expansion.
NCSPDEF[NCSPD_EVENT_CAPTURE7] NCSPD_CTL[NCSPD_CTL7] NCSPD_STAT[NCSPD_STAT7]	NCSPD7	DDR_PLL	VDD_HV_PLL_DDR0
NCSPDEF[NCSPD_EVENT_CAPTURE8] NCSPD_CTL[NCSPD_CTL8] NCSPD_STAT[NCSPD_STAT8]	NCSPD8	DDR_PLL	VDD_HV_PLL_DDR0
NCSPDEF[NCSPD_EVENT_CAPTURE9] NCSPD_CTL[NCSPD_CTL9] NCSPD_STAT[NCSPD_STAT9]	NCSPD9	AURORA_PLL	VDD_HV_PLL_AUR
NCSPDEF[NCSPD_EVENT_CAPTURE10] NCSPD_CTL[NCSPD_CTL10] NCSPD_STAT[NCSPD_STAT10]	NCSPD10	AURORA_PLL	VDD_LV_PLL_AUR[1]
NCSPDEF[NCSPD_EVENT_CAPTURE11] NCSPD_CTL[NCSPD_CTL11] NCSPD_STAT[NCSPD_STAT11]	NCSPD11	Standby domain	VDD_IO_STBY[2]
—	NCSPD12 – NCSPD16	—	Not mapped to any monitor for this chip. Reserved for future expansion.
NCSPDEF[NCSPD_EVENT_CAPTURE17] NCSPD_CTL[NCSPD_CTL17] NCSPD_STAT[NCSPD_STAT17]	NCSPD17	GPIO	VDD_IO_A
NCSPDEF[NCSPD_EVENT_CAPTURE18] NCSPD_CTL[NCSPD_CTL18] NCSPD_STAT[NCSPD_STAT18]	NCSPD18	GPIO	VDD_IO_B
—	NCSPD19 – NCSPD20	—	Not mapped to any monitor for this chip. Reserved for future expansion.

Table continues on the next page...

Table 185. PMC NCSPD mapping (continued)

PMC NCSPD register fields	NCSPD index	Module	Supply
NCSPDEF[NCSPD_EVENT_CAPTURE21] NCSPD_CTL[NCSPD_CTL21] NCSPD_STAT[NCSPD_STAT21]	NCSPD21	USB	VDD_IO_USB
NCSPDEF[NCSPD_EVENT_CAPTURE22] NCSPD_CTL[NCSPD_CTL22] NCSPD_STAT[NCSPD_STAT22]	NCSPD22	uSDHC	VDD_IO_SDHC
NCSPDEF[NCSPD_EVENT_CAPTURE23] NCSPD_CTL[NCSPD_CTL23] NCSPD_STAT[NCSPD_STAT23]	NCSPD23	QuadSPI	VDD_IO_QSPI
—	NCSPD24 – NCSPD27	—	Not mapped to any monitor for this chip. Reserved for future expansion.
—	NCSPD28	—	VDD_IO_CLKOUT
—	NCSPD29 – NCSPD31	—	Not mapped to any monitor for this chip. Reserved for future expansion.

1. NCSPD on the corresponding HV supply may also trigger when the LV supply drops significantly.
2. When VDD_IO_STBY drops below the NCSPD threshold, it may generate a POR. In this case, the hardware writes 1 to PMC_SSR[POR_SUP].

NOTE

- If the core supply is not active, incorrect results may occur in these CSPD/NCSPD registers:
  - CSPD_EVENT_CAPTURE
  - NCSPD_STAT
  - SSR
- NCSPD11 monitors the 3.3 V supply (VDD_IO_STBY) of the POR_B pad segment to ensure that the chip can enter the destructive reset sequence even if the connection to VDD_IO_STBY is broken and the POR_B signal cannot be properly asserted. For this, the software must enable NCSPD11 in PMC.

  In some cases, the connection to VDD_IO_STBY may be broken but the POR_B signal is still asserted enough to be interpreted as a power-on reset. This results in the PMC_SSR[POR_SUP] written with 1, but the NCSPD11 event capture flag not being set, even if NCSPD11 is enabled.

31.7.2.2 CSPD configuration

The following table shows the field mapping in the PMC_CSPDEF register. See the chip *Data Sheet* for descriptions of all supplies.

Table 186. PMC CSPD mapping

PMC_CSPDEF field	CSPD index	Module	Supply
HVCPOREF	HVCPOREF	PMC	VDD_VREF

Table continues on the next page...

Table 186. PMC CSPD mapping (continued)

PMC_CSPDEF field	CSPD index	Module	Supply
CSPD_EVENT_CAPTURE1	CSPD1	FIRC	VDD_FIRC
CSPD_EVENT_CAPTURE2	CSPD2	FIRC	VDD
3–31	CSPD3 – CSPD31	—	Unused

NOTE

If the core supply is not active, incorrect results may occur in these CSPD/NCSPD registers:

- CSPD_EVENT_CAPTURE
- NCSPD_STAT
- SSR

31.7.3 Voltage overshoot and undershoot protection

PCFS features are provided to ensure smooth clock frequency transitions:

- Ramp-up to protect against an LVD event occurring due to a sudden increase in current consumption when turning on a portion of the chip during Run mode.
- Ramp-down to protect against an HVD event occurring due to a sudden decrease in current consumption due to a functional reset.

There is no on-chip protection against an HVD event occurring due to a sudden decrease in the current consumption of the chip due to a destructive reset.

The off-chip power supply can react quickly enough to a sudden decrease in chip current consumption in the case of a destructive reset.

31.7.4 External PMIC interface

The external PMIC contains safety-critical functionalities:

- Voltage supervision
- Watchdog operation (POR_WDOG)
- Error monitoring
- Analog BIST to check safety functions,

If any failure occurs, the external PMIC asserts POR_B.

31.7.4.1 Use case: Interfacing to external PMIC

For interfacing to the PMIC, see the chip Hardware Design Guidelines.

31.7.4.2 Debug use case: Interfacing to external PMIC

In the debug use case, RESET_B is bidirectional.

The external debugger must clear the watchdog in the external PMIC after RESET_B asserts. Otherwise, the PMIC can assert POR_B after a defined number of RESET_B assertions. Figure 168 shows the debug use case.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4265-4821-48a8-b231-3903ab624c1

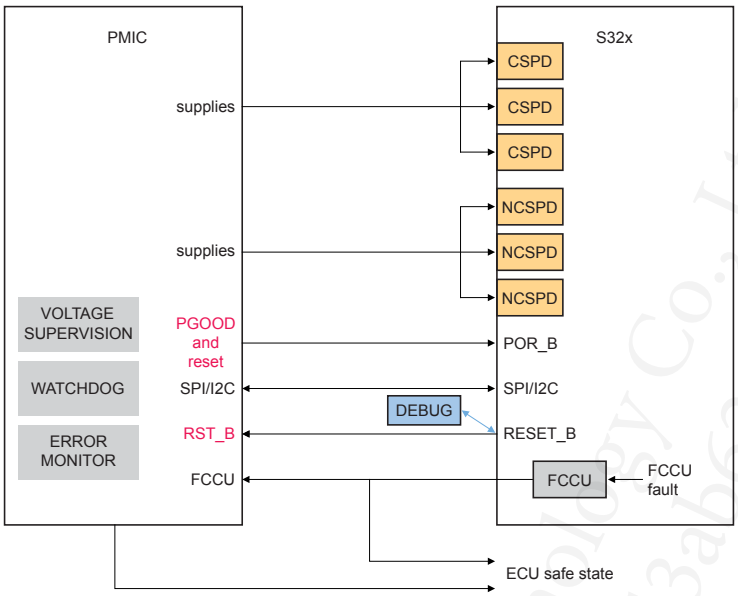


Figure 168. Debug use case: Interfacing to external PMIC

31.7.4.3 Standby mode use case: Interfacing to external PMIC

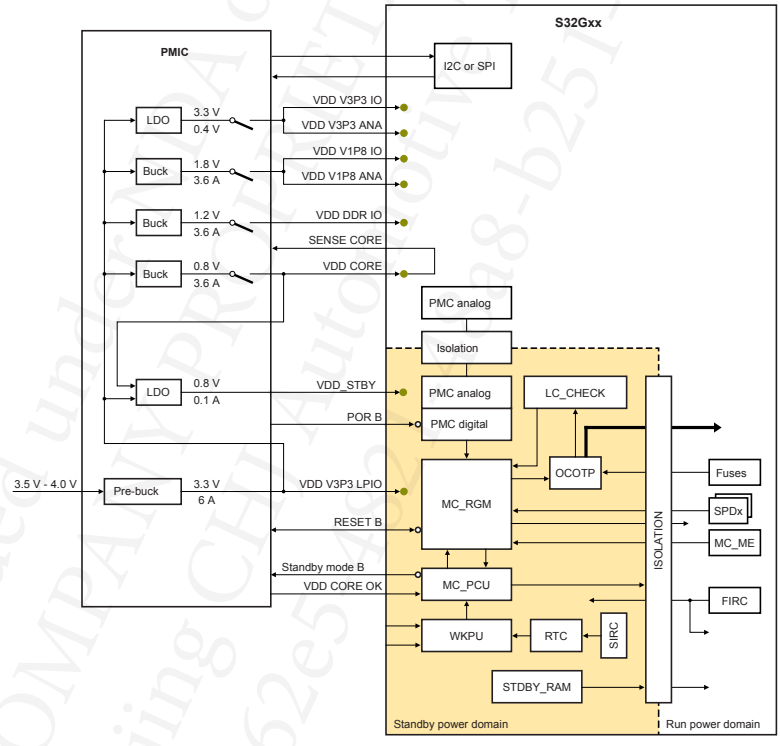


Figure 169. Standby mode use case interfacing external PMIC

31.7.4.4 Safety context for PMIC WDOG

The PMIC WDOG must be available at the system level immediately when safety outputs release. Before that, RSTB (reset output from PMIC) or POR_B (power good output from PMIC) are high so the chip can start its own initialization and WDOG has a timeout of 256 ms for the first time if the chip is stuck.

Sequence of steps:

1. PMIC powers up.
2. The power-up sequence configuration in the PMIC determines chip and peripheral supplies.
3. After power supplies reach correct normal operation, the PMIC POR_B and RSTB pins are automatically deasserted. Upon this deassertion, the PMIC starts an internal counter with a period of 256 ms. During this 256 ms window, the chip starts execution of its own initialization procedure. You must configure the PMIC before the lapse of the window resets the first watchdog.
4. At this point WDOG is working in normal mode. The watchdog refreshes every *x* ms based on the watchdog window period configured during the initialization phase (for example, 10 ms).
5. The chip can now request the PMIC to release the safety output. The chip is now able to start the application.

When the chip is unable to program the PMIC (for example, it is stuck in reset), PMIC detects this case even with an unconfigured watchdog timer. If the chip is pulling reset low continuously, and the PMIC has an internal counter (8 s), the PMIC automatically transitions to Deep Fail-Safe mode.

In Deep Fail-Safe mode:

1. All regulators in the PMIC switch off.
2. The chip enters Reset mode.
3. The PMIC FS0B (fail-safe output 0) and POR_B pins are asserted (driven low).
4. If configured on the chip, the automatic restart function executes.

The automatic restart option, if configured at the factory via fuses, operates according to one of the following options:

- No automatic restart
- Automatic restart (executes a maximum of 15 times)
- Automatic restart (no limit to the number of times it can execute)

31.8 Power mode control blocks

31.8.1 PMIC control

Software controls the PMIC during Run mode via the I^2C communication interface.

Additionally, PMIC control during Standby mode is via a signal from the chip indicating whether VDD can be turned off, or must be turned on. The PMIC uses a hardware signal to indicate to the chip whether the VDD voltage is within its operating range, separate from the POR_B input.

31.8.1.1 Static Voltage Scaling (SVS)

SVS is a method in which the voltage of a power domain can be adjusted up or down to optimize power consumption at operational frequencies used in that domain. The decision of whether the voltage should be adjusted, and by how much, is based on process data for the particular chip.

Beijing CU Automotive Technology Co., Ltd.
COMPANY PROPRIETARY
Limited under NDA only
c8c46225-4241-48a8-b251-390a0ab624f1

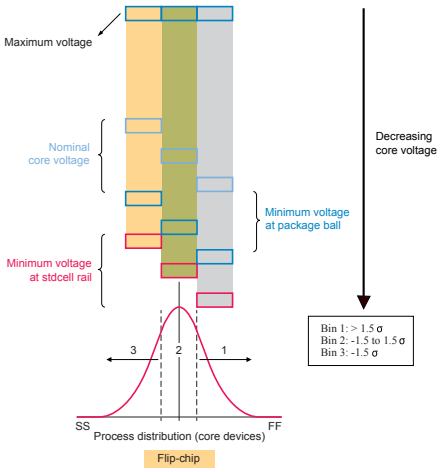


Figure 170. SVS example

NOTE

Voltage levels shown in Figure 170 are examples. See the chip data sheet for actual operating levels.

The process data for each die is stored in the DIE_PROCESS[1:0] fuses according to the following coding:

- DIE_PROCESS[1:0] = 00b, SVS is disabled
- DIE_PROCESS[1:0] = 01b, SVS mode 1
- DIE_PROCESS[1:0] = 10b, SVS mode 2
- DIE_PROCESS[1:0] = 11b, SVS mode 3

For more information on power domain SVS voltage levels, see the “Operating Conditions" section of the chip data sheet.

For more information on power domain SVS voltage levels, see the “DC Electricals" section in the chip data sheet.

An overview of the SVS execution flow is shown in the following diagram.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-9251-3903ab624c1

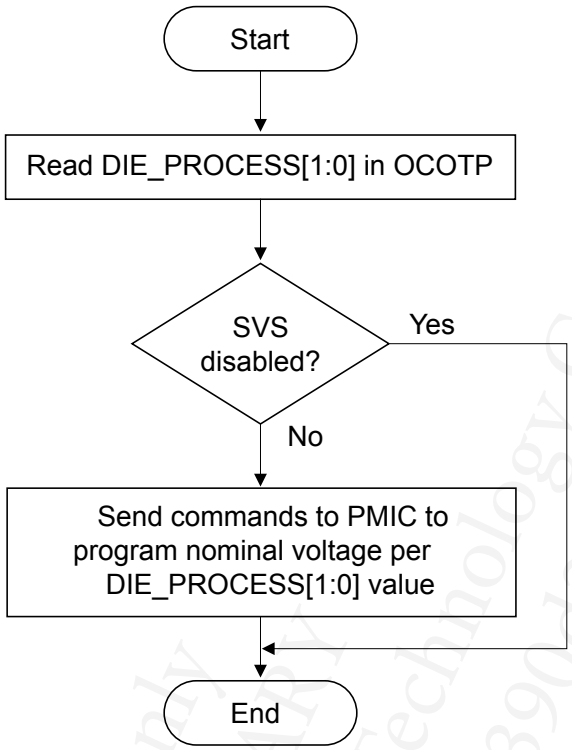


Figure 171. SVS execution steps

The above SVS execution steps are performed by the application software.

Note that PMIC voltage monitoring of the core supply may be disabled while SVS voltage trimming is occurring, which would impact the functional safety viability during this time. Therefore, we recommend that you apply SVS prior to starting the safety application.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b51-390d3ab624c1

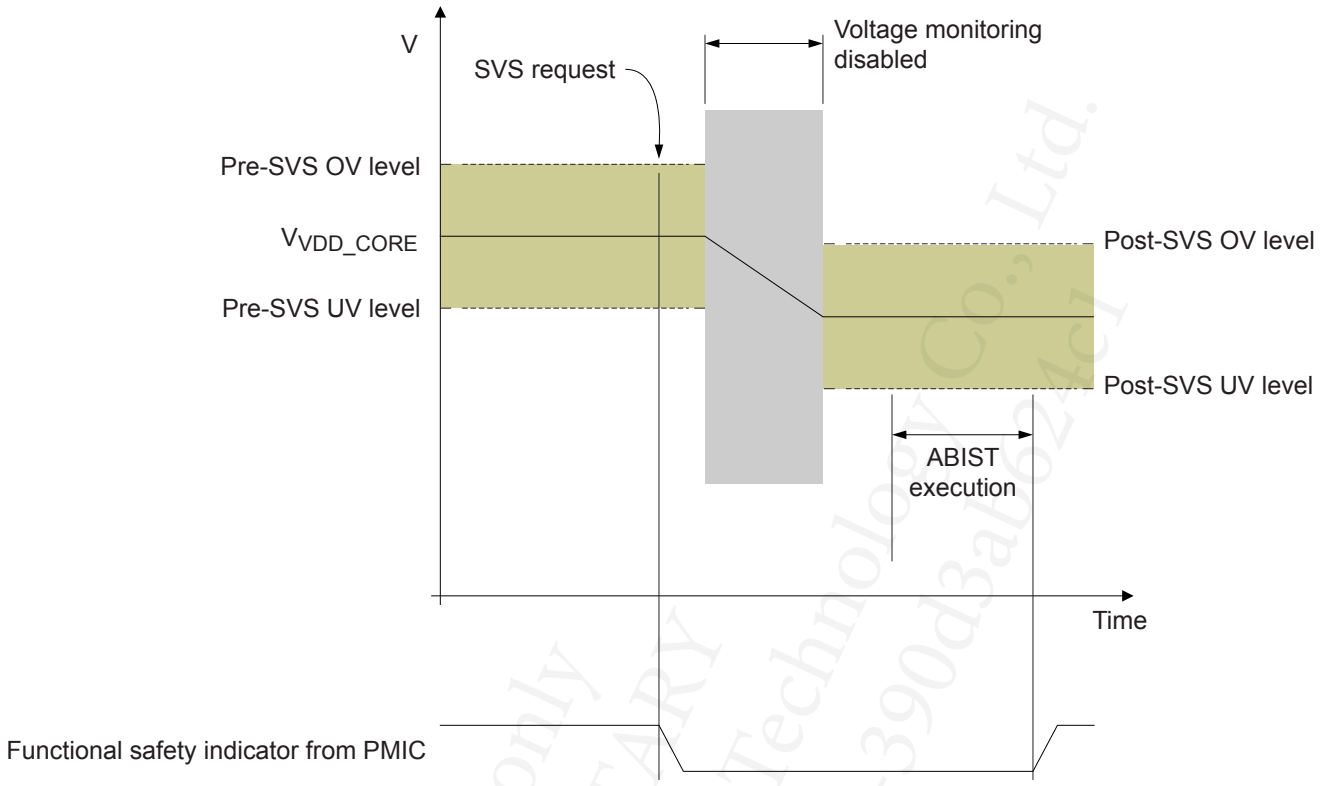


Figure 172. Possible voltage monitor disabling during SVS

Disabling of voltage monitors by the PMIC during SVS is indicated by the PMIC to the chip (for example, via the FSOB output on NXP Semiconductor PMICs).

VDD_STBY is typically trimmed per the SVS setting during Run mode, but at the nominal voltage during Standby mode. In addition, core supply and VDD_STBY may still be at nominal voltage after Standby mode exit and require software to execute SVS again.

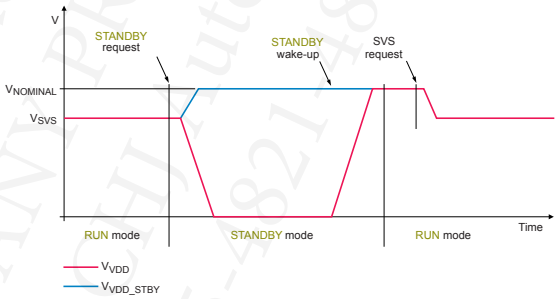


Figure 173. Core supply and VDD_STBY voltages before, during, and after Standby mode

31.8.2 PMC

The analog portion of PMC:

- Ensures that if any of the signals that can assert the raw POR signal in the chip are asserted, the raw POR signal is asserted for a minimum of 1 µs.
- Provides a band gap voltage.

- Contains CSPDs to monitor the following power rails:
  - Core supply - supplies power to the POR signal generation logic.
  - PMC supply - supplies the band gap.

PMC also contains a watchdog to recover from undesired stuck-in-reset conditions.

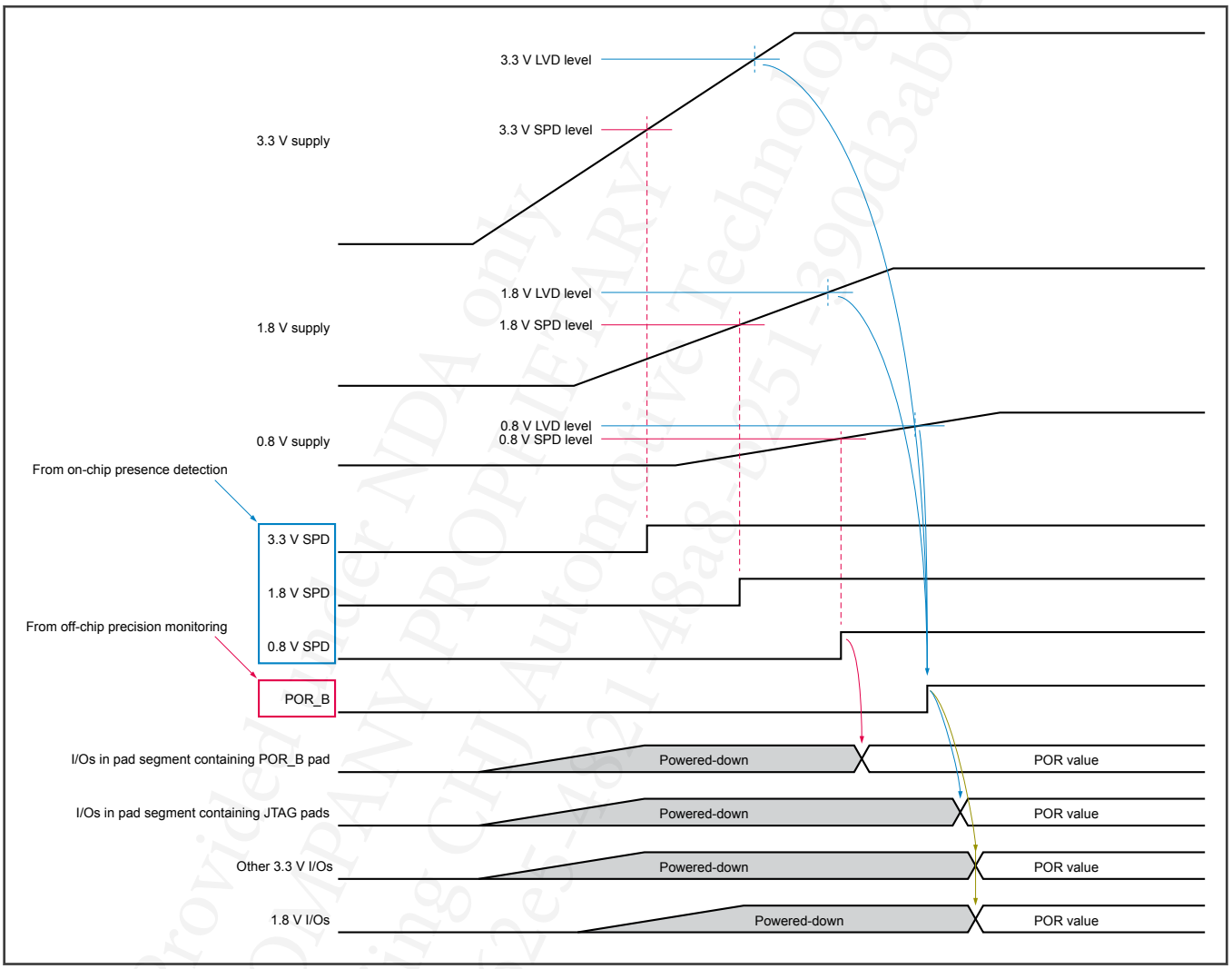
The digital portion of the PMC contains logic for generating a POR used to reset MC_RGM and the POR signal.

The digital portion of the PMC contains logic and registers required to manage the CSPD and NCSPD signals.

31.9 Powerup/powerdown ramp rates and sequences

See the chip's data sheet for supply ramp rate and sequence specifications.

31.9.1 Pin behavior during powerup



31.9.2 Chip startup and shutdown

When the core supply is turned on during startup, VDD_STBY is also turned on, and the other way around. However, the actual turn-on times and/or ramp rates may differ.

When the core supply is turned off during shutdown, VDD_STBY is also turned off, and the other way around. However, the actual turn-off times and/or ramp rates may differ.

When VDD_IO_A and VDD_IO_B are turned on during startup, VDD_IO_STBY is also turned on, and the other way around. However, the actual turn-on times and/or ramp rates may differ.

When VDD_IO_A and VDD_IO_B are turned off during shutdown, VDD_IO_STBY is also turned off, and the other way around. However, the actual turn-off times and/or ramp rates may differ.

31.9.3 Standby mode entry

Switchable power supplies are not turned off during Standby mode entry until the PMIC_STBY_MODE_B output pin asserts.

Turning off switchable power supplies during Standby mode entry does not cause the supplies still on to have voltage over/undershoots.

Switchable power supplies may be turned off together with no requirement for a specific shutdown sequence.

31.9.4 Standby mode exit

For more information on Standby mode exit sequence, refer to the chip data sheet.

31.10 Low power modes

31.10.1 Wake-up sources

The device supports 23 wake-up pads in Standby mode (WKUP0 – WKUP22). These pads are identified in the IOMUX spreadsheet as being in the VDD_IO_STBY power domain.

To determine the mapping of WKUP*n* pins with I/O Power Domain, see the IO Signal tab in the attached IOMUX spreadsheet.

To reduce standby current, wake-up pins are input-only.

An internal interrupt, API wake-up signal from RTC, is also supported.

A typical use case for the API is to trigger a periodic wake-up for the S32G to feed an external watchdog before returning to Standby mode. Typically the WDOG is in the PMIC.

The power profile for this sequence should be kept to a minimum (for example, the power consumption during the wake-up transition, the feed watchdog code and the Standby mode entry transition should be kept as low as possible).

The watchdog timeout can be as low as 1 s, requiring one API wake-up event per second.

The data sheet provides a typical and maximum spec for the transition time into and out of Standby mode. The entry time is the time from software triggering a transition until Standby mode entry. Exit time is the time from a wake-up event until customer software execution.

31.11 Power management and power domains

31.11.1 I/O power domains

Each general-purpose I/O segment has an independent package supply pin, named VDD_IO_x (see the Supply Pins tab in the attached IOMUX spreadsheet).

The I/O segments are shown in following table.

Table 187. I/O power segments

I/O segment	Voltage	Description
VDD_IO_GMAC0	1.8 V / 3.3 V	Ethernet
VDD_IO_GMAC1	1.8 V / 3.3 V	Ethernet
VDD_IO_SDHC	1.8 V / 3.3 V	SDHC smart card

Table continues on the next page...

Table 187. I/O power segments (continued)

I/O segment	Voltage	Description
VDD_IO_QSPI	1.8 V	Quad SPI
VDD_IO_USB	1.8 V / 3.3 V	USB
VDD_IO_B	3.3 V	General purpose 3.3 V
VDD_IO_STBY	3.3 V	Standby 3.3 V
VDD_IO_A	3.3 V	General purpose 3.3 V
VDD_IO_CLKOUT	1.8 V	1.8 V Clkout (GPIO + LVDS)
VDD_IO_DDR0	1.1 V / 1.2 V / 1.35 V	DDR I/O
VDD_IO_PCIE0	1.8 V	PCIe I/O
VDD_IO_PCIE1	1.8 V	PCIe I/O
VDD_IO_AUR	1.8 V	1.8 V Aurora I/O
VDD_ADC	1.8 V	ADC 1.8 V supply

31.11.2 Standby pads

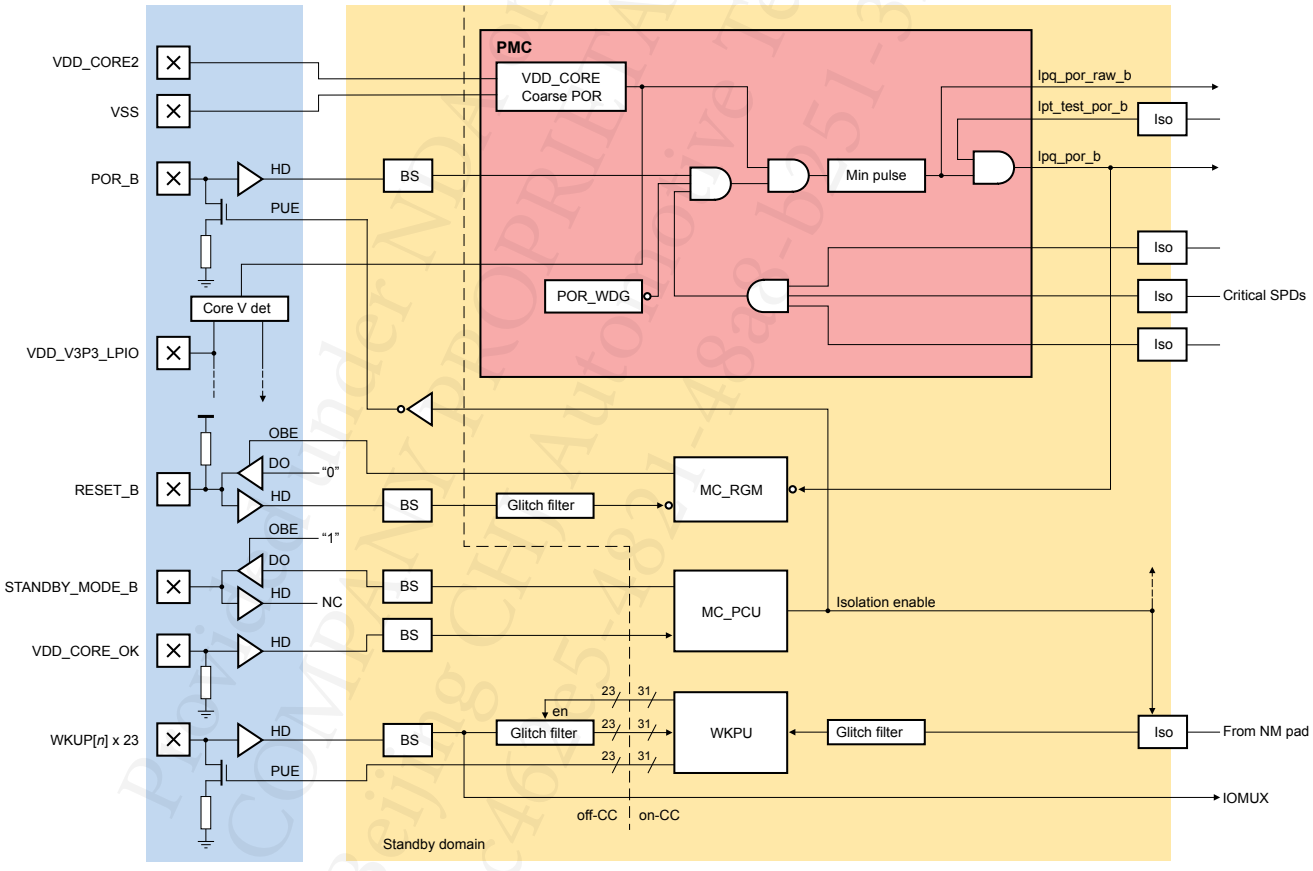


Figure 174. Standby pads overview

Table 188. Standby pad types

Pad name	Pad type	Pull type	Comment
POR_B	3.3 V GPI	Down	Pulldown is disabled when isolation is enabled by MC_PCU; otherwise it is enabled.
RESET_B	3.3 V GPIO	Up	Pullup is always enabled; DO pin is tied to 0.
STANDBY_MODE_B	3.3 V GPIO	None	OBE pin is tied to 1; IND pin remains unconnected.
VDD_CORE_OK	3.3 V GPI	Down	Pulldown is always enabled.
WKUP[*]	3.3 V GPI	Up/Down	PUE and PUS pins are controlled via SIUL and WKPU.

31.12 Glossary

CSPD Critical Supply Presence Detector. See Supply Presence Detectors (SPD) for details.

HVD High Voltage Detector

LVD Low Voltage Detector

NCSPD Noncritical Supply Presence Detector. See Supply Presence Detectors (SPD) for details.

PCFS Progressive Clock Frequency Switching. See the "Clock Generation Module (MC_CGM)" chapter for details.

PMIC_SENSE VDD sense point output to PMIC for low and high-voltage detection

SBC System Basis Chip (for example, PMIC)

VDD_IO_DDR0 DDR I/O power supply

VDD_STBY 0.8 V standby core logic power supply

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.
c8c462e5-4821-48a8-b2...-390d3a...

Chapter 32
Mode Entry Module (MC_ME)

32.1 Chip-specific MC_ME information

32.1.1 MC_ME modes

This chip implements these modes:

- Reset
- Run
- Standby

The chip always enters Run mode on exit from Reset mode.

Run mode is the only mode during which the chip can operate its computational and communications functions.

The chip remains fully powered. The boot core is the only core enabled on Run entry.

Software can enable and disable cores and reset partitions as needed by the application, based on functional and power requirements.

Software can configure the pins as needed by the application. Software can set up and trigger BIST sequence execution off-line as needed by the application:

- Pins are safe-stated.
- No computational or communication activities are possible.

Reset mode is automatically entered after BIST has completed on-line on complete chip or reset partition 0. Some computational and communication activities can run while a BIST sequence runs on one or more reset partitions.

32.1.2 MC_ME partition mapping

32.1.2.1 MC_ME partition 0 mapping

The following table shows partition 0 and its register mapping.

Table 189. MC_ME partition 0 mapping

Register	Register field	Description	Used for
MC_ME_PRTN0_PCONF	PCE	Partition clock enable	Read only (for more information on this field, see the PRTN0_PCONF[PCE] description of MC_ME.)
MC_ME_PRTN0_STAT	PCS	Partition clock status	Read only
MC_ME_PRTN0_COFB0_STAT	BLOCK0	IP block status	uSDHC
	BLOCK1	IP block status	DDR_0
	BLOCK2–4	IP block status	Reserved
	BLOCK5–31	IP block status	Reserved (Off-CC)
MC_ME_PRTN0_COFB0_CLKEN	REQ0	Clock enable	uSDHC
	REQ1	Clock enable	DDR_0

Table continues on the next page...

Table 189. MC_ME partition 0 mapping (continued)

Register	Register field	Description	Used for
	REQ2–4	Clock enable	Reserved
	REQ5–31	Clock enable	Reserved (Off-CC)
MC_ME_PRTN0_CORE0_PCONF	CCE	Core 0 clock enable	CM7_0
MC_ME_PRTN0_CORE0_PUPD	CCUPD	Core 0 clock update	
MC_ME_PRTN0_CORE0_STAT	CCS	Core 0 clock process status	
	WFI	Core 0 wait for interrupt status	
MC_ME_PRTN0_CORE0_ADDR	ADDR	Core 0 address	
MC_ME_PRTN0_CORE1_PCONF	CCE	Core 1 clock enable	CM7_1
MC_ME_PRTN0_CORE1_PUPD	CCUPD	Core 1 clock update	
MC_ME_PRTN0_CORE1_STAT	CCS	Core 1 clock process status	
	WFI	Core 1 wait for interrupt status	
MC_ME_PRTN0_CORE1_ADDR	ADDR	Core 1 address	
MC_ME_PRTN0_CORE2_PCONF	CCE	Core 2 clock enable	CM7_2
MC_ME_PRTN0_CORE2_PUPD	CCUPD	Core 2 clock update	
MC_ME_PRTN0_CORE2_STAT	CCS	Core 2 clock process status	
	WFI	Core 2 wait for interrupt status	
MC_ME_PRTN0_CORE2_ADDR	ADDR	Core 2 address	
MC_ME_PRTN0_CORE3_PCONF	CCE	Core 3 clock enable	Reserved
MC_ME_PRTN0_CORE3_PUPD	CCUPD	Core 3 clock update	Reserved
MC_ME_PRTN0_CORE3_STAT	CCS	Core 3 clock process status	Reserved
	WFI	Core 3 wait for interrupt status	HSE-CM7
MC_ME_PRTN0_CORE3_ADDR	ADDR	Core 3 address	Reserved

32.1.2.2 MC_ME partition 1 mapping

The following table below shows partition 1 and its register mapping.

Table 190. MC_ME partition 1 mapping

Register	Register field	Description	Used for
MC_ME_PRTN1_PCONF	PCE	Partition clock enable	Reset domain 1
MC_ME_PRTN1_PCONF	OSSE	Output safe stating enable	Reset domain 1
MC_ME_PRTN1_PUPD	PCUD	Partition clock update	Reset domain 1
MC_ME_PRTN1_PUPD	OSSUD	Output safe stating update	Reset domain 1
MC_ME_PRTN1_STAT	PCS	Partition clock status	Reset domain 1
MC_ME_PRTN1_STAT	OSSS	Output safe stating status	Reset domain 1

Table continues on the next page...

Table 190. MC_ME partition 1 mapping (continued)

Register	Register field	Description	Used for
MC_ME_PRTN1_CORE0_PCONF	CCE	Core 0 clock enable	In Lockstep mode, these registers control CA53 cluster 0/1 (shared control for Cortex-A53 core 0/1/2/3) In Performance mode, these registers control CA53 cluster0 (shared control for Cortex-A53 core0/1)
MC_ME_PRTN1_CORE0_PUPD	CCUPD	Core 0 clock update	
MC_ME_PRTN1_CORE0_STAT	CCS	Core 0 clock process status	CA53 cluster0 core0/1
	WFI	Core 0 wait for interrupt status	CA53 cluster0 core0
MC_ME_PRTN1_CORE0_ADDR	ADDR	Core 0 address	Performance mode - CA53 cluster0 core0 Lockstep mode - shared control for core0 of CA53 cluster0 and cluster1
MC_ME_PRTN1_CORE1_PCONF	CCE	Core 1 clock enable	Reserved
MC_ME_PRTN1_CORE1_PUPD	CCUPD	Core 1 clock update	Reserved
MC_ME_PRTN1_CORE1_STAT	CCS	Core 1 clock process status	Reserved
	WFI	Core 1 wait for interrupt status	CA53 cluster0 core1
MC_ME_PRTN1_CORE1_ADDR	ADDR	Core 1 address	Performance mode - CA53 cluster0 core1 Lockstep mode - shared control for core1 of CA53 cluster0 and cluster1
MC_ME_PRTN1_CORE2_PCONF	CCE	Core 2 clock enable	In Lockstep mode, these registers are reserved In the Performance mode, these registers control CA53 cluster1 (shared control for Cortex-A53 core2/3)
MC_ME_PRTN1_CORE2_PUPD	CCUPD	Core 2 clock update	
MC_ME_PRTN1_CORE2_STAT	CCS	Core 2 clock process status	CA53 cluster1 core0/1
	WFI	Core 2 wait for interrupt status	CA53 cluster1 core0

Table continues on the next page...

Table 190. MC_ME partition 1 mapping (continued)

Register	Register field	Description	Used for
MC_ME_PRTN1_CORE2_ADDR	ADDR	Core 2 address	Performance mode - CA53 cluster1 core0 Lockstep mode - Reserved
MC_ME_PRTN1_CORE3_PCONF	CCE	Core 3 clock enable	Reserved
MC_ME_PRTN1_CORE3_PUPD	CCUPD	Core 3 clock update	Reserved
MC_ME_PRTN1_CORE3_STAT	CCS	Core 3 clock process status	Reserved
	WFI	Core 3 wait for interrupt status	CA53 cluster1 core1
MC_ME_PRTN1_CORE3_ADDR	ADDR	Core 3 address	Performance mode - CA53 cluster1 core1 Lockstep mode - Reserved

32.1.2.3 MC_ME partition 2 mapping

The following table shows partition 2 and its register mapping.

Table 191. MC_ME partition 2 mapping

Register	Register field	Description	Used for
MC_ME_PRTN2_PCONF	PCE	Partition clock enable	Reset domain 2
MC_ME_PRTN2_PCONF	OSSE	Output safe stating enable	Reset domain 2
MC_ME_PRTN2_PUPD	PCUD	Partition clock update	Reset domain 2
MC_ME_PRTN2_PUPD	OSSUD	Output safe stating update	Reset domain 2
MC_ME_PRTN2_STAT	PCS	Partition clock status	Reset domain 2
MC_ME_PRTN2_STAT	OSSS	Output safe stating status	Reset domain 2
MC_ME_PRTN2_COFB0_STAT	BLOCK0–31	IP block status	Reserved
MC_ME_PRTN2_COFB0_CLKEN	REQ0	Clock enable	PFE-MAC0
	REQ1	Clock enable	PFE-MAC1
	REQ2	Clock enable	PFE-MAC2
	REQ3	Clock enable	PFE-TS_CLK
	REQ4–31	Clock enable	Reserved

32.1.2.4 MC_ME partition 3 mapping

The following table shows partition 3 and its register mapping.

Table 192. MC_ME partition 3 mapping

Register	Register field	Description	Used for
MC_ME_PRTN3_PCONF	PCE	Partition clock enable	Reset domain 3
MC_ME_PRTN3_PCONF	OSSE	Output safe stating enable	Reset domain 3
MC_ME_PRTN3_PUPD	PCUD	Partition clock update	Reset domain 3
MC_ME_PRTN3_PUPD	OSSUD	Output safe stating update	Reset domain 3
MC_ME_PRTN3_STAT	PCS	Partition clock status	Reset domain 3
MC_ME_PRTN3_STAT	OSSS	Output safe stating status	Reset domain 3
MC_ME_PRTN3_COFB0_STAT	BLOCK0–31	IP block status	Reserved
MC_ME_PRTN3_COFB0_CLKEN	REQ0–31	Clock enable	Reserved

32.1.3 Core controls

The MC_ME provides the core boot addresses via the MC_ME_PRTNm_COREn_ADDR registers.

For CA53_x, ADDR[39:32] bits come from A53 GPR registers, and ADDR[31:0] bits come from the MC_ME_PRTNm_COREn_ADDR register.

Each core determines its boot address one of two possible ways:

- After core reset deassertion, the core jumps to the location specified in MC_ME_PRTNm_COREn_ADDR and starts executing from there.
- The core obtains the boot address specified in the corresponding MC_ME_PRTNm_COREn_ADDR register via signals from the MC_ME and then boots from that address on reset exit.

32.1.4 Turning off cores and partitions

There is no hardware handshake to ensure a module (core, peripheral, or subsystem) has been shut down before disabling its clock. Software must disable the clock, and when software submits a request to disable a module's clock, the clock is disabled regardless of the module state.

NOTE

For details about turning cores and partitions on or off, see these sections of the Reset chapter:

- Software reset partition turn-on flow chart
- Software reset partition turn-off flowchart
- Core turn-on sequence
- Core turn-off sequence

The subsections below provide the details of procedures related to MC_ME that are referred in the above mentioned sub-sections of the Reset chapter of this document.

32.1.4.1 Enable output isolation

1. Indicate that you want to enable output safe stating for the partition (MC_ME_PRTNm_PCONF[OSSE]).
2. Enable the partition clock hardware-update process (MC_ME_PRTNm_COREn_PUPD[CCUPD]).
3. Start the enabled hardware-update process (MC_ME_CTL_KEY[KEY]).

   MC_ME requests the partition's output isolation cells to switch to isolation mode.
4. Make sure the hardware process has completed (MC_ME_PRTNm_PUPD[PCUD]).

5. Make sure the partition's output isolation cells are transitioned to isolation mode (MC_ME_PRTNm_STAT[OSSS]).

32.1.4.2 Disable core clocks

1. Indicate that you want to disable the core clock (MC_ME_PRTNm_COREn_PCONF[CCE]).
2. Enable the clock hardware-update processes (MC_ME_PRTNm_COREn_PUPD[CCUPD]).
3. Start the enabled hardware-update processes (MC_ME_CTL_KEY[KEY]).

   MC_ME requests closing of the clock gates for the specified core.
4. Make sure the clock hardware process has completed (MC_ME_PRTNm_PUPD[PCUD]).
5. Make sure the core clock gates are closed (MC_ME_PRTNm_COREn_STAT[CCS]).

32.1.4.3 Disable partition clocks

The MC_ME manages the disabling of a partition's clocks by the following steps:

1. Software writes 0 to a partition's MC_ME_PRTNm_PCONF[PCE] and then executes a partition clock hardware-update-process sequence by:
   - Setting MC_ME_PRTNm_PUPD to 1
   - Writing the valid KEY combinations to the MC_ME_CTL_KEY register
2. The MC_ME requests the clock gating logic to close the clock gates corresponding to that partition.
3. When the hardware process is completed, MC_ME_PRTNm_PUPD[PCUD] is reset.
4. When the clock gating logic completes closing the partition's clock gates, the partition's MC_ME_PRTNm_STAT[PCS] is cleared to 0.

32.1.4.4 Disable COFB clocks

To disable COFB clocks using MC_ME, follow the steps mentioned below:

1. Indicate that you want to disable the COFB clock using MC_ME_PRTNm_COFBn_CLKEN[REQ] fields followed by writing 1 into the partition's MC_ME_PRTNm_PCONF[PCE] register field
2. Initiate the clock hardware process by writing “1” to MC_ME_PRNTm_PUPD[PCUD]
3. Write the valid key combinations into the MC_ME_CTRL_KEY register
4. Make sure the COFB clock is gated by reading MC_ME_PRTNm_COFBn_STAT[BLOCK]

32.1.4.5 Assert partition reset signals

1. Request assertion of the partition's reset signals (MC_RGM_PRSTn_0).
2. Make sure the partition reset signals are asserted (MC_RGM_PSTATn_0).

32.2 Introduction

The MC_ME module generates control signals for a set of modules of the SoC. The set of signals are defined in corresponding 'Partition Configuration Registers'. It also implements a software-based mechanism for initiating a functional and destructive reset sequence and standby entry handshake with power management of SoC. See Figure 175 for the MC_ME block diagram.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEIA Automotive Technology Co., Ltd.
c8c4055-821-4ac6251-90d3ab624c1

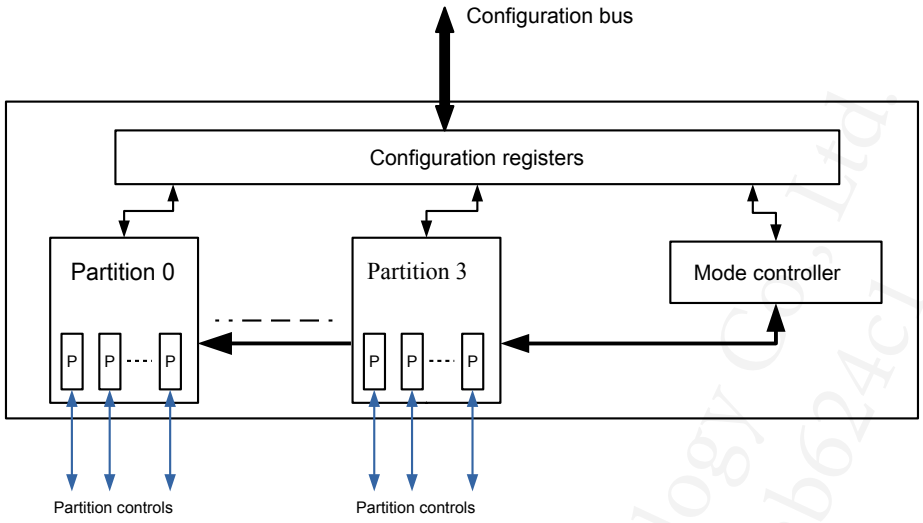


Figure 175. MC_ME block diagram

32.3 Features

MC_ME includes the following features:

- 4 logic partitions implementation and their controls
- Core clock controls
- Partition clock control
- Partition output safe stating control
- Control mechanism for initiating a destructive or functional reset sequence to MC_RGM
- Control mechanism for initiating standby mode entry for SoC

The logic partition inside MC_ME refers to a certain group of on-chip resources (or IP blocks) that are clubbed together to represent a single 'Partition' inside MC_ME. The MC_ME partition can be the same or different than an LBIST partition. Each of the MC_ME partitions implements a certain number of hardware processes. These hardware processes provide a mechanism to regulate various control signals provided to or received from the IP blocks. The corresponding status signals can also be monitored from MC_ME register(s). Each of the hardware processes is bound to finish in 256 cycles of the MC_ME register configuration clocks. Therefore, the hardware processes are non blocking in nature. Mismatch in the expected versus actual status of any hardware process is controlled by a pre-defined software.

32.4 MC_ME register descriptions

MC_ME implements set hardware processes that can be used by the software for changing the mode of operation for a partition. Following are the features of MC_ME registers:

- All registers are 32-bit wide.
- Only 32-bit read and write accesses are supported.
- Read/write accesses of less than 32 bits do not terminate with an error, and hardware behavior is unpredictable.
- Writes to read-only register fields in writable registers are ignored and do not provide an error message.
- Writes to read-only registers are aborted with an error message.

32.4.1 MC_ME memory map

MC_ME base address: 4008_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control Key Register (CTL_KEY)	32	RW	0000_5AF0h
4h	Mode Configuration Register (MODE_CONF)	32	RW	0000_0000h
8h	Mode Update Register (MODE_UPD)	32	RW	0000_0000h
Ch	Mode Status Register (MODE_STAT)	32	RO	0000_0000h
10h	Main Core ID Register (MAIN_COREID)	32	RW	0000_0000h
100h	Partition 0 Process Configuration Register (PRTN0_PCONF)	32	RW	0000_0001h
104h	Partition 0 Process Update Register (PRTN0_PUPD)	32	RW	0000_0000h
108h	Partition 0 Status Register (PRTN0_STAT)	32	RO	0000_0001h
110h	Partition 0 COFB Set 0 Clock Status Register (PRTN0_COFB0_STAT)	32	RO	0000_0000h
130h	Partition 0 COFB Set 0 Clock Enable Register (PRTN0_COFB0_CLKEN)	32	RW	0000_0000h
140h	Partition 0 Core 0 Process Configuration Register (PRTN0_CORE0_PCONF)	32	RW	0000_0000h
144h	Partition 0 Core 0 Process Update Register (PRTN0_CORE0_PUPD)	32	RW	0000_0000h
148h	Partition 0 Core 0 Status Register (PRTN0_CORE0_STAT)	32	RO	0000_0000h
14Ch	Partition 0 Core 0 Address Register (PRTN0_CORE0_ADDR)	32	RW	0000_0000h
160h	Partition 0 Core 1 Process Configuration Register (PRTN0_CORE1_PCONF)	32	RW	0000_0000h
164h	Partition 0 Core 1 Process Update Register (PRTN0_CORE1_PUPD)	32	RW	0000_0000h
168h	Partition 0 Core 1 Status Register (PRTN0_CORE1_STAT)	32	RO	0000_0000h
16Ch	Partition 0 Core 1 Address Register (PRTN0_CORE1_ADDR)	32	RW	0000_0000h
180h	Partition 0 Core 2 Process Configuration Register (PRTN0_CORE2_PCONF)	32	RW	0000_0000h
184h	Partition 0 Core 2 Process Update Register (PRTN0_CORE2_PUPD)	32	RW	0000_0000h
188h	Partition 0 Core 2 Status Register (PRTN0_CORE2_STAT)	32	RO	0000_0000h
18Ch	Partition 0 Core 2 Address Register (PRTN0_CORE2_ADDR)	32	RW	0000_0000h
1A0h	Partition 0 Core 3 Process Configuration Register (PRTN0_CORE3_PCONF)	32	RW	0000_0000h
1A4h	Partition 0 Core 3 Process Update Register (PRTN0_CORE3_PUPD)	32	RW	0000_0000h
1A8h	Partition 0 Core 3 Status Register (PRTN0_CORE3_STAT)	32	RO	0000_0000h
1ACh	Partition 0 Core 3 Address Register (PRTN0_CORE3_ADDR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
300h	Partition 1 Process Configuration Register (PRTN1_PCONF)	32	RW	0000_0004h
304h	Partition 1 Process Update Register (PRTN1_PUPD)	32	RW	0000_0000h
308h	Partition 1 Status Register (PRTN1_STAT)	32	RO	0000_0004h
340h	Partition 1 Core 0 Process Configuration Register (PRTN1_CORE0_PCONF)	32	RW	0000_0000h
344h	Partition 1 Core 0 Process Update Register (PRTN1_CORE0_PUPD)	32	RW	0000_0000h
348h	Partition 1 Core 0 Status Register (PRTN1_CORE0_STAT)	32	RO	0000_0000h
34Ch	Partition 1 Core 0 Address Register (PRTN1_CORE0_ADDR)	32	RW	0000_0000h
360h	Partition 1 Core 1 Process Configuration Register (PRTN1_CORE1_PCONF)	32	RW	0000_0000h
364h	Partition 1 Core 1 Process Update Register (PRTN1_CORE1_PUPD)	32	RW	0000_0000h
368h	Partition 1 Core 1 Status Register (PRTN1_CORE1_STAT)	32	RO	0000_0000h
36Ch	Partition 1 Core 1 Address Register (PRTN1_CORE1_ADDR)	32	RW	0000_0000h
380h	Partition 1 Core 2 Process Configuration Register (PRTN1_CORE2_PCONF)	32	RW	0000_0000h
384h	Partition 1 Core 2 Process Update Register (PRTN1_CORE2_PUPD)	32	RW	0000_0000h
388h	Partition 1 Core 2 Status Register (PRTN1_CORE2_STAT)	32	RO	0000_0000h
38Ch	Partition 1 Core 2 Address Register (PRTN1_CORE2_ADDR)	32	RW	0000_0000h
3A0h	Partition 1 Core 3 Process Configuration Register (PRTN1_CORE3_PCONF)	32	RW	0000_0000h
3A4h	Partition 1 Core 3 Process Update Register (PRTN1_CORE3_PUPD)	32	RW	0000_0000h
3A8h	Partition 1 Core 3 Status Register (PRTN1_CORE3_STAT)	32	RO	0000_0000h
3ACh	Partition 1 Core 3 Address Register (PRTN1_CORE3_ADDR)	32	RW	0000_0000h
500h	Partition 2 Process Configuration Register (PRTN2_PCONF)	32	RW	0000_0004h
504h	Partition 2 Process Update Register (PRTN2_PUPD)	32	RW	0000_0000h
508h	Partition 2 Status Register (PRTN2_STAT)	32	RO	0000_0004h
510h	Partition 2 COFB Set 0 Clock Status Register (PRTN2_COFB0_STAT)	32	RO	0000_0000h
530h	Partition 2 COFB Set 0 Clock Enable Register (PRTN2_COFB0_CLKEN)	32	RW	0000_0000h
700h	Partition 3 Process Configuration Register (PRTN3_PCONF)	32	RW	0000_0004h
704h	Partition 3 Process Update Register (PRTN3_PUPD)	32	RW	0000_0000h
708h	Partition 3 Status Register (PRTN3_STAT)	32	RO	0000_0004h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
710h	Partition 3 COFB Set 0 Clock Status Register (PRTN3_COFB0_STAT)	32	RO	0000_0000h
730h	Partition 3 COFB Set 0 Clock Enable Register (PRTN3_COFB0_CLKEN)	32	RW	0000_0000h

32.4.2 Control Key Register (CTL_KEY)

Offset

Register	Offset
CTL_KEY	0h

Function

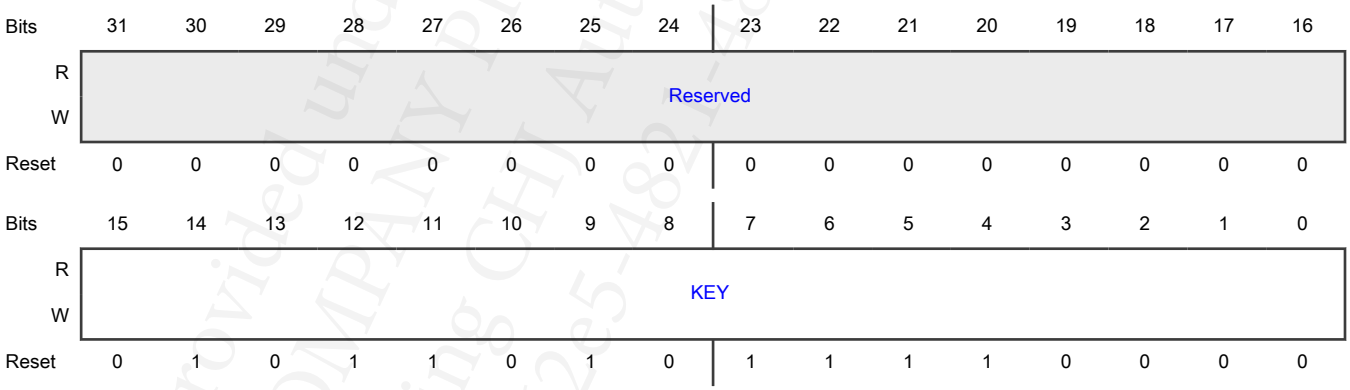
This register provides the mechanism to MC_ME for starting the hardware processes for the partition(s) and standby entry sequence. The hardware processes for partitions are triggered through the corresponding PRTNn_PCONF register. The mechanism to trigger the hardware processes of the respective partitions require two write operations: first time with key and second time with inverted key. The hexadecimal value of key is 0x5AF0 whereas for inverted key is 0xA50F.

For initiating a standby entry sequence, the MODE_CONF register is used for providing a standby entry request along with a valid key combination.

NOTE

Reads from this register return a valid key value to be written next.

Diagram



Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-b25-990d3ab624c1

Fields

Field	Function
31-16 —	Reserved This field is reserved and read returns zeros.
15-0 KEY	Control key Key for starting the hardware processes. Writes with a value other than key or inverted key are ignored. Reads return bit inverted value corresponding to last write.

32.4.3 Mode Configuration Register (MODE_CONF)

Offset

Register	Offset
MODE_CONF	4h

Function

This register is used for initiating a standby request or a reset event (destructive or functional) for the chip. The functional or destructive events are signaled to MC_RGM for further handling.

NOTE

Software must not enable mode entry if the value of multiple fields is 1 in the MODE_CONF register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	STAN DBY	Reserved													FUNC_ RST	DEST_ RST
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 STANDBY	Standby request Writing a logic-1 to this bit along with the MODE_UPD register configuration and followed with a valid key combination makes a standby entry sequence request to MC_ME.
14-2 —	Reserved This field is reserved and read returns zeros.
1 FUNC_RST	Functional reset request Writing a logic-1 to this bit along with the MODE_UPD register configuration and followed with a valid key combination makes a functional reset event signaling to MC_RGM.
0 DEST_RST	Destructive reset request Writing a logic-1 to this bit along with the MODE_UPD register configuration and followed with a valid key combination makes a destructive reset event signaling to MC_RGM.

32.4.4 Mode Update Register (MODE_UPD)

Offset

Register	Offset
MODE_UPD	8h

Function

This register is used for initiating a mode change. Mode change refers to initiating a standby request, or generating a destructive or functional reset event to MC_RGM. Setting mode update field to logic-1, along with programming MODE_CONF registers and then followed by a valid key combination will generate a mode transition request.

NOTE

The MODE_UPD register is implemented to make mode transition programming model the same as partition programming model. This is for future expansion inside MC_ME.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															MODE_UPD
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8a467e5-4828-b251-390d3ab624c1

Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 MODE_UPD	Mode update Writing a logic-1 to this bit, followed by a valid key combination initiates a mode change as per the MODE_CONF register.

32.4.5 Mode Status Register (MODE_STAT)

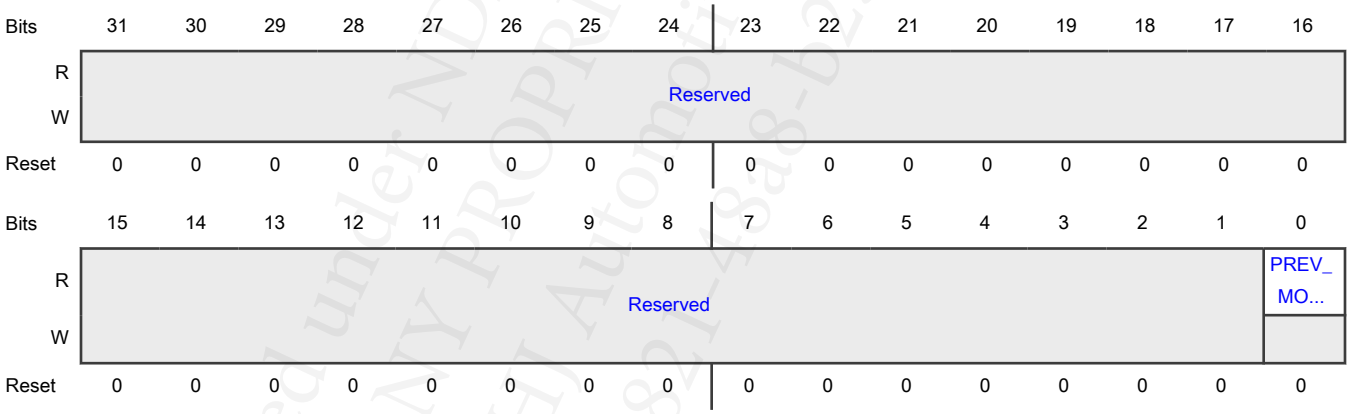
Offset

Register	Offset
MODE_STAT	Ch

Function

This register provides the status of the previous mode. In case of standby exit, if the reset event status register of MC_RGM are set, then contents of this register should be ignored.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 PREV_MODE	Previous mode This bit shows the status of the previous mode.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - The previous mode was reset (any reset). 1b - The previous mode was standby.

32.4.6 Main Core ID Register (MAIN_COREID)

Offset

Register	Offset
MAIN_COREID	10h

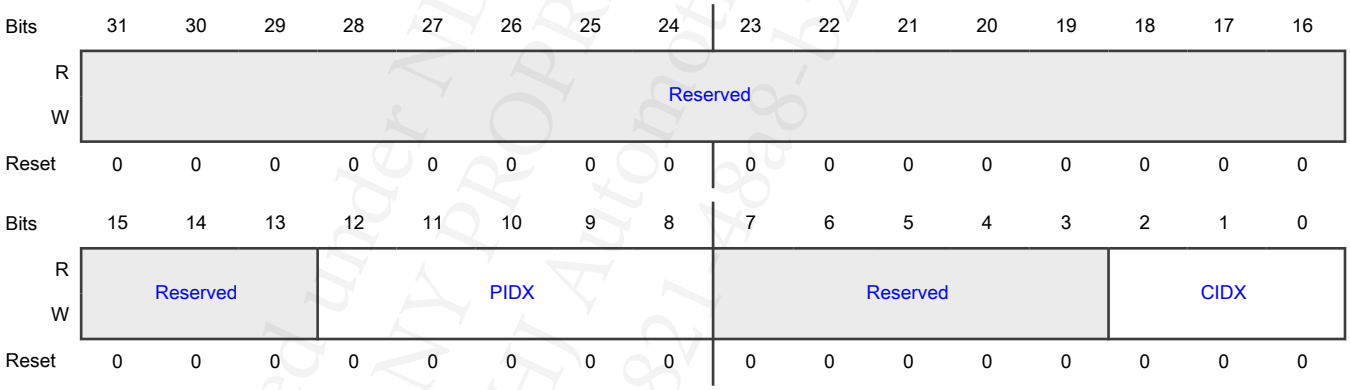
Function

This register provides the ID of the main core sequencing the operation for the standby sequence. Core ID is required for entering in the standby mode, and using this MC_ME locates the WFI instruction execution of the main core. The core ID in this register is specified by the partition index along with the core index.

NOTE

Before initiating a standby entry sequence, the contents of this register should point to the correct main core. Providing non-existing or incorrect core ID leads to unpredictable hardware behavior.

Diagram



Fields

Field	Function
31-13 —	Reserved This field is reserved and read returns zeros.
12-8 PIDX	Partition index Provides the partition index of the main core. Only values 0 - 3 can be written.

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-3 —	Reserved This field is reserved and read returns zeros.
2-0 CIDX	Core index Provides the core index of the main core inside the partition.

32.4.7 Partition 0 Process Configuration Register (PRTN0_PCONF)

Offset

Register	Offset
PRTN0_PCONF	100h

Function

This register provides a configuration for the hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_PCONF and PRTN0_PUPD registers are used to determine the hardware processes to be executed. These are triggered in parallel and independent of each other. All dependent processes should be requested one after another from the software.

NOTE

The partition clock enable/disable and output safe stating enable/disable are not standalone and must be done coherently in a fixed sequence. For details, see Software Reset Partition Turn-On Flow Chart and Software reset partition turn-off flowchart in Reset chapter.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	PCE	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Provided under DISCLAIMER
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-482...-8a8...-123...-90d3ab624c1

Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCE	Partition clock enable This bit controls whether the clock to IPs (other than core(s)) in the partition should be enabled or disabled. The software must not perform any writes to this field. This field must remain unchanged and always equal to its reset value. 0b - Disable the clock to IPs 1b - Enable the clock to IPs

32.4.8 Partition 0 Process Update Register (PRTN0_PUPD)

Offset

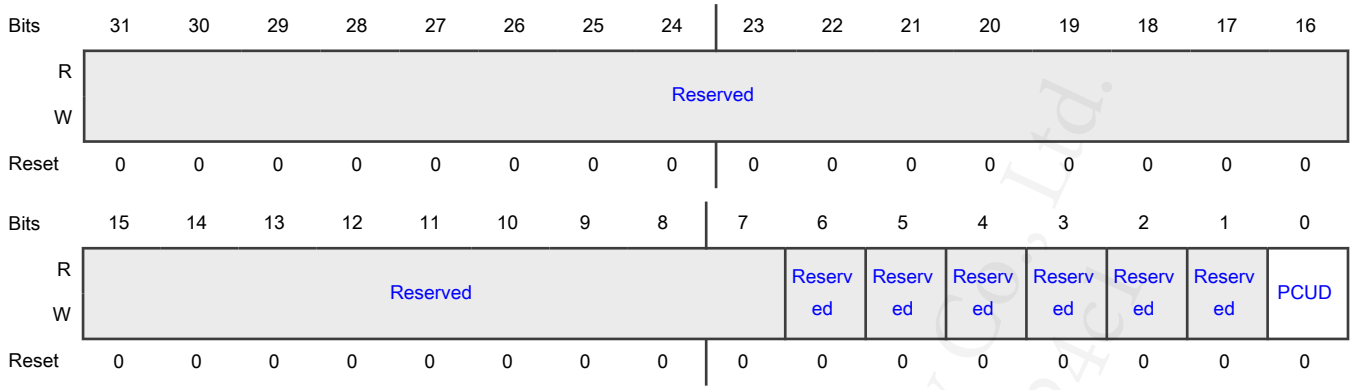
Register	Offset
PRTN0_PUPD	104h

Function

This register provides trigger signaling for the hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_PCONF register. When the hardware process is finished the corresponding bit in this register is auto-cleared to logic-0.

Provided under NDA only
CONTAINS PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c0e625-482-48a8-8251-390d3ab624c1

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1 —	Reserved This field is reserved and read returns zeros.
0 PCUD	Partition clock update This bit controls whether the hardware processes for enabling/disabling the clock to IPs (other than core(s)) in the partition should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4b625-4821-48a8-b251-390d3ab624c3

32.4.9 Partition 0 Status Register (PRTN0_STAT)

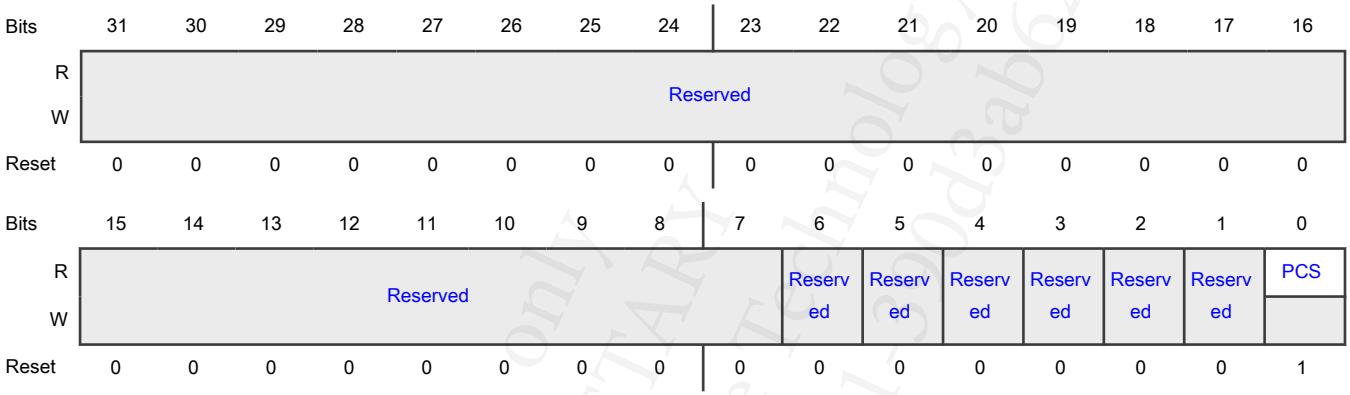
Offset

Register	Offset
PRTN0_STAT	108h

Function

This register provides the current status of the control signals from the partition 0.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 —	Reserved This field is reserved and read returns zeros.
1	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is reserved and read returns zeros.
0 PCS	Partition clock status This bit provides the status of the clock to partition. 0b - Clock is inactive 1b - Clock is active

32.4.10 Partition 0 COFB Set 0 Clock Status Register (PRTN0_COFB0_STAT)

Offset

Register	Offset
PRTN0_COFB0_STAT	110h

Function

This register provides the status of set 0 of COFBs inside partition 0.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK31	BLOCK30	BLOCK29	BLOCK28	BLOCK27	BLOCK26	BLOCK25	BLOCK24	BLOCK23	BLOCK22	BLOCK21	BLOCK20	BLOCK19	BLOCK18	BLOCK17	BLOCK16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK15	BLOCK14	BLOCK13	BLOCK12	BLOCK11	BLOCK10	BLOCK9	BLOCK8	BLOCK7	BLOCK6	BLOCK5	BLOCK4	BLOCK3	BLOCK2	BLOCK1	BLOCK0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BLOCK31	IP block status This bit provides the clock status of block 31 in partition 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
30 BLOCK30	IP block status This bit provides the clock status of block 30 in partition 0. 0b - Clock is not running. 1b - Clock is running.
29 BLOCK29	IP block status This bit provides the clock status of block 29 in partition 0. 0b - Clock is not running. 1b - Clock is running.
28 BLOCK28	IP block status This bit provides the clock status of block 28 in partition 0. 0b - Clock is not running. 1b - Clock is running.
27 BLOCK27	IP block status This bit provides the clock status of block 27 in partition 0. 0b - Clock is not running. 1b - Clock is running.
26 BLOCK26	IP block status This bit provides the clock status of block 26 in partition 0. 0b - Clock is not running. 1b - Clock is running.
25 BLOCK25	IP block status This bit provides the clock status of block 25 in partition 0. 0b - Clock is not running. 1b - Clock is running.
24 BLOCK24	IP block status This bit provides the clock status of block 24 in partition 0. 0b - Clock is not running. 1b - Clock is running.
23	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK23	This bit provides the clock status of block 23 in partition 0. 0b - Clock is not running. 1b - Clock is running.
22 BLOCK22	IP block status This bit provides the clock status of block 22 in partition 0. 0b - Clock is not running. 1b - Clock is running.
21 BLOCK21	IP block status This bit provides the clock status of block 21 in partition 0. 0b - Clock is not running. 1b - Clock is running.
20 BLOCK20	IP block status This bit provides the clock status of block 20 in partition 0. 0b - Clock is not running. 1b - Clock is running.
19 BLOCK19	IP block status This bit provides the clock status of block 19 in partition 0. 0b - Clock is not running. 1b - Clock is running.
18 BLOCK18	IP block status This bit provides the clock status of block 18 in partition 0. 0b - Clock is not running. 1b - Clock is running.
17 BLOCK17	IP block status This bit provides the clock status of block 17 in partition 0. 0b - Clock is not running. 1b - Clock is running.
16 BLOCK16	IP block status This bit provides the clock status of block 16 in partition 0. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 BLOCK15	IP block status This bit provides the clock status of block 15 in partition 0. 0b - Clock is not running. 1b - Clock is running.
14 BLOCK14	IP block status This bit provides the clock status of block 14 in partition 0. 0b - Clock is not running. 1b - Clock is running.
13 BLOCK13	IP block status This bit provides the clock status of block 13 in partition 0. 0b - Clock is not running. 1b - Clock is running.
12 BLOCK12	IP block status This bit provides the clock status of block 12 in partition 0. 0b - Clock is not running. 1b - Clock is running.
11 BLOCK11	IP block status This bit provides the clock status of block 11 in partition 0. 0b - Clock is not running. 1b - Clock is running.
10 BLOCK10	IP block status This bit provides the clock status of block 10 in partition 0. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK9	IP block status This bit provides the clock status of block 9 in partition 0. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK8	IP block status This bit provides the clock status of block 8 in partition 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
7 BLOCK7	IP block status This bit provides the clock status of block 7 in partition 0. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK6	IP block status This bit provides the clock status of block 6 in partition 0. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK5	IP block status This bit provides the clock status of block 5 in partition 0. 0b - Clock is not running. 1b - Clock is running.
4 BLOCK4	IP block status This bit provides the clock status of block 4 in partition 0. 0b - Clock is not running. 1b - Clock is running.
3 BLOCK3	IP block status This bit provides the clock status of block 3 in partition 0. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK2	IP block status This bit provides the clock status of block 2 in partition 0. 0b - Clock is not running. 1b - Clock is running.
1 BLOCK1	IP block status This bit provides the clock status of block 1 in partition 0. 0b - Clock is not running. 1b - Clock is running.
0	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK0	This bit provides the clock status of block 0 in partition 0. 0b - Clock is not running. 1b - Clock is running.

32.4.11 Partition 0 COFB Set 0 Clock Enable Register (PRTN0_COFB0_CLKEN)

Offset

Register	Offset
PRTN0_COFB0_CLKEN	130h

Function

This register provides clock control signaling to the individual COFBs in set 0 inside partition 0. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	REQ31	REQ30	REQ29	REQ28	REQ27	REQ26	REQ25	REQ24	REQ23	REQ22	REQ21	REQ20	REQ19	REQ18	REQ17	REQ16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	REQ15	REQ14	REQ13	REQ12	REQ11	REQ10	REQ9	REQ8	REQ7	REQ6	REQ5	REQ4	REQ3	REQ2	REQ1	REQ0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 REQ31	Clock enable This bit provides the clock enable control for block 31 in partition 0. 0b - Clock is turned off.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Clock is turned on.
30 REQ30	Clock enable This bit provides the clock enable control for block 30 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
29 REQ29	Clock enable This bit provides the clock enable control for block 29 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
28 REQ28	Clock enable This bit provides the clock enable control for block 28 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
27 REQ27	Clock enable This bit provides the clock enable control for block 27 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
26 REQ26	Clock enable This bit provides the clock enable control for block 26 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
25 REQ25	Clock enable This bit provides the clock enable control for block 25 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
24 REQ24	Clock enable This bit provides the clock enable control for block 24 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
23 REQ23	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit provides the clock enable control for block 23 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
22 REQ22	Clock enable This bit provides the clock enable control for block 22 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
21 REQ21	Clock enable This bit provides the clock enable control for block 21 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
20 REQ20	Clock enable This bit provides the clock enable control for block 20 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
19 REQ19	Clock enable This bit provides the clock enable control for block 19 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
18 REQ18	Clock enable This bit provides the clock enable control for block 18 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
17 REQ17	Clock enable This bit provides the clock enable control for block 17 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
16 REQ16	Clock enable This bit provides the clock enable control for block 16 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 REQ15	Clock enable This bit provides the clock enable control for block 15 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
14 REQ14	Clock enable This bit provides the clock enable control for block 14 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
13 REQ13	Clock enable This bit provides the clock enable control for block 13 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
12 REQ12	Clock enable This bit provides the clock enable control for block 12 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
11 REQ11	Clock enable This bit provides the clock enable control for block 11 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
10 REQ10	Clock enable This bit provides the clock enable control for block 10 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ9	Clock enable This bit provides the clock enable control for block 9 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
8 REQ8	Clock enable This bit provides the clock enable control for block 8 in partition 0.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8462e_4821_48a8_b251_390d3ab624c1

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
7 REQ7	Clock enable This bit provides the clock enable control for block 7 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
6 REQ6	Clock enable This bit provides the clock enable control for block 6 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
5 REQ5	Clock enable This bit provides the clock enable control for block 5 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
4 REQ4	Clock enable This bit provides the clock enable control for block 4 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
3 REQ3	Clock enable This bit provides the clock enable control for block 3 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
2 REQ2	Clock enable This bit provides the clock enable control for block 2 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
1 REQ1	Clock enable This bit provides the clock enable control for block 1 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.
0	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
REQ0	This bit provides the clock enable control for block 0 in partition 0. 0b - Clock is turned off. 1b - Clock is turned on.

32.4.12 Partition 0 Core 0 Process Configuration Register (PRTN0_CORE0_PCONF)

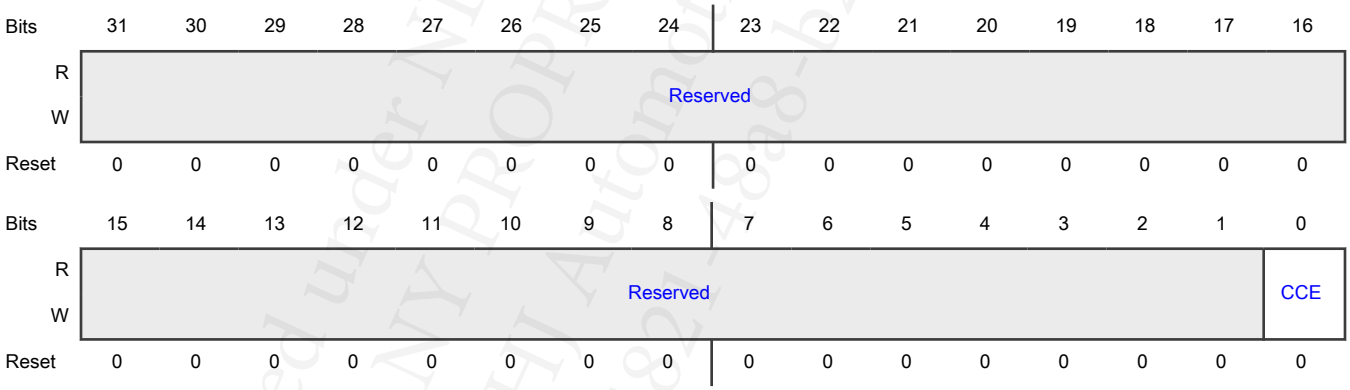
Offset

Register	Offset
PRTN0_CORE0_PCONF	140h

Function

This register provides configurations for the Core 0 hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_CORE0_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_CORE0_PUPD and PRTN0_CORE0_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 0 clock enable This bit controls whether the clock to Core 0 in partition 0 should be enabled or disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable the core clock 1b - Enable the core clock

32.4.13 Partition 0 Core 0 Process Update Register (PRTN0_CORE0_PUPD)

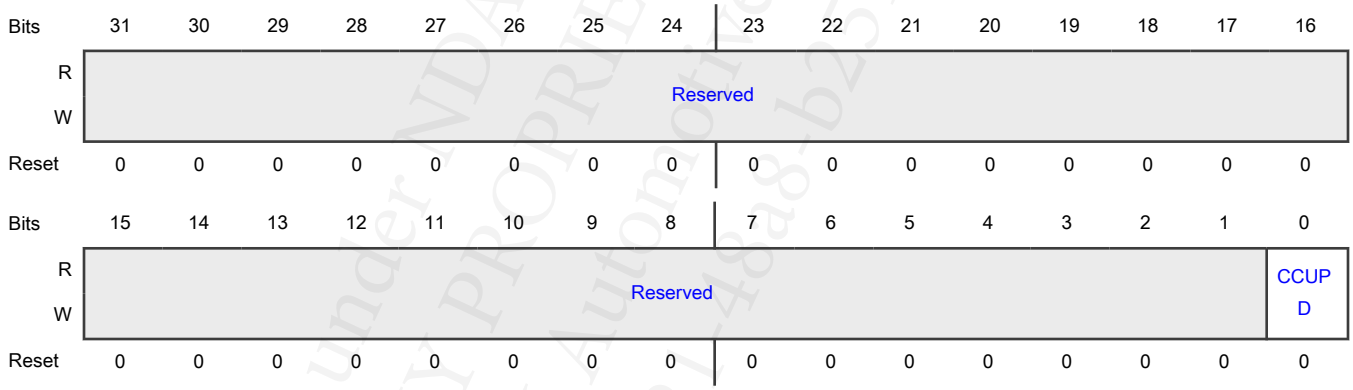
Offset

Register	Offset
PRTN0_CORE0_PUPD	144h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_CORE0_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCUPD	Core 0 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 0 in the partition 0 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

32.4.14 Partition 0 Core 0 Status Register (PRTN0_CORE0_STAT)

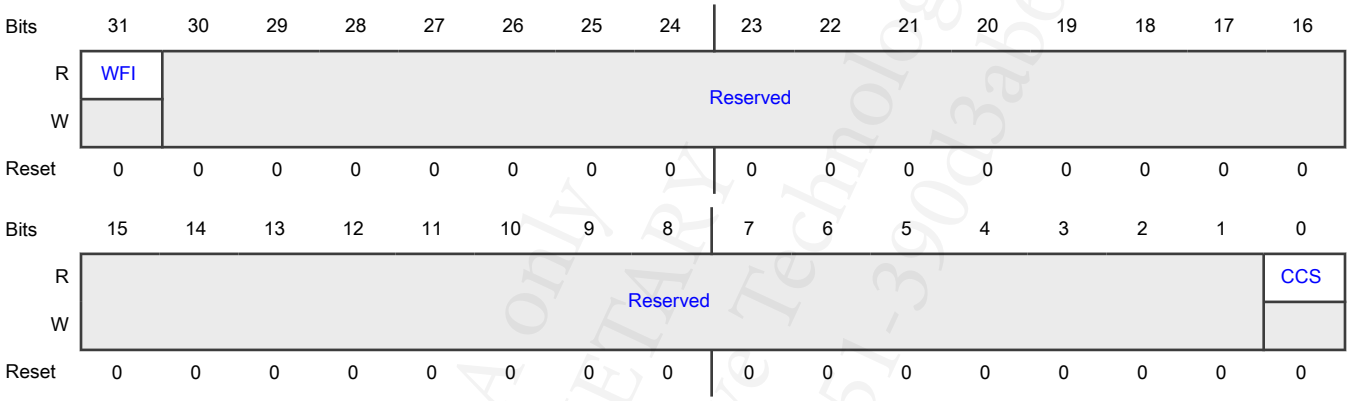
Offset

Register	Offset
PRTN0_CORE0_STAT	148h

Function

This register provides the status corresponding to Core 0 in partition 0. The status signal corresponds to clock states and the WFI signal included from Core 0.

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 0 in partition 0. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 0 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a8-b251-39003ab62401

32.4.15 Partition 0 Core 0 Address Register (PRTN0_CORE0_ADDR)

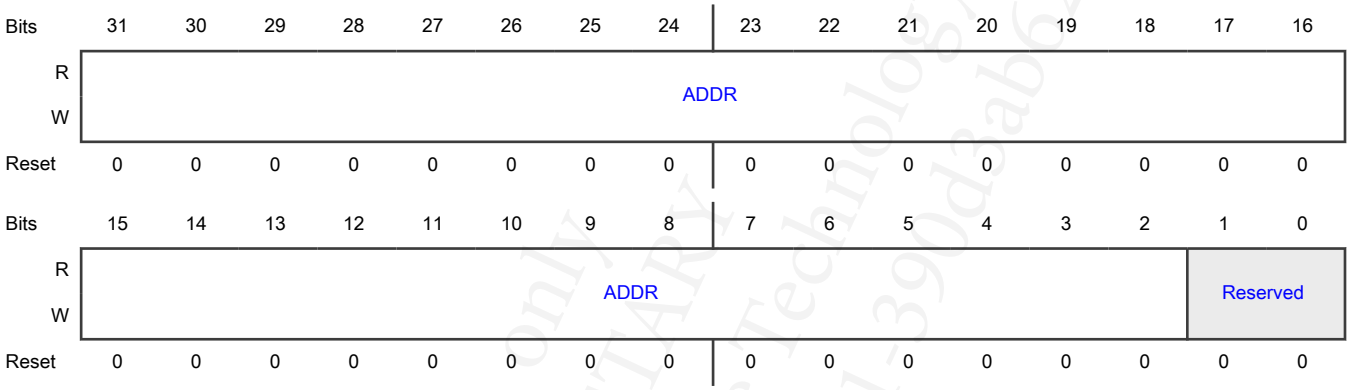
Offset

Register	Offset
PRTN0_CORE0_ADDR	14Ch

Function

This register contains the boot address for Core 0 in partition 0.

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 0 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

32.4.16 Partition 0 Core 1 Process Configuration Register (PRTN0_CORE1_PCONF)

Offset

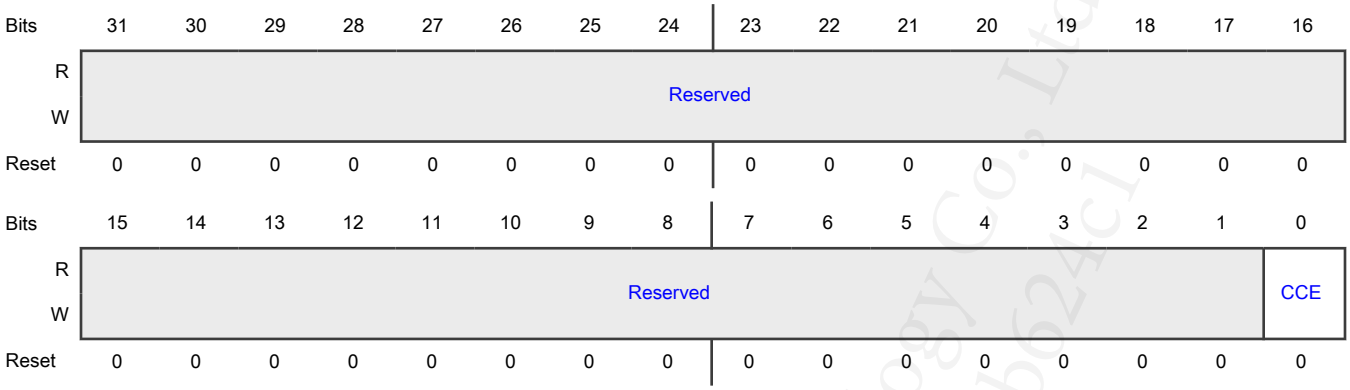
Register	Offset
PRTN0_CORE1_PCONF	160h

Function

This register provides configurations for the Core 1 hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_CORE1_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_CORE1_PUPD and PRTN0_CORE1_PCONF registers are used to determine the hardware processes to be executed.

These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 1 clock enable This bit controls whether the clock to Core 1 in partition 0 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

32.4.17 Partition 0 Core 1 Process Update Register (PRTN0_CORE1_PUPD)

Offset

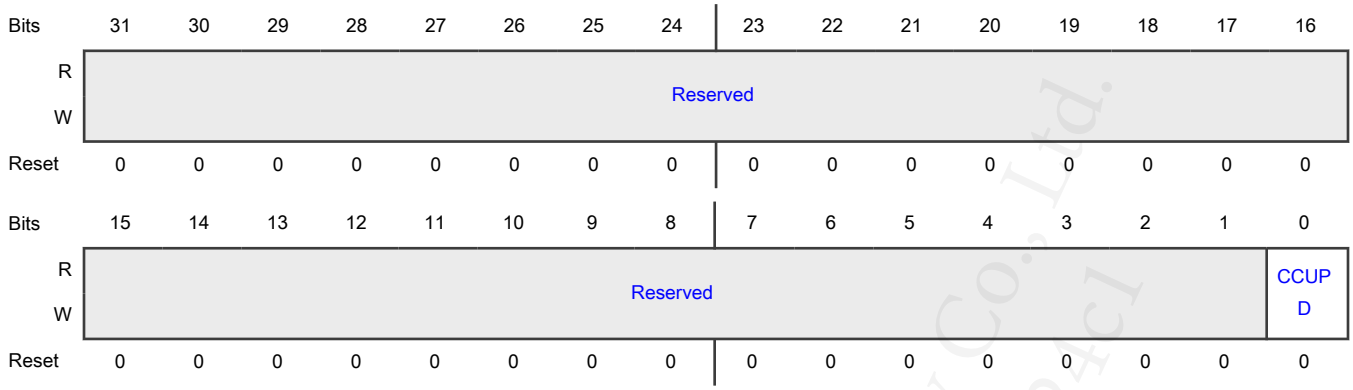
Register	Offset
PRTN0_CORE1_PUPD	164h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_CORE1_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46f2-4821-4828-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCUPD	Core 1 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 1 in the partition 0 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

32.4.18 Partition 0 Core 1 Status Register (PRTN0_CORE1_STAT)

Offset

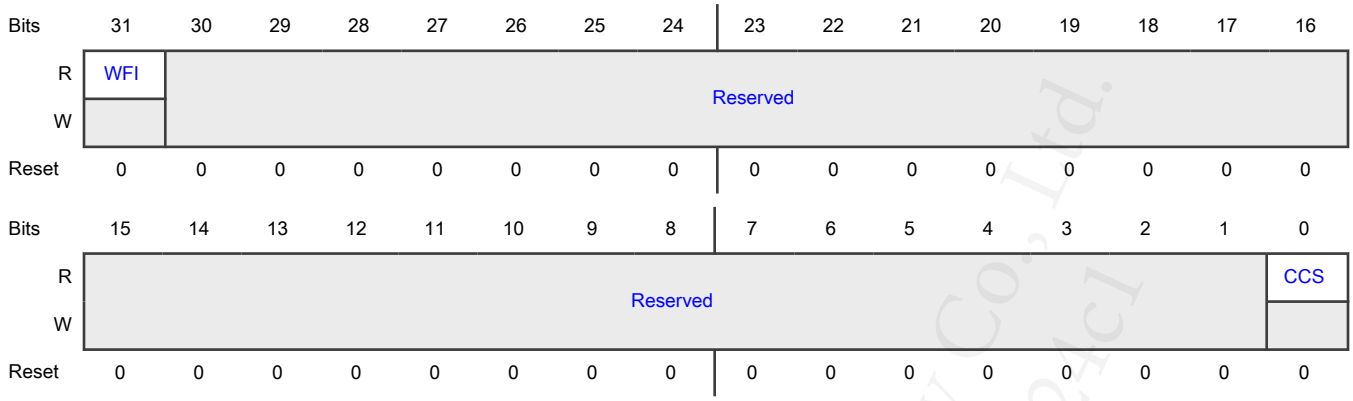
Register	Offset
PRTN0_CORE1_STAT	168h

Function

This register provides the status corresponding to Core 1 in partition 0. The status signal corresponds to clock states and the WFI signal included from Core 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co.,Ltd.
c8c462e5-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 1 in partition 0. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 1 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

32.4.19 Partition 0 Core 1 Address Register (PRTN0_CORE1_ADDR)

Offset

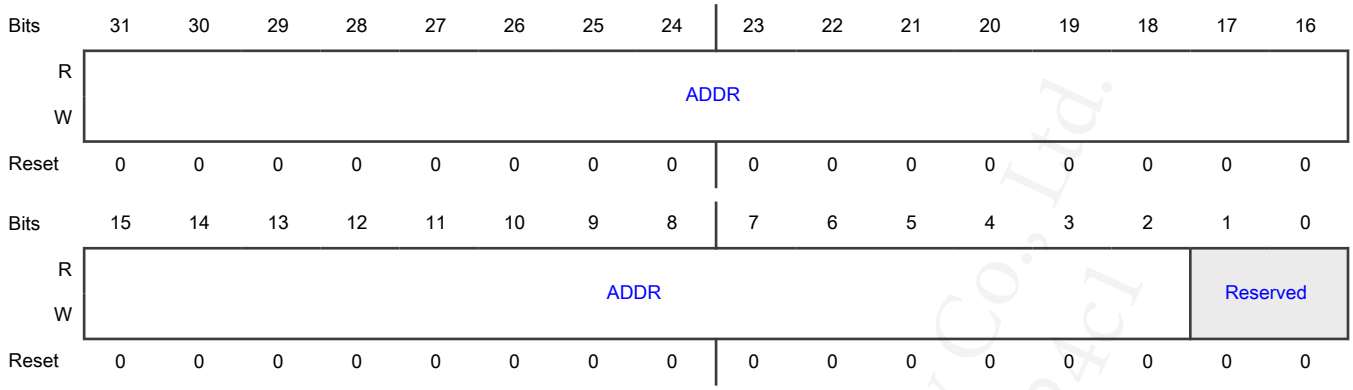
Register	Offset
PRTN0_CORE1_ADDR	16Ch

Function

This register contains the boot address for Core 1 in partition 0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 1 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

32.4.20 Partition 0 Core 2 Process Configuration Register (PRTN0_CORE2_PCONF)

Offset

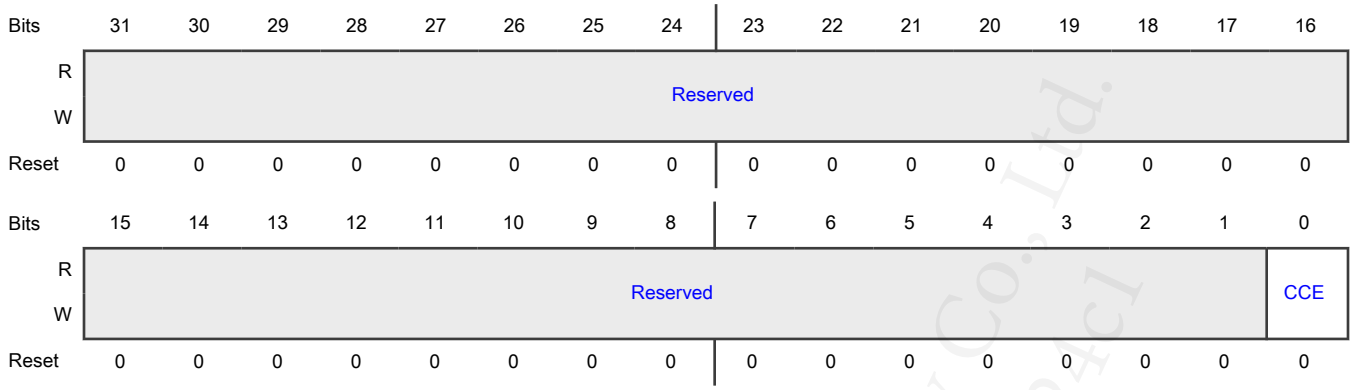
Register	Offset
PRTN0_CORE2_PCONF	180h

Function

This register provides configurations for the Core 2 hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_CORE2_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_CORE2_PUPD and PRTN0_CORE2_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CarRong Automotive Technology Co., Ltd.
c8c462e5-4c1e-48a8-b213-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 2 clock enable This bit controls whether the clock to Core 2 in partition 0 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

32.4.21 Partition 0 Core 2 Process Update Register (PRTN0_CORE2_PUPD)

Offset

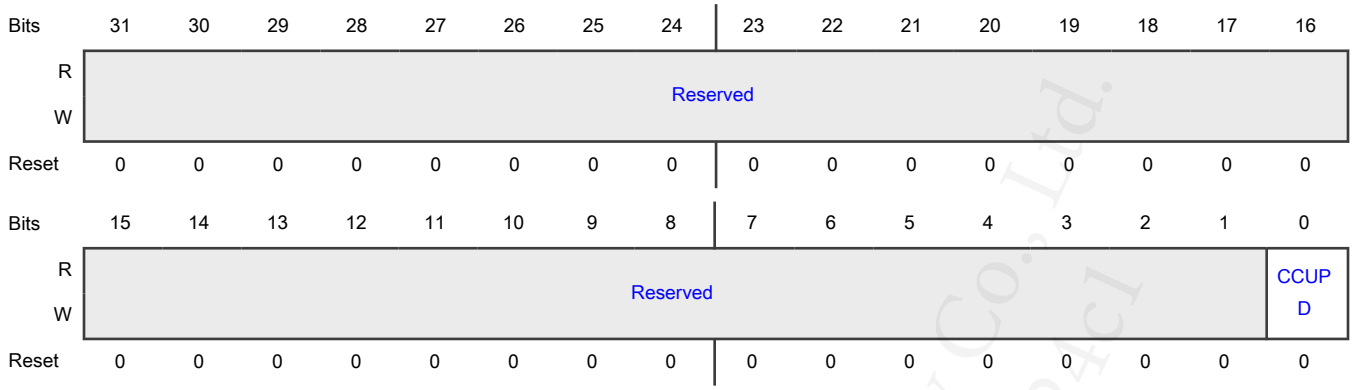
Register	Offset
PRTN0_CORE2_PUPD	184h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_CORE2_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Produced under NDA only
COMPANY PROPRIETARY
Beijing ESAU Automotive Technology Co.,Ltd.
c8c462e5-2214-48a8-251-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCUPD	Core 2 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 2 in the partition 0 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

32.4.22 Partition 0 Core 2 Status Register (PRTN0_CORE2_STAT)

Offset

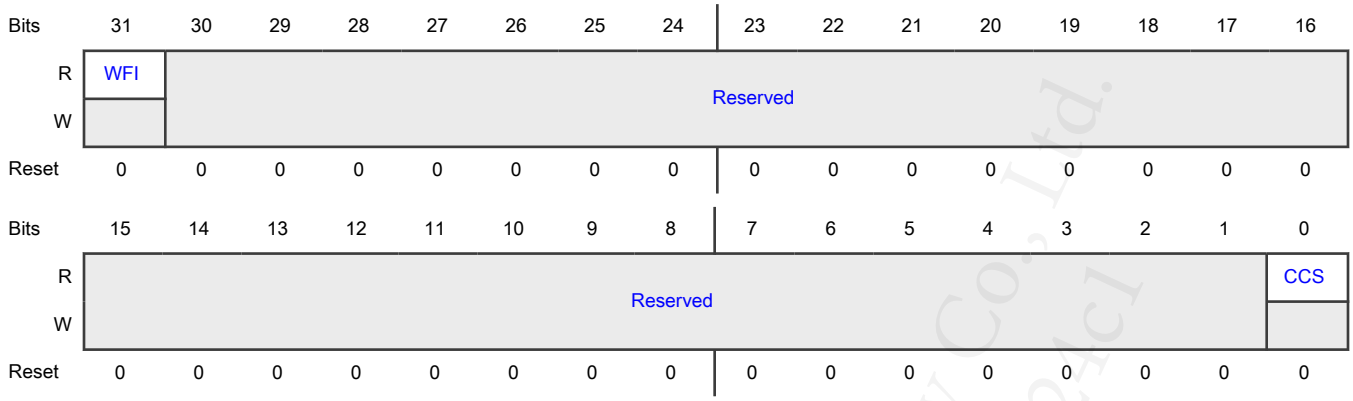
Register	Offset
PRTN0_CORE2_STAT	188h

Function

This register provides the status corresponding to Core 2 in partition 0. The status signal corresponds to clock states and the WFI signal included from Core 2.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co.,Ltd.
c8c462e5-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 2 in partition 0. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 2 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

32.4.23 Partition 0 Core 2 Address Register (PRTN0_CORE2_ADDR)

Offset

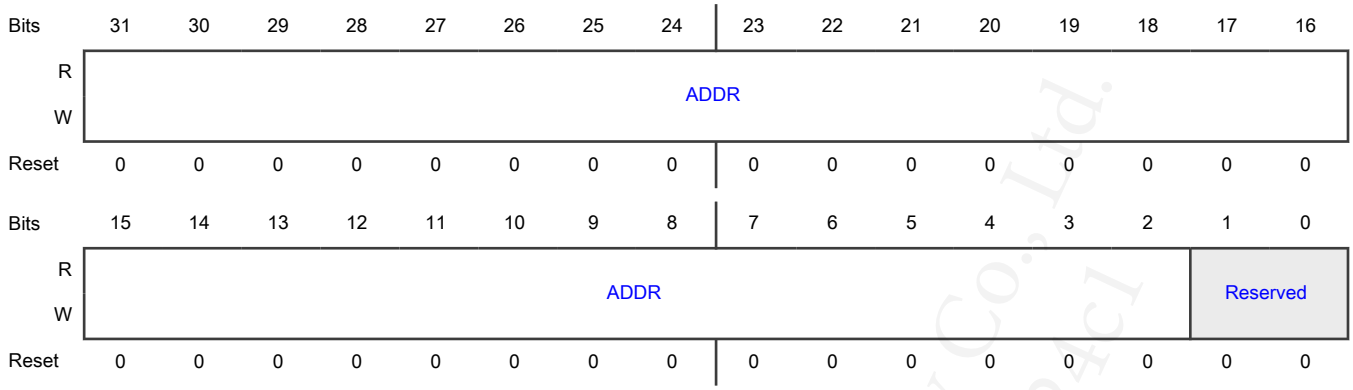
Register	Offset
PRTN0_CORE2_ADDR	18Ch

Function

This register contains the boot address for Core 2 in partition 0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 2 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

32.4.24 Partition 0 Core 3 Process Configuration Register (PRTN0_CORE3_PCONF)

Offset

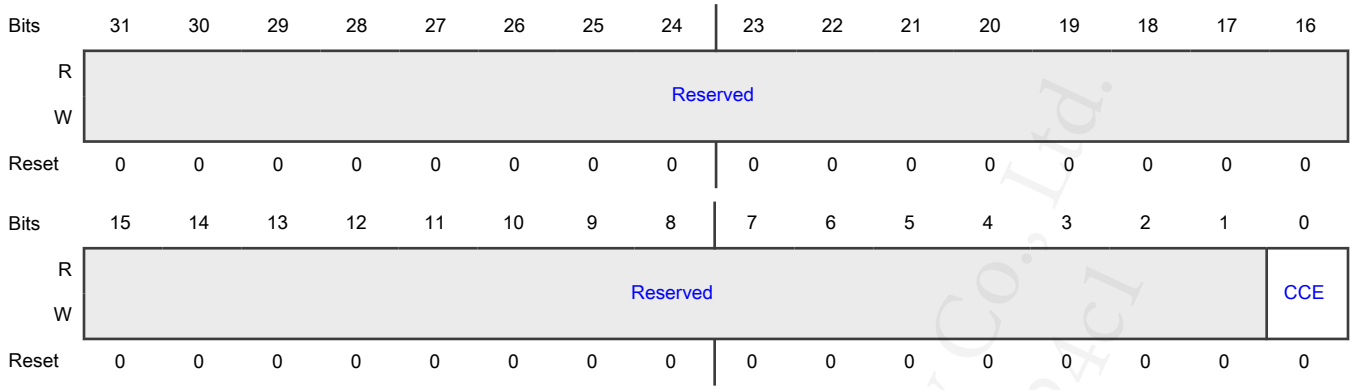
Register	Offset
PRTN0_CORE3_PCONF	1A0h

Function

This register provides configurations for the Core 3 hardware processes corresponding to partition 0. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN0_CORE3_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN0_CORE3_PUPD and PRTN0_CORE3_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Provided under NDA only
COMPANY PROPRIETARY
Beijing Chipsea Automotive Technology Co.,Ltd.
c8c462e5-4c1e-48a8-b213-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 3 clock enable This bit controls whether the clock to Core 3 in partition 0 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

32.4.25 Partition 0 Core 3 Process Update Register (PRTN0_CORE3_PUPD)

Offset

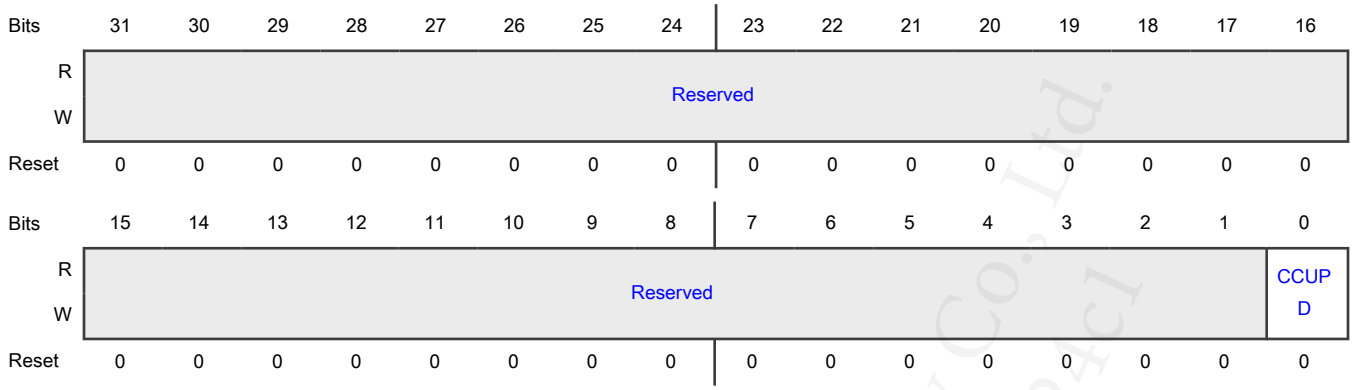
Register	Offset
PRTN0_CORE3_PUPD	1A4h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 0. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN0_CORE3_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Produced under NDA only
COMPANY PROPRIETARY
Beijing eSOL Automotive Technology Co.,Ltd.
c8c462e5-2214-48a8-251-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCUPD	Core 3 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 3 in the partition 0 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

32.4.26 Partition 0 Core 3 Status Register (PRTN0_CORE3_STAT)

Offset

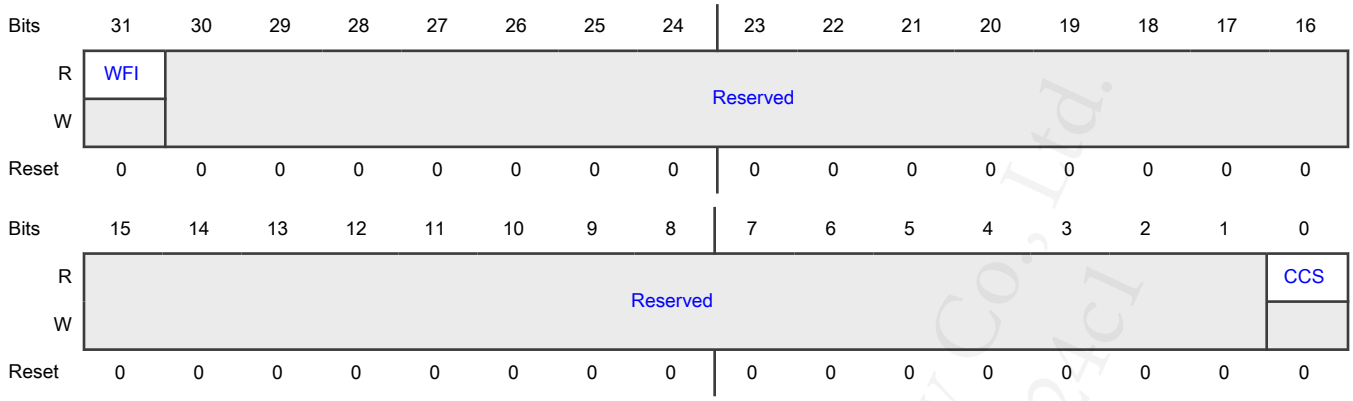
Register	Offset
PRTN0_CORE3_STAT	1A8h

Function

This register provides the status corresponding to Core 3 in partition 0. The status signal corresponds to clock states and the WFI signal included from Core 3.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co.,Ltd.
c8c462e5-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 3 in partition 0. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 3 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

32.4.27 Partition 0 Core 3 Address Register (PRTN0_CORE3_ADDR)

Offset

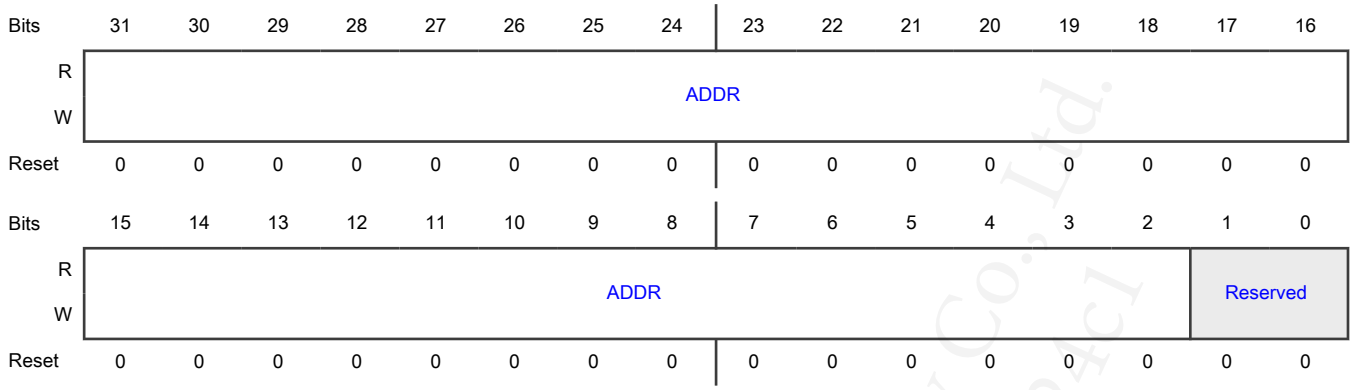
Register	Offset
PRTN0_CORE3_ADDR	1ACh

Function

This register contains the boot address for Core 3 in partition 0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 3 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

32.4.28 Partition 1 Process Configuration Register (PRTN1_PCONF)

Offset

Register	Offset
PRTN1_PCONF	300h

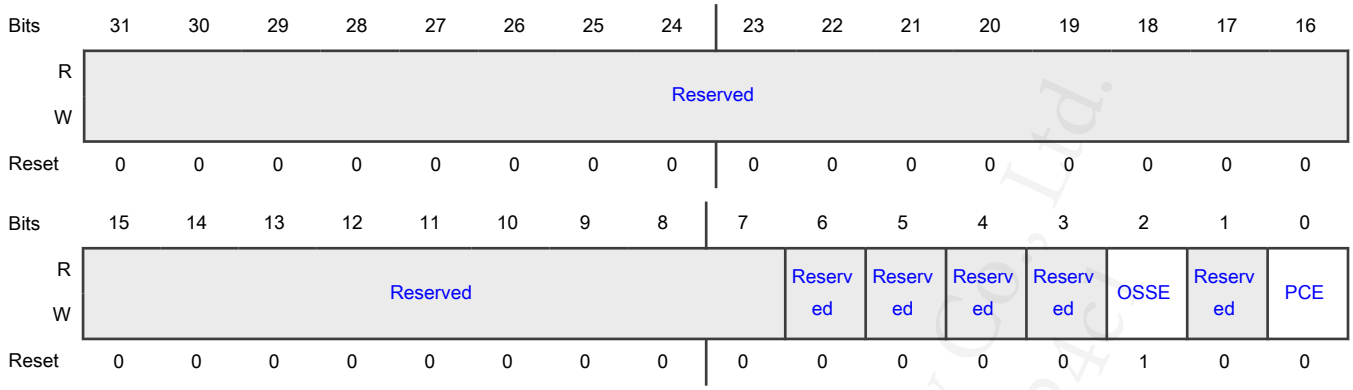
Function

This register provides a configuration for the hardware processes corresponding to partition 1. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN1_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN1_PCONF and PRTN1_PUPD registers are used to determine the hardware processes to be executed. These are triggered in parallel and independent of each other. All dependent processes should be requested one after another from the software.

NOTE

The partition clock enable/disable and output safe stating enable/disable are not standalone and must be done coherently in a fixed sequence. For details, see Software Reset Partition Turn-On Flow Chart and Software reset partition turn-off flowchart in Reset chapter.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 OSSE	Output safe stating enable This bit controls whether the outputs of the partition should be safe stated or not. 0b - Disable output safe stating 1b - Enable output safe stating
1 —	Reserved This field is reserved and read returns zeros.
0 PCE	Partition clock enable This bit controls whether the clock to IPs (other than core(s)) in the partition should be enabled or disabled. 0b - Disable the clock to IPs 1b - Enable the clock to IPs

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cs4c02e_482f_48a8_b251_390d3ab624c

32.4.29 Partition 1 Process Update Register (PRTN1_PUPD)

Offset

Register	Offset
PRTN1_PUPD	304h

Function

This register provides trigger signaling for the hardware processes corresponding to partition 1. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN1_PCONF register. When the hardware process is finished the corresponding bit in this register is auto-cleared to logic-0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								Reserved	Reserved	Reserved	Reserved	OSSUD	Reserved	PCUD	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 OSSUD	Output safe stating update This bit controls whether the hardware processes for enabling/disabling the safe stating for outputs of the partition should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process
1 —	Reserved This field is reserved and read returns zeros.
0 PCUD	Partition clock update This bit controls whether the hardware processes for enabling/disabling the clock to IPs (other than core(s)) in the partition should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

32.4.30 Partition 1 Status Register (PRTN1_STAT)

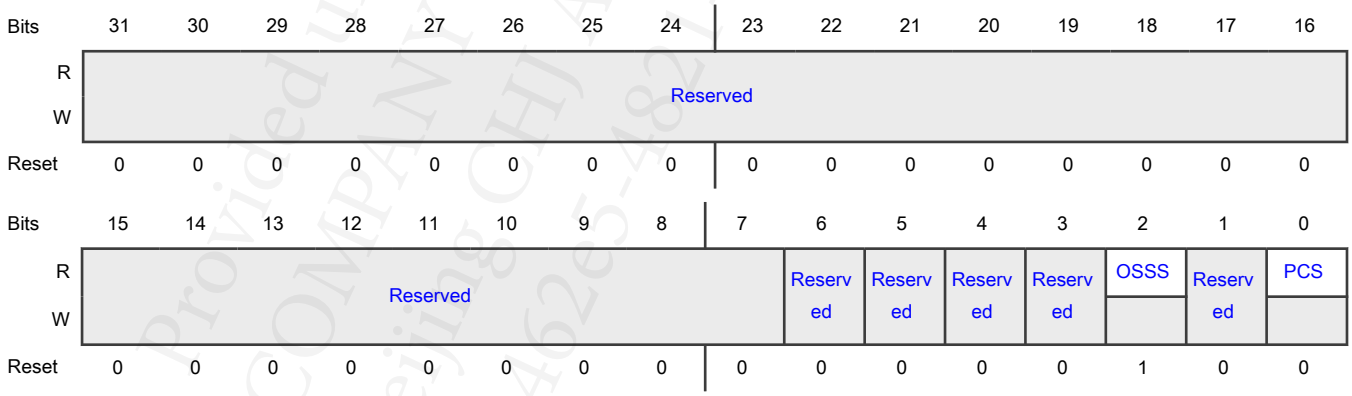
Offset

Register	Offset
PRTN1_STAT	308h

Function

This register provides the current status of the control signals from the partition 1.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482a-4ea8-b251-390d3ab624c1

Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 OSSS	Output safe stating status This bit provides the status for output safe stating of the partition. 0b - Output safe stating is inactive 1b - Output safe stating is active
1 —	Reserved This field is reserved and read returns zeros.
0 PCS	Partition clock status This bit provides the status of the clock to partition. 0b - Clock is inactive 1b - Clock is active

32.4.31 Partition 1 Core 0 Process Configuration Register (PRTN1_CORE0_PCONF)

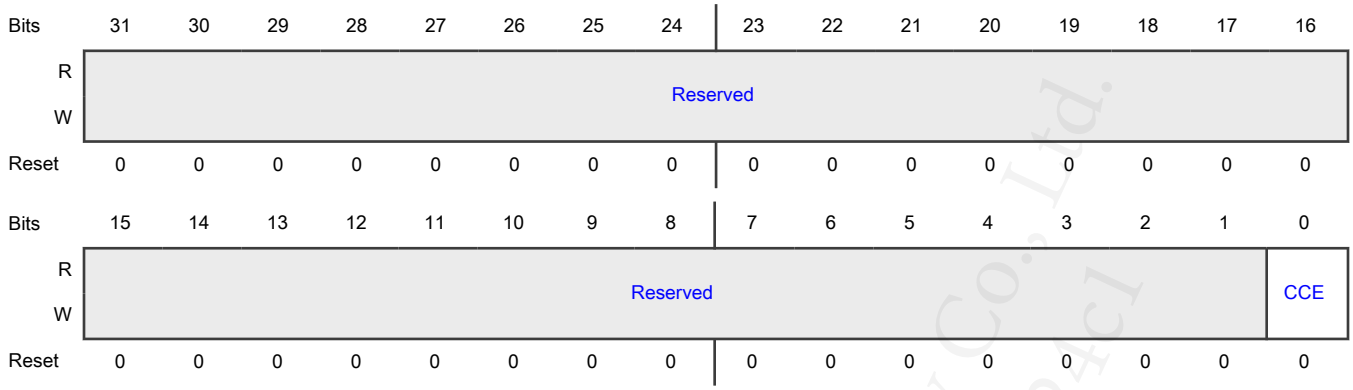
Offset

Register	Offset
PRTN1_CORE0_PCONF	340h

Function

This register provides configurations for the Core 0 hardware processes corresponding to partition 1. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN1_CORE0_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN1_CORE0_PUPD and PRTN1_CORE0_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 0 clock enable This bit controls whether the clock to Core 0 in partition 1 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

32.4.32 Partition 1 Core 0 Process Update Register (PRTN1_CORE0_PUPD)

Offset

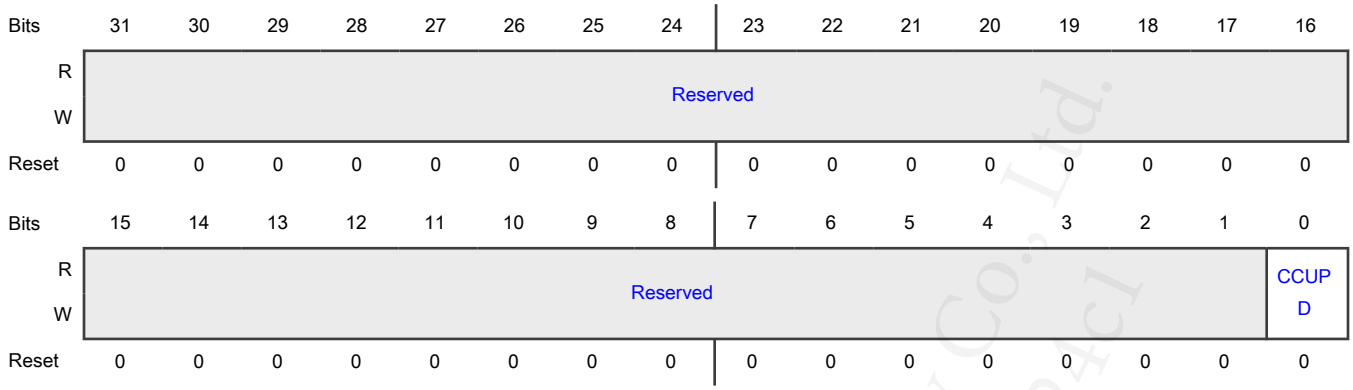
Register	Offset
PRTN1_CORE0_PUPD	344h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 1. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN1_CORE0_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Produced under NDA only
COMPANY PROPRIETARY
Beijing eSOL Automotive Technology Co.,Ltd.
c8c462e5-2214-48a8-251-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCUPD	Core 0 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 0 in the partition 1 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

32.4.33 Partition 1 Core 0 Status Register (PRTN1_CORE0_STAT)

Offset

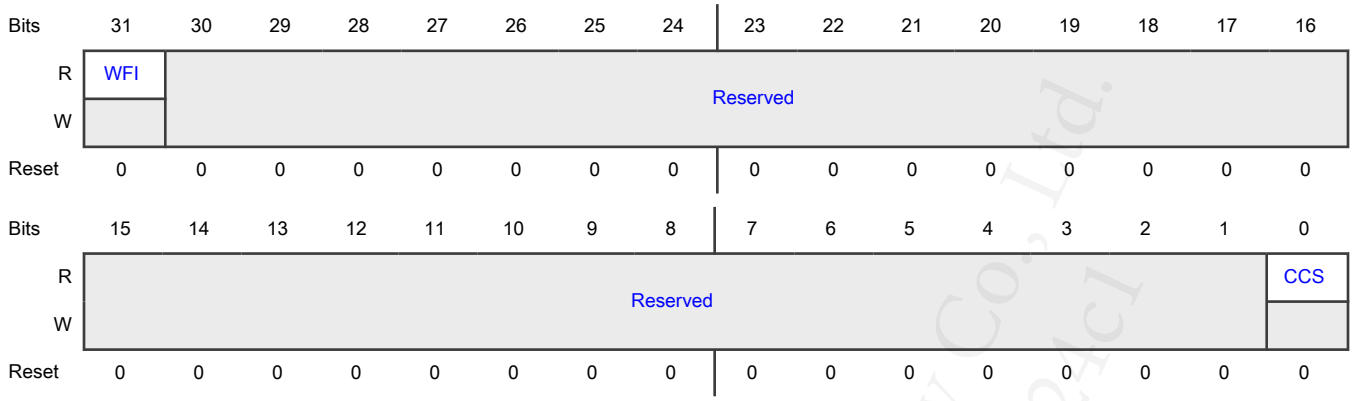
Register	Offset
PRTN1_CORE0_STAT	348h

Function

This register provides the status corresponding to Core 0 in partition 1. The status signal corresponds to clock states and the WFI signal included from Core 0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co.,Ltd.
c8c462e5-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 0 in partition 1. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 0 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

32.4.34 Partition 1 Core 0 Address Register (PRTN1_CORE0_ADDR)

Offset

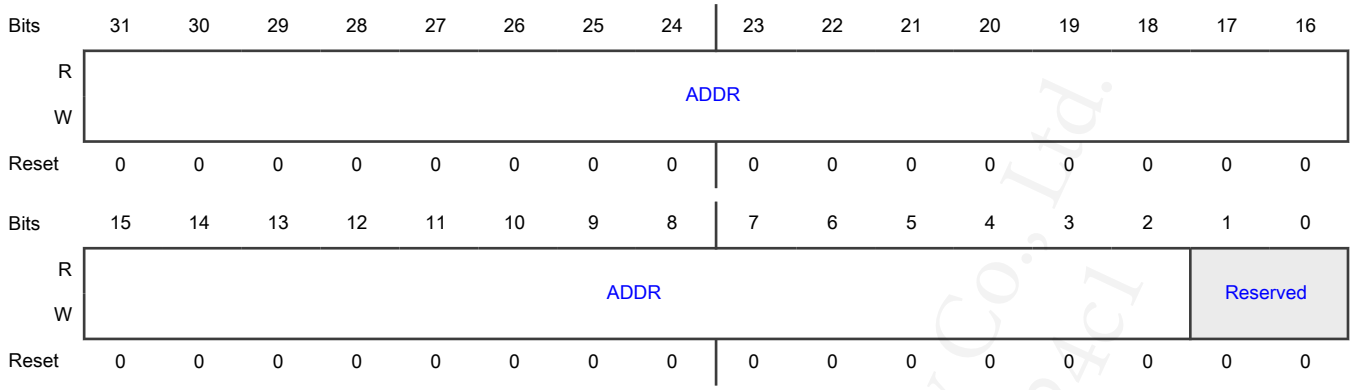
Register	Offset
PRTN1_CORE0_ADDR	34Ch

Function

This register contains the boot address for Core 0 in partition 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 0 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

32.4.35 Partition 1 Core 1 Process Configuration Register (PRTN1_CORE1_PCONF)

Offset

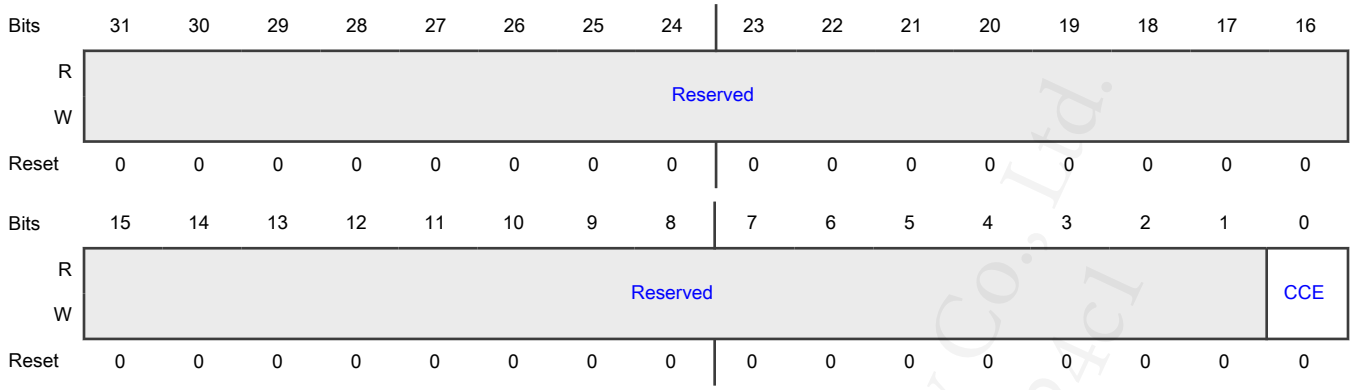
Register	Offset
PRTN1_CORE1_PCONF	360h

Function

This register provides configurations for the Core 1 hardware processes corresponding to partition 1. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN1_CORE1_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN1_CORE1_PUPD and PRTN1_CORE1_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Provided under NDA only
COMPANY PROPRIETARY
Beijing Chipsea Automotive Technology Co.,Ltd.
c8c462e5-4c1e-48a8-b213-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 1 clock enable This bit controls whether the clock to Core 1 in partition 1 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

32.4.36 Partition 1 Core 1 Process Update Register (PRTN1_CORE1_PUPD)

Offset

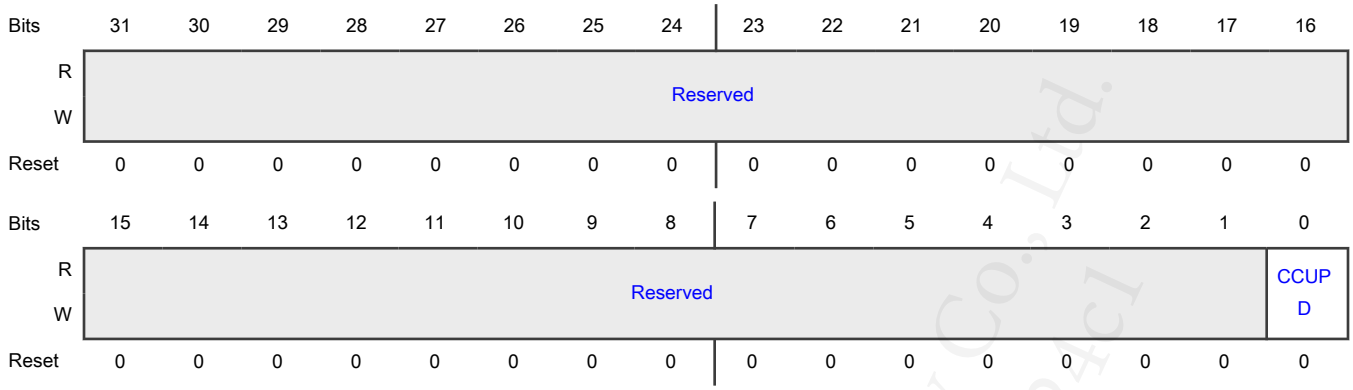
Register	Offset
PRTN1_CORE1_PUPD	364h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 1. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN1_CORE1_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Pro ...d under NDA only
CONFIDENTIAL PROPRIETARY
Beijing ... U Automotive Technology Co.,Ltd.
c8c462e5-21...48a8-251-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCUPD	Core 1 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 1 in the partition 1 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

32.4.37 Partition 1 Core 1 Status Register (PRTN1_CORE1_STAT)

Offset

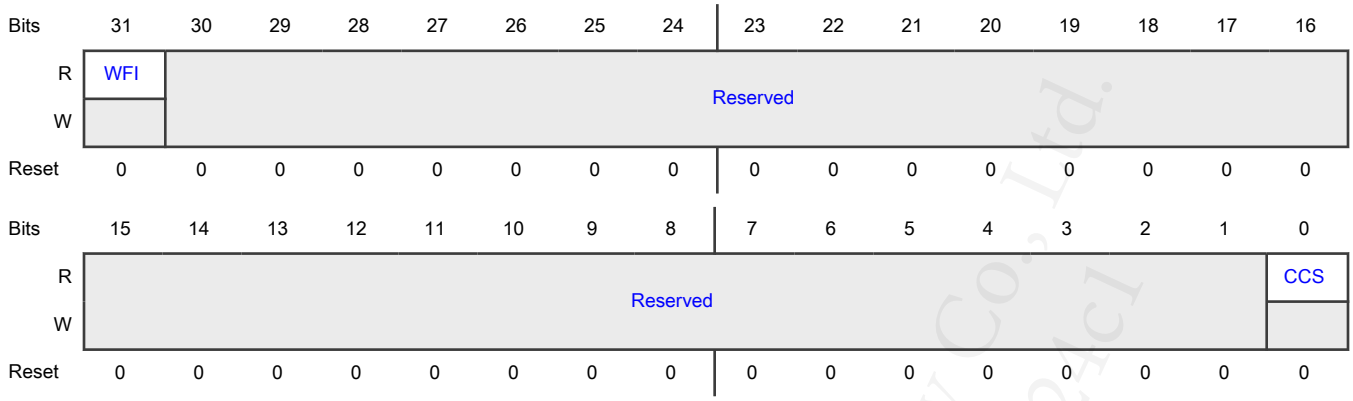
Register	Offset
PRTN1_CORE1_STAT	368h

Function

This register provides the status corresponding to Core 1 in partition 1. The status signal corresponds to clock states and the WFI signal included from Core 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co.,Ltd.
c8c462e5-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 1 in partition 1. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 1 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

32.4.38 Partition 1 Core 1 Address Register (PRTN1_CORE1_ADDR)

Offset

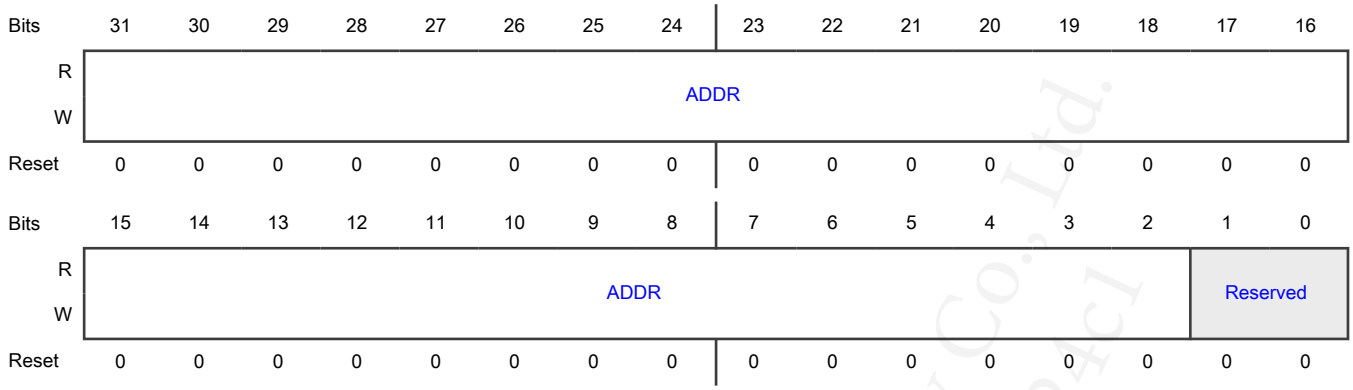
Register	Offset
PRTN1_CORE1_ADDR	36Ch

Function

This register contains the boot address for Core 1 in partition 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 1 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

32.4.39 Partition 1 Core 2 Process Configuration Register (PRTN1_CORE2_PCONF)

Offset

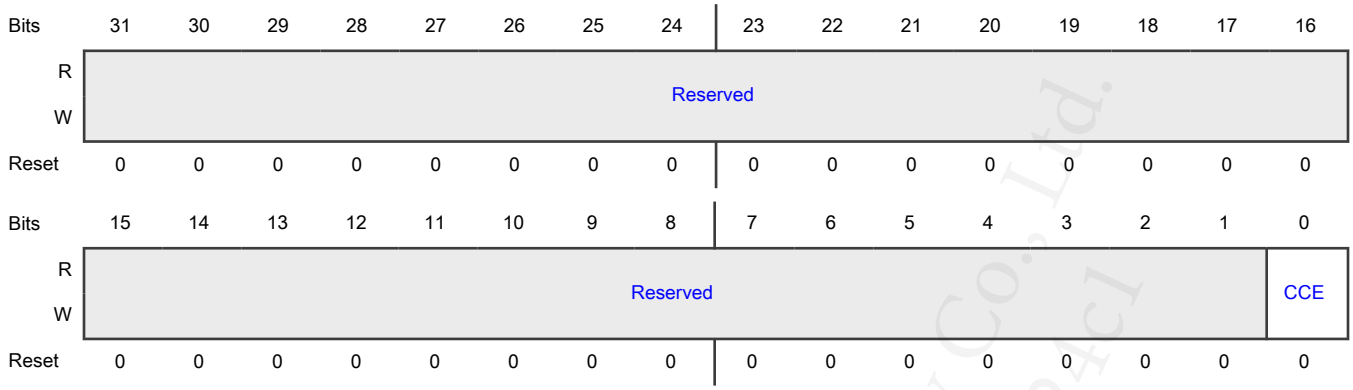
Register	Offset
PRTN1_CORE2_PCONF	380h

Function

This register provides configurations for the Core 2 hardware processes corresponding to partition 1. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN1_CORE2_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN1_CORE2_PUPD and PRTN1_CORE2_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Provided under NDA only
COMPANY PROPRIETARY
Beijing Chipsea Automotive Technology Co.,Ltd.
c8c462e5-4c1e-48a8-b213-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 2 clock enable This bit controls whether the clock to Core 2 in partition 1 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

32.4.40 Partition 1 Core 2 Process Update Register (PRTN1_CORE2_PUPD)

Offset

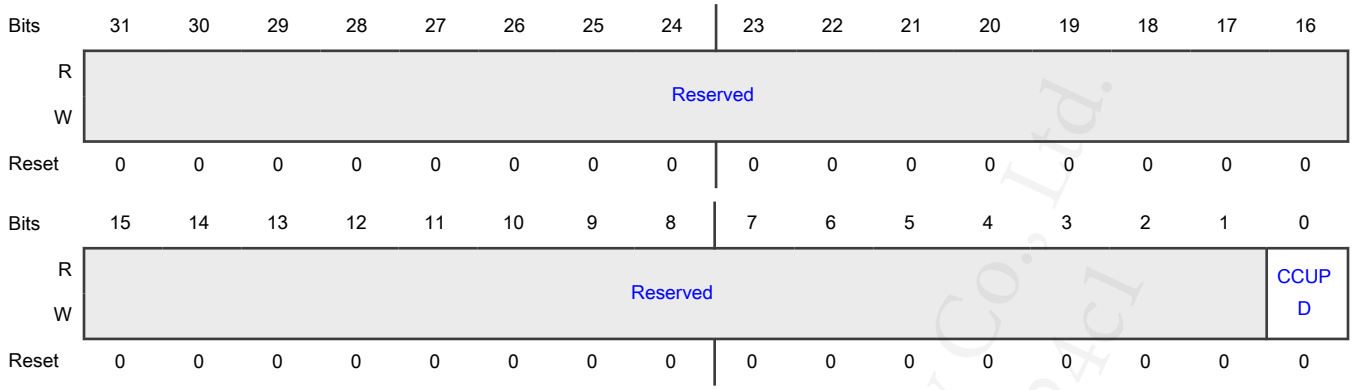
Register	Offset
PRTN1_CORE2_PUPD	384h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 1. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN1_CORE2_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Produced under NDA only
COMPANY PROPRIETARY
Beijing eSOL Automotive Technology Co.,Ltd.
c8c462e5-2214-48a8-251-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCUPD	Core 2 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 2 in the partition 1 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

32.4.41 Partition 1 Core 2 Status Register (PRTN1_CORE2_STAT)

Offset

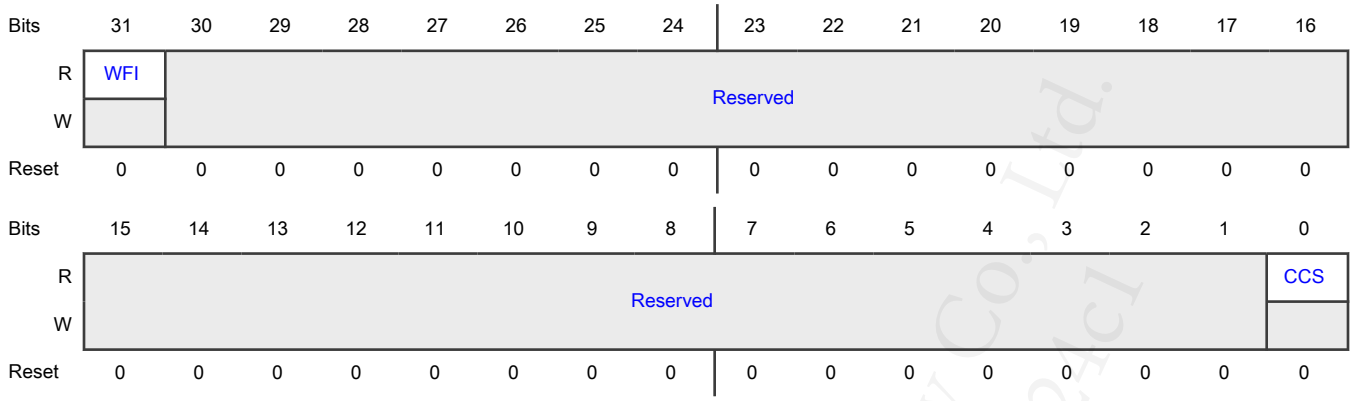
Register	Offset
PRTN1_CORE2_STAT	388h

Function

This register provides the status corresponding to Core 2 in partition 1. The status signal corresponds to clock states and the WFI signal included from Core 2.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co.,Ltd.
c8c462e5-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 2 in partition 1. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 2 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

32.4.42 Partition 1 Core 2 Address Register (PRTN1_CORE2_ADDR)

Offset

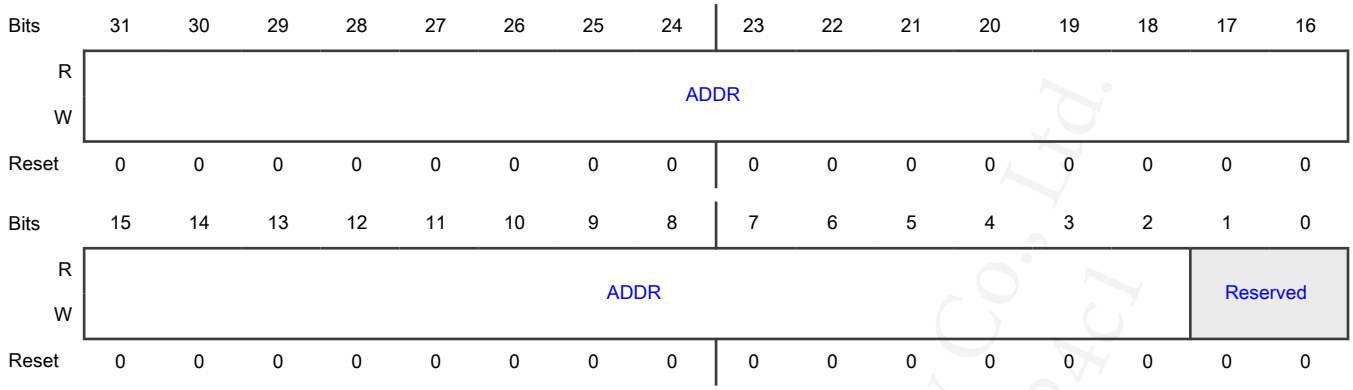
Register	Offset
PRTN1_CORE2_ADDR	38Ch

Function

This register contains the boot address for Core 2 in partition 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 2 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

32.4.43 Partition 1 Core 3 Process Configuration Register (PRTN1_CORE3_PCONF)

Offset

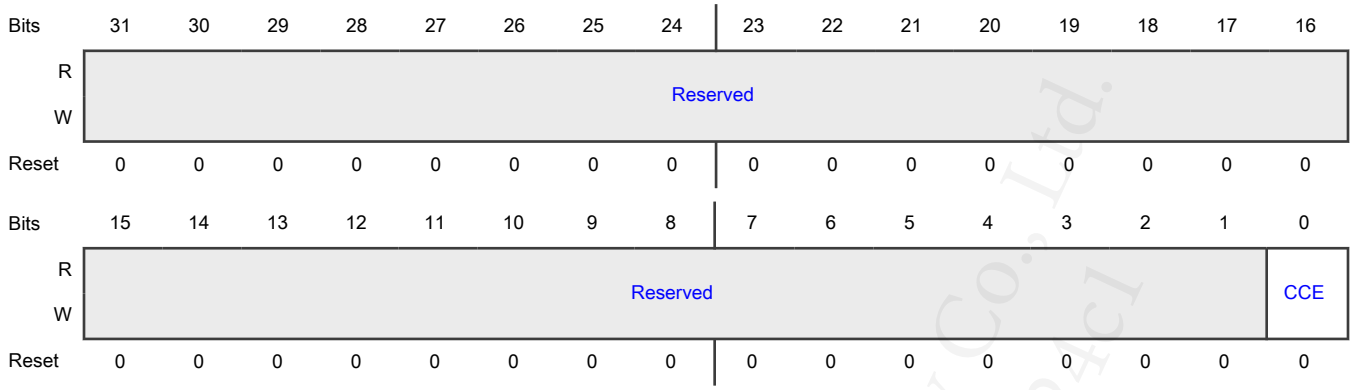
Register	Offset
PRTN1_CORE3_PCONF	3A0h

Function

This register provides configurations for the Core 3 hardware processes corresponding to partition 1. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN1_CORE3_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN1_CORE3_PUPD and PRTN1_CORE3_PCONF registers are used to determine the hardware processes to be executed. These processes are triggered in parallel and are independent of each other. All dependent processes should be requested one after another from the software.

Provided under NDA only
COMPANY PROPRIETARY
Beijing Chipsea Automotive Technology Co.,Ltd.
c8c462e5-4c1e-48a8-b21f-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCE	Core 3 clock enable This bit controls whether the clock to Core 3 in partition 1 should be enabled or disabled. 0b - Disable the core clock 1b - Enable the core clock

32.4.44 Partition 1 Core 3 Process Update Register (PRTN1_CORE3_PUPD)

Offset

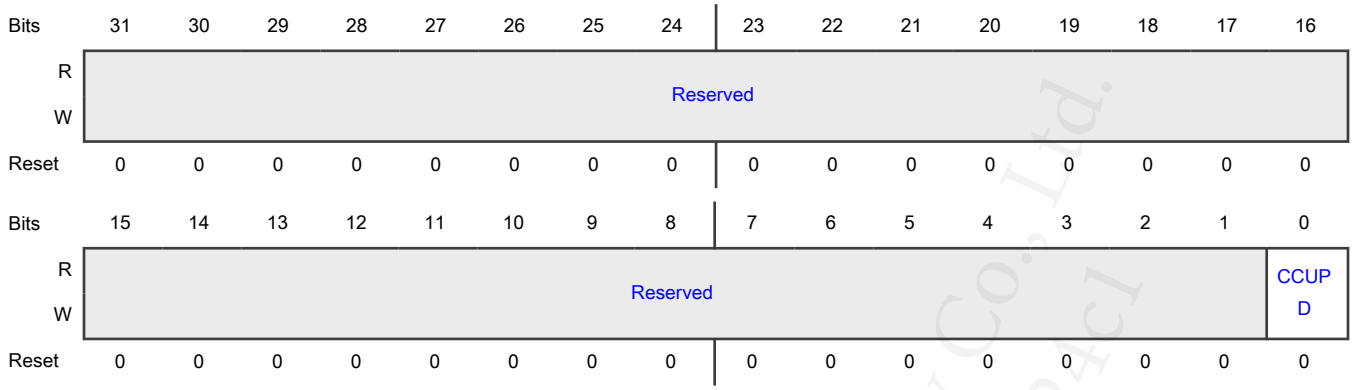
Register	Offset
PRTN1_CORE3_PUPD	3A4h

Function

This register provides trigger signaling for the core hardware processes corresponding to partition 1. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN1_CORE3_PCONF register. When the hardware process is finished, the corresponding bit in this register is auto-cleared to logic-0.

Produced under NDA only
COMPANY PROPRIETARY
Beijing ESUN Automotive Technology Co.,Ltd.
c8c462e5-2214-48a8-251-390d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved This field is reserved and read returns zeros.
0 CCUPD	Core 3 clock update This bit controls whether the hardware processes for enabling/disabling the clock to Core 3 in the partition 1 should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

32.4.45 Partition 1 Core 3 Status Register (PRTN1_CORE3_STAT)

Offset

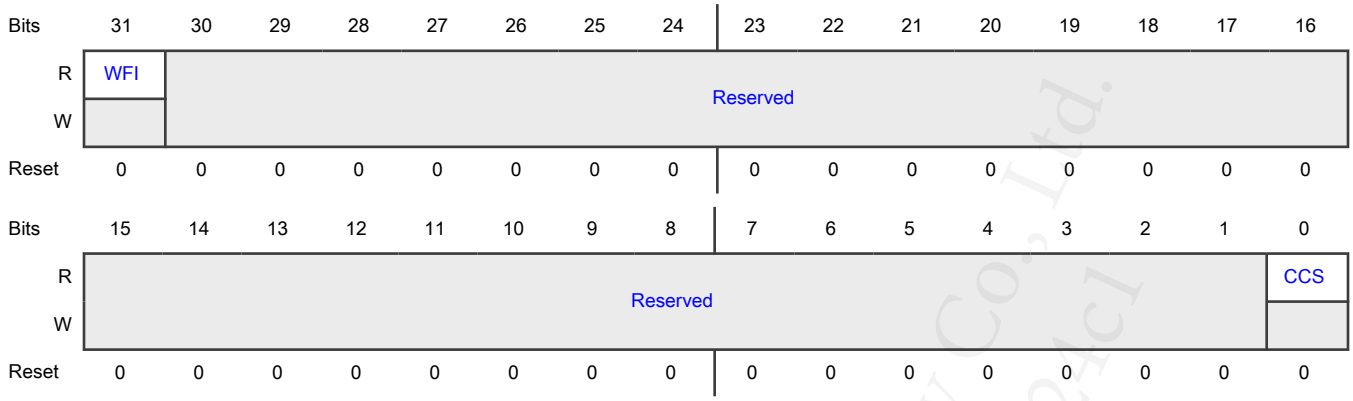
Register	Offset
PRTN1_CORE3_STAT	3A8h

Function

This register provides the status corresponding to Core 3 in partition 1. The status signal corresponds to clock states and the WFI signal included from Core 3.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co.,Ltd.
c8c462e5-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 WFI	Wait for interrupt status This bit provides the WFI status approaching from Core 3 in partition 1. 0b - No WFI executed 1b - WFI executed
30-1 —	Reserved This field is reserved and read returns zeros.
0 CCS	Core 3 clock process status This bit provides the status of the clock corresponding to core clock enablement/disablement. 0b - Clock is inactive. 1b - Clock is active.

32.4.46 Partition 1 Core 3 Address Register (PRTN1_CORE3_ADDR)

Offset

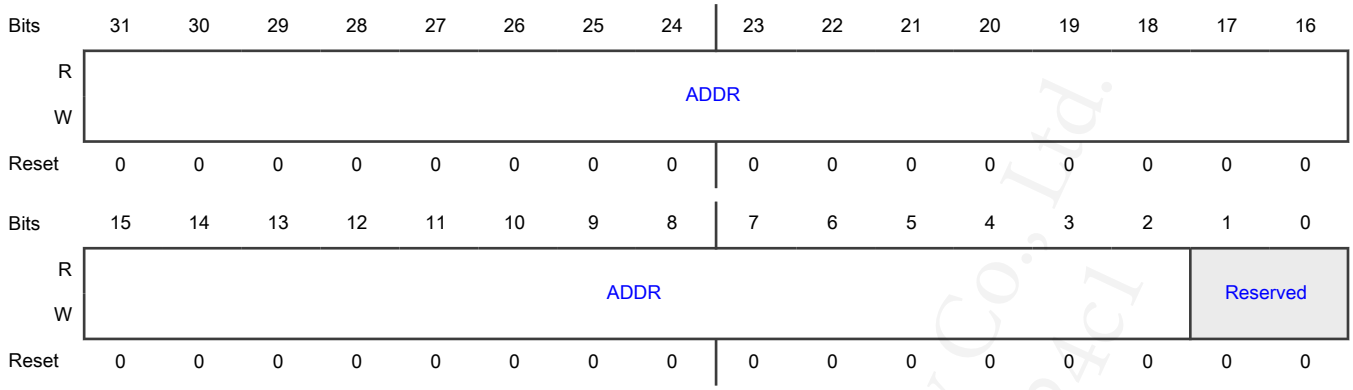
Register	Offset
PRTN1_CORE3_ADDR	3ACh

Function

This register contains the boot address for Core 3 in partition 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 ADDR	Address Core 3 boot address
1-0 —	Reserved This field is reserved and read returns zeros.

32.4.47 Partition 2 Process Configuration Register (PRTN2_PCONF)

Offset

Register	Offset
PRTN2_PCONF	500h

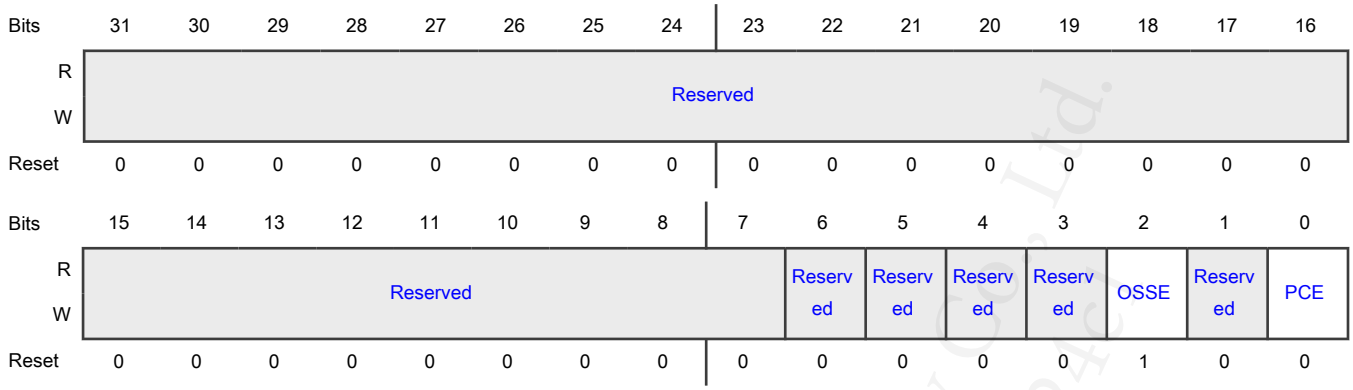
Function

This register provides a configuration for the hardware processes corresponding to partition 2. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN2_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN2_PCONF and PRTN2_PUPD registers are used to determine the hardware processes to be executed. These are triggered in parallel and independent of each other. All dependent processes should be requested one after another from the software.

NOTE

The partition clock enable/disable and output safe stating enable/disable are not standalone and must be done coherently in a fixed sequence. For details, see Software Reset Partition Turn-On Flow Chart and Software reset partition turn-off flowchart in Reset chapter.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 OSSE	Output safe stating enable This bit controls whether the outputs of the partition should be safe stated or not. 0b - Disable output safe stating 1b - Enable output safe stating
1 —	Reserved This field is reserved and read returns zeros.
0 PCE	Partition clock enable This bit controls whether the clock to IPs (other than core(s)) in the partition should be enabled or disabled. 0b - Disable the clock to IPs 1b - Enable the clock to IPs

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cs402e_482f-48a8-b251-390d3ab624c

32.4.48 Partition 2 Process Update Register (PRTN2_PUPD)

Offset

Register	Offset
PRTN2_PUPD	504h

Function

This register provides trigger signaling for the hardware processes corresponding to partition 2. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN2_PCONF register. When the hardware process is finished the corresponding bit in this register is auto-cleared to logic-0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								Reserved	Reserved	Reserved	Reserved	OSSUD	Reserved	PCUD	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 OSSUD	Output safe stating update This bit controls whether the hardware processes for enabling/disabling the safe stating for outputs of the partition should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process
1 —	Reserved This field is reserved and read returns zeros.
0 PCUD	Partition clock update This bit controls whether the hardware processes for enabling/disabling the clock to IPs (other than core(s)) in the partition should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

32.4.49 Partition 2 Status Register (PRTN2_STAT)

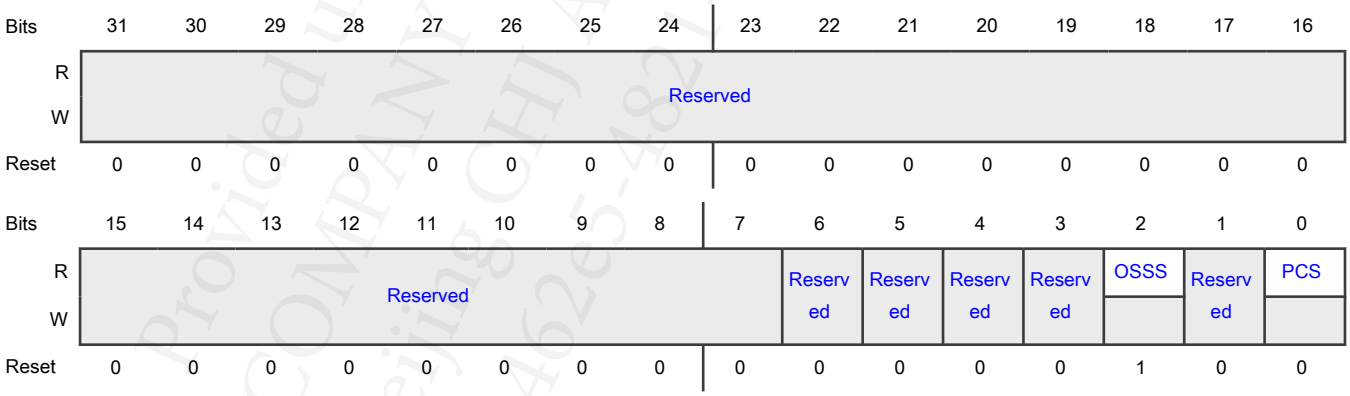
Offset

Register	Offset
PRTN2_STAT	508h

Function

This register provides the current status of the control signals from the partition 2.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482a-4ea8-b251-390d3ab624c1

Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 OSSS	Output safe stating status This bit provides the status for output safe stating of the partition. 0b - Output safe stating is inactive 1b - Output safe stating is active
1 —	Reserved This field is reserved and read returns zeros.
0 PCS	Partition clock status This bit provides the status of the clock to partition. 0b - Clock is inactive 1b - Clock is active

32.4.50 Partition 2 COFB Set 0 Clock Status Register (PRTN2_COFB0_STAT)

Offset

Register	Offset
PRTN2_COFB0_STAT	510h

Function

This register provides the status of set 0 of COFBs inside partition 2.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 2.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c162e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK31	BLOCK30	BLOCK29	BLOCK28	BLOCK27	BLOCK26	BLOCK25	BLOCK24	BLOCK23	BLOCK22	BLOCK21	BLOCK20	BLOCK19	BLOCK18	BLOCK17	BLOCK16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK15	BLOCK14	BLOCK13	BLOCK12	BLOCK11	BLOCK10	BLOCK9	BLOCK8	BLOCK7	BLOCK6	BLOCK5	BLOCK4	BLOCK3	BLOCK2	BLOCK1	BLOCK0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BLOCK31	IP block status This bit provides the clock status of block 31 in partition 2. 0b - Clock is not running. 1b - Clock is running.
30 BLOCK30	IP block status This bit provides the clock status of block 30 in partition 2. 0b - Clock is not running. 1b - Clock is running.
29 BLOCK29	IP block status This bit provides the clock status of block 29 in partition 2. 0b - Clock is not running. 1b - Clock is running.
28 BLOCK28	IP block status This bit provides the clock status of block 28 in partition 2. 0b - Clock is not running. 1b - Clock is running.
27 BLOCK27	IP block status This bit provides the clock status of block 27 in partition 2. 0b - Clock is not running. 1b - Clock is running.
26	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK26	This bit provides the clock status of block 26 in partition 2. 0b - Clock is not running. 1b - Clock is running.
25 BLOCK25	IP block status This bit provides the clock status of block 25 in partition 2. 0b - Clock is not running. 1b - Clock is running.
24 BLOCK24	IP block status This bit provides the clock status of block 24 in partition 2. 0b - Clock is not running. 1b - Clock is running.
23 BLOCK23	IP block status This bit provides the clock status of block 23 in partition 2. 0b - Clock is not running. 1b - Clock is running.
22 BLOCK22	IP block status This bit provides the clock status of block 22 in partition 2. 0b - Clock is not running. 1b - Clock is running.
21 BLOCK21	IP block status This bit provides the clock status of block 21 in partition 2. 0b - Clock is not running. 1b - Clock is running.
20 BLOCK20	IP block status This bit provides the clock status of block 20 in partition 2. 0b - Clock is not running. 1b - Clock is running.
19 BLOCK19	IP block status This bit provides the clock status of block 19 in partition 2. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 BLOCK18	IP block status This bit provides the clock status of block 18 in partition 2. 0b - Clock is not running. 1b - Clock is running.
17 BLOCK17	IP block status This bit provides the clock status of block 17 in partition 2. 0b - Clock is not running. 1b - Clock is running.
16 BLOCK16	IP block status This bit provides the clock status of block 16 in partition 2. 0b - Clock is not running. 1b - Clock is running.
15 BLOCK15	IP block status This bit provides the clock status of block 15 in partition 2. 0b - Clock is not running. 1b - Clock is running.
14 BLOCK14	IP block status This bit provides the clock status of block 14 in partition 2. 0b - Clock is not running. 1b - Clock is running.
13 BLOCK13	IP block status This bit provides the clock status of block 13 in partition 2. 0b - Clock is not running. 1b - Clock is running.
12 BLOCK12	IP block status This bit provides the clock status of block 12 in partition 2. 0b - Clock is not running. 1b - Clock is running.
11 BLOCK11	IP block status This bit provides the clock status of block 11 in partition 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
10 BLOCK10	IP block status This bit provides the clock status of block 10 in partition 2. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK9	IP block status This bit provides the clock status of block 9 in partition 2. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK8	IP block status This bit provides the clock status of block 8 in partition 2. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK7	IP block status This bit provides the clock status of block 7 in partition 2. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK6	IP block status This bit provides the clock status of block 6 in partition 2. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK5	IP block status This bit provides the clock status of block 5 in partition 2. 0b - Clock is not running. 1b - Clock is running.
4 BLOCK4	IP block status This bit provides the clock status of block 4 in partition 2. 0b - Clock is not running. 1b - Clock is running.
3	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK3	This bit provides the clock status of block 3 in partition 2. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK2	IP block status This bit provides the clock status of block 2 in partition 2. 0b - Clock is not running. 1b - Clock is running.
1 BLOCK1	IP block status This bit provides the clock status of block 1 in partition 2. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK0	IP block status This bit provides the clock status of block 0 in partition 2. 0b - Clock is not running. 1b - Clock is running.

32.4.51 Partition 2 COFB Set 0 Clock Enable Register (PRTN2_COFB0_CLKEN)

Offset

Register	Offset
PRTN2_COFB0_CLKEN	530h

Function

This register provides clock control signaling to the individual COFBs in set 0 inside partition 2. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Preliminary under NDA only
COMPANY PROPRIETARY
Beijing CHI Automotive Technology Co., Ltd.
c8c4625-421-4888-b051-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	REQ31	REQ30	REQ29	REQ28	REQ27	REQ26	REQ25	REQ24	REQ23	REQ22	REQ21	REQ20	REQ19	REQ18	REQ17	REQ16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	REQ15	REQ14	REQ13	REQ12	REQ11	REQ10	REQ9	REQ8	REQ7	REQ6	REQ5	REQ4	REQ3	REQ2	REQ1	REQ0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 REQ31	Clock enable This bit provides the clock enable control for block 31 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
30 REQ30	Clock enable This bit provides the clock enable control for block 30 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
29 REQ29	Clock enable This bit provides the clock enable control for block 29 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
28 REQ28	Clock enable This bit provides the clock enable control for block 28 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
27 REQ27	Clock enable This bit provides the clock enable control for block 27 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
26 REQ26	Clock enable This bit provides the clock enable control for block 26 in partition 2.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
25 REQ25	Clock enable This bit provides the clock enable control for block 25 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
24 REQ24	Clock enable This bit provides the clock enable control for block 24 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
23 REQ23	Clock enable This bit provides the clock enable control for block 23 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
22 REQ22	Clock enable This bit provides the clock enable control for block 22 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
21 REQ21	Clock enable This bit provides the clock enable control for block 21 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
20 REQ20	Clock enable This bit provides the clock enable control for block 20 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
19 REQ19	Clock enable This bit provides the clock enable control for block 19 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
18	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
REQ18	This bit provides the clock enable control for block 18 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
17 REQ17	Clock enable This bit provides the clock enable control for block 17 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
16 REQ16	Clock enable This bit provides the clock enable control for block 16 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
15 REQ15	Clock enable This bit provides the clock enable control for block 15 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
14 REQ14	Clock enable This bit provides the clock enable control for block 14 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
13 REQ13	Clock enable This bit provides the clock enable control for block 13 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
12 REQ12	Clock enable This bit provides the clock enable control for block 12 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
11 REQ11	Clock enable This bit provides the clock enable control for block 11 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 REQ10	Clock enable This bit provides the clock enable control for block 10 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ9	Clock enable This bit provides the clock enable control for block 9 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
8 REQ8	Clock enable This bit provides the clock enable control for block 8 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
7 REQ7	Clock enable This bit provides the clock enable control for block 7 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
6 REQ6	Clock enable This bit provides the clock enable control for block 6 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
5 REQ5	Clock enable This bit provides the clock enable control for block 5 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
4 REQ4	Clock enable This bit provides the clock enable control for block 4 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
3 REQ3	Clock enable This bit provides the clock enable control for block 3 in partition 2.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8462e3_4821_48a8_b251_390d3ab624c1

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
2 REQ2	Clock enable This bit provides the clock enable control for block 2 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
1 REQ1	Clock enable This bit provides the clock enable control for block 1 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.
0 REQ0	Clock enable This bit provides the clock enable control for block 0 in partition 2. 0b - Clock is turned off. 1b - Clock is turned on.

32.4.52 Partition 3 Process Configuration Register (PRTN3_PCONF)

Offset

Register	Offset
PRTN3_PCONF	700h

Function

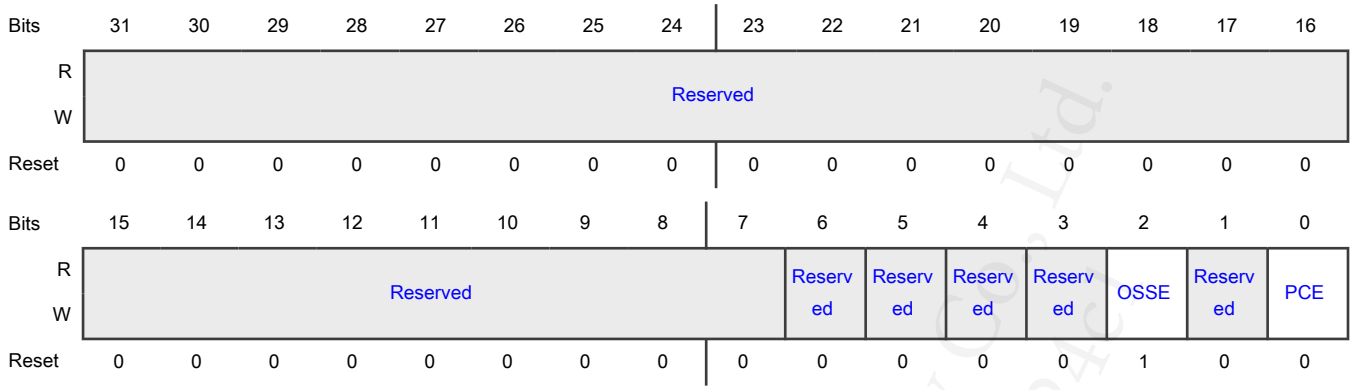
This register provides a configuration for the hardware processes corresponding to partition 3. Each of the configuration bit corresponds to the 'nature' of the processes; for example, enabling/disabling and the trigger is controlled by the corresponding field in the PRTN3_PUPD register. When valid KEY combinations are written onto the CTL_KEY register, the PRTN3_PCONF and PRTN3_PUPD registers are used to determine the hardware processes to be executed. These are triggered in parallel and independent of each other. All dependent processes should be requested one after another from the software.

NOTE

The partition clock enable/disable and output safe stating enable/disable are not standalone and must be done coherently in a fixed sequence. For details, see Software Reset Partition Turn-On Flow Chart and Software reset partition turn-off flowchart in Reset chapter.

Provided under NDA only
COMPANY PROPRIETARY
Beijing Chipsea Automotive Technology Co., Ltd.
c8c4625-...-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 OSSE	Output safe stating enable This bit controls whether the outputs of the partition should be safe stated or not. 0b - Disable output safe stating 1b - Enable output safe stating
1 —	Reserved This field is reserved and read returns zeros.
0 PCE	Partition clock enable This bit controls whether the clock to IPs (other than core(s)) in the partition should be enabled or disabled. 0b - Disable the clock to IPs 1b - Enable the clock to IPs

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cs402e_482f-48a8-b251-390d3ab624c

32.4.53 Partition 3 Process Update Register (PRTN3_PUPD)

Offset

Register	Offset
PRTN3_PUPD	704h

Function

This register provides trigger signaling for the hardware processes corresponding to partition 3. Each of the control bit acts as a trigger for the corresponding hardware processes. When valid KEY combinations are written onto the CTL_KEY register, the hardware checks the bit fields that are programmed as logic-1 in this register, and then triggers the hardware process per the value in the corresponding bit field in the PRTN3_PCONF register. When the hardware process is finished the corresponding bit in this register is auto-cleared to logic-0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								Reserved	Reserved	Reserved	Reserved	OSSUD	Reserved	PCUD	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 OSSUD	Output safe stating update This bit controls whether the hardware processes for enabling/disabling the safe stating for outputs of the partition should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process
1 —	Reserved This field is reserved and read returns zeros.
0 PCUD	Partition clock update This bit controls whether the hardware processes for enabling/disabling the clock to IPs (other than core(s)) in the partition should be triggered or not. 0b - Do not trigger the hardware process 1b - Trigger the hardware process

32.4.54 Partition 3 Status Register (PRTN3_STAT)

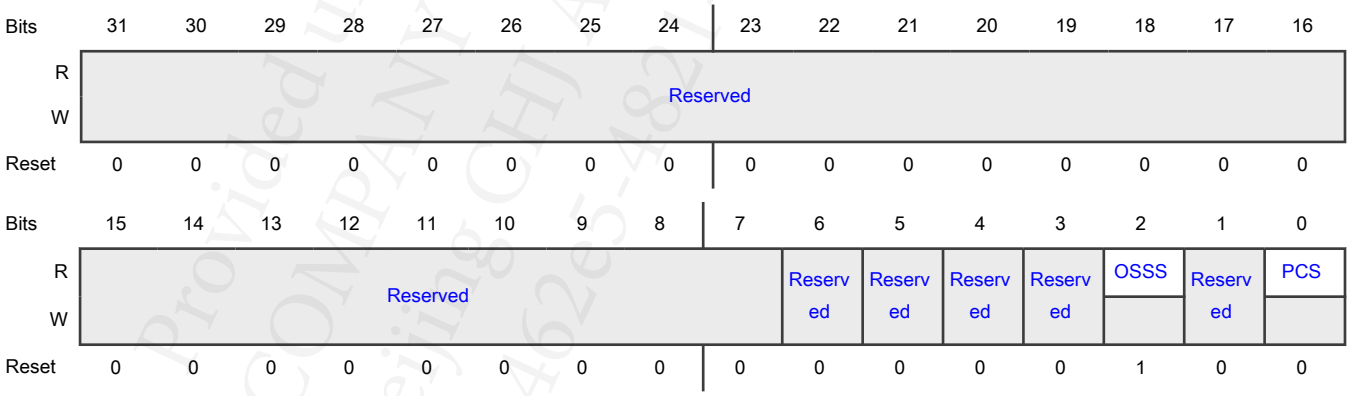
Offset

Register	Offset
PRTN3_STAT	708h

Function

This register provides the current status of the control signals from the partition 3.

Diagram



Fields

Field	Function
31-7 —	Reserved This field is reserved and read returns zeros.
6 —	Reserved This field is reserved and read returns zeros.
5 —	Reserved This field is reserved and read returns zeros.
4 —	Reserved This field is reserved and read returns zeros.
3 —	Reserved This field is reserved and read returns zeros.
2 OSSS	Output safe stating status This bit provides the status for output safe stating of the partition. 0b - Output safe stating is inactive 1b - Output safe stating is active
1 —	Reserved This field is reserved and read returns zeros.
0 PCS	Partition clock status This bit provides the status of the clock to partition. 0b - Clock is inactive 1b - Clock is active

32.4.55 Partition 3 COFB Set 0 Clock Status Register (PRTN3_COFB0_STAT)

Offset

Register	Offset
PRTN3_COFB0_STAT	710h

Function

This register provides the status of set 0 of COFBs inside partition 3.

NOTE

The reset value of this register can vary depending on the availability of active clock pulses inside partition 3.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c162e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BLOCK31	BLOCK30	BLOCK29	BLOCK28	BLOCK27	BLOCK26	BLOCK25	BLOCK24	BLOCK23	BLOCK22	BLOCK21	BLOCK20	BLOCK19	BLOCK18	BLOCK17	BLOCK16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BLOCK15	BLOCK14	BLOCK13	BLOCK12	BLOCK11	BLOCK10	BLOCK9	BLOCK8	BLOCK7	BLOCK6	BLOCK5	BLOCK4	BLOCK3	BLOCK2	BLOCK1	BLOCK0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BLOCK31	IP block status This bit provides the clock status of block 31 in partition 3. 0b - Clock is not running. 1b - Clock is running.
30 BLOCK30	IP block status This bit provides the clock status of block 30 in partition 3. 0b - Clock is not running. 1b - Clock is running.
29 BLOCK29	IP block status This bit provides the clock status of block 29 in partition 3. 0b - Clock is not running. 1b - Clock is running.
28 BLOCK28	IP block status This bit provides the clock status of block 28 in partition 3. 0b - Clock is not running. 1b - Clock is running.
27 BLOCK27	IP block status This bit provides the clock status of block 27 in partition 3. 0b - Clock is not running. 1b - Clock is running.
26	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK26	This bit provides the clock status of block 26 in partition 3. 0b - Clock is not running. 1b - Clock is running.
25 BLOCK25	IP block status This bit provides the clock status of block 25 in partition 3. 0b - Clock is not running. 1b - Clock is running.
24 BLOCK24	IP block status This bit provides the clock status of block 24 in partition 3. 0b - Clock is not running. 1b - Clock is running.
23 BLOCK23	IP block status This bit provides the clock status of block 23 in partition 3. 0b - Clock is not running. 1b - Clock is running.
22 BLOCK22	IP block status This bit provides the clock status of block 22 in partition 3. 0b - Clock is not running. 1b - Clock is running.
21 BLOCK21	IP block status This bit provides the clock status of block 21 in partition 3. 0b - Clock is not running. 1b - Clock is running.
20 BLOCK20	IP block status This bit provides the clock status of block 20 in partition 3. 0b - Clock is not running. 1b - Clock is running.
19 BLOCK19	IP block status This bit provides the clock status of block 19 in partition 3. 0b - Clock is not running. 1b - Clock is running.

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 BLOCK18	IP block status This bit provides the clock status of block 18 in partition 3. 0b - Clock is not running. 1b - Clock is running.
17 BLOCK17	IP block status This bit provides the clock status of block 17 in partition 3. 0b - Clock is not running. 1b - Clock is running.
16 BLOCK16	IP block status This bit provides the clock status of block 16 in partition 3. 0b - Clock is not running. 1b - Clock is running.
15 BLOCK15	IP block status This bit provides the clock status of block 15 in partition 3. 0b - Clock is not running. 1b - Clock is running.
14 BLOCK14	IP block status This bit provides the clock status of block 14 in partition 3. 0b - Clock is not running. 1b - Clock is running.
13 BLOCK13	IP block status This bit provides the clock status of block 13 in partition 3. 0b - Clock is not running. 1b - Clock is running.
12 BLOCK12	IP block status This bit provides the clock status of block 12 in partition 3. 0b - Clock is not running. 1b - Clock is running.
11 BLOCK11	IP block status This bit provides the clock status of block 11 in partition 3.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is not running. 1b - Clock is running.
10 BLOCK10	IP block status This bit provides the clock status of block 10 in partition 3. 0b - Clock is not running. 1b - Clock is running.
9 BLOCK9	IP block status This bit provides the clock status of block 9 in partition 3. 0b - Clock is not running. 1b - Clock is running.
8 BLOCK8	IP block status This bit provides the clock status of block 8 in partition 3. 0b - Clock is not running. 1b - Clock is running.
7 BLOCK7	IP block status This bit provides the clock status of block 7 in partition 3. 0b - Clock is not running. 1b - Clock is running.
6 BLOCK6	IP block status This bit provides the clock status of block 6 in partition 3. 0b - Clock is not running. 1b - Clock is running.
5 BLOCK5	IP block status This bit provides the clock status of block 5 in partition 3. 0b - Clock is not running. 1b - Clock is running.
4 BLOCK4	IP block status This bit provides the clock status of block 4 in partition 3. 0b - Clock is not running. 1b - Clock is running.
3	IP block status

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLOCK3	This bit provides the clock status of block 3 in partition 3. 0b - Clock is not running. 1b - Clock is running.
2 BLOCK2	IP block status This bit provides the clock status of block 2 in partition 3. 0b - Clock is not running. 1b - Clock is running.
1 BLOCK1	IP block status This bit provides the clock status of block 1 in partition 3. 0b - Clock is not running. 1b - Clock is running.
0 BLOCK0	IP block status This bit provides the clock status of block 0 in partition 3. 0b - Clock is not running. 1b - Clock is running.

32.4.56 Partition 3 COFB Set 0 Clock Enable Register (PRTN3_COFB0_CLKEN)

Offset

Register	Offset
PRTN3_COFB0_CLKEN	730h

Function

This register provides clock control signaling to the individual COFBs in set 0 inside partition 3. Whenever a partition clock enable (non-core) hardware process is initiated, the value of logic-1 in the corresponding bit locations of this register enables the clock to the corresponding block in the partition.

NOTE

The reset value of this register is not defined and is as per the availability of the clock source. See Chip-specific MC_ME information for clock source availability.

Preliminary under NDA only
COMPANY PROPRIETARY
Beijing CHI Automotive Technology Co., Ltd.
c8c4625-421-4888-b051-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REQ31	REQ30	REQ29	REQ28	REQ27	REQ26	REQ25	REQ24	REQ23	REQ22	REQ21	REQ20	REQ19	REQ18	REQ17	REQ16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REQ15	REQ14	REQ13	REQ12	REQ11	REQ10	REQ9	REQ8	REQ7	REQ6	REQ5	REQ4	REQ3	REQ2	REQ1	REQ0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 REQ31	Clock enable This bit provides the clock enable control for block 31 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
30 REQ30	Clock enable This bit provides the clock enable control for block 30 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
29 REQ29	Clock enable This bit provides the clock enable control for block 29 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
28 REQ28	Clock enable This bit provides the clock enable control for block 28 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
27 REQ27	Clock enable This bit provides the clock enable control for block 27 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
26 REQ26	Clock enable This bit provides the clock enable control for block 26 in partition 3.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
25 REQ25	Clock enable This bit provides the clock enable control for block 25 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
24 REQ24	Clock enable This bit provides the clock enable control for block 24 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
23 REQ23	Clock enable This bit provides the clock enable control for block 23 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
22 REQ22	Clock enable This bit provides the clock enable control for block 22 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
21 REQ21	Clock enable This bit provides the clock enable control for block 21 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
20 REQ20	Clock enable This bit provides the clock enable control for block 20 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
19 REQ19	Clock enable This bit provides the clock enable control for block 19 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
18	Clock enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
REQ18	This bit provides the clock enable control for block 18 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
17 REQ17	Clock enable This bit provides the clock enable control for block 17 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
16 REQ16	Clock enable This bit provides the clock enable control for block 16 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
15 REQ15	Clock enable This bit provides the clock enable control for block 15 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
14 REQ14	Clock enable This bit provides the clock enable control for block 14 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
13 REQ13	Clock enable This bit provides the clock enable control for block 13 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
12 REQ12	Clock enable This bit provides the clock enable control for block 12 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
11 REQ11	Clock enable This bit provides the clock enable control for block 11 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 REQ10	Clock enable This bit provides the clock enable control for block 10 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
9 REQ9	Clock enable This bit provides the clock enable control for block 9 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
8 REQ8	Clock enable This bit provides the clock enable control for block 8 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
7 REQ7	Clock enable This bit provides the clock enable control for block 7 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
6 REQ6	Clock enable This bit provides the clock enable control for block 6 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
5 REQ5	Clock enable This bit provides the clock enable control for block 5 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
4 REQ4	Clock enable This bit provides the clock enable control for block 4 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
3 REQ3	Clock enable This bit provides the clock enable control for block 3 in partition 3.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8462e5_4821_48a8_b251_390d3ab624c1

Table continued from the previous page...

Field	Function
	0b - Clock is turned off. 1b - Clock is turned on.
2 REQ2	Clock enable This bit provides the clock enable control for block 2 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
1 REQ1	Clock enable This bit provides the clock enable control for block 1 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.
0 REQ0	Clock enable This bit provides the clock enable control for block 0 in partition 3. 0b - Clock is turned off. 1b - Clock is turned on.

32.5 Partition processes

Each of the processes inside the partition controls register space and corresponds to a control signal provided to that partition. A partition can include a core, or COFBs, or both. The MC_ME hardware processes provide control and status via signals provided to partitions. Each partition can be assigned a signal for control and a signal for status. Each of the control signals implements functionality for the partition. For example, clock gating and peripheral control.

The hardware process can be triggered and monitored using a set of three registers:

- Configuration register; for example, Partition n Process Configuration register
- Update register; for example, Partition n Process Update register
- Status register; for example, Partition n Status register

Similar registers exist for cores inside the partition.

The process setup and triggering procedure is shown in Figure 176. Each of the processes is independent of others and can be triggered or retriggered in parallel or sequential to other processes. The triggering or retriggering mechanism remains the same.

Provided by IHS DA only
COMPANY PROPRIETARY
Beijing CEC Huada Electronic Technology Co., Ltd.
c8c462e5-8214-8a5-251-390d3ab624c1

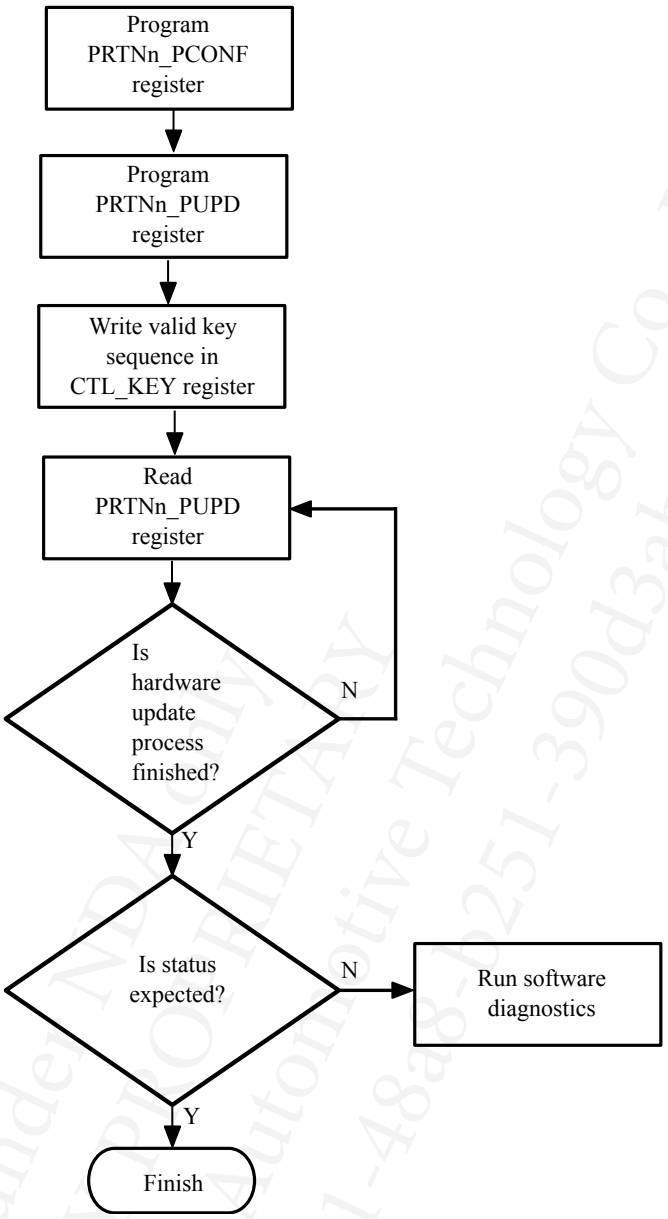


Figure 176. Partition process setup procedure

All the hardware processes are bound to finish in 256 cycles of the MC_ME configuration clock. If the actual and the expected status for a process does not match, then the diagnostics is left as a software responsibility. The software diagnostic can include further wait cycles for the status to match.

32.6 Mode transition

MC_ME implements a mode transition mechanism, whereby the mode of operation for SoC can be changed. Then module implements a mechanism that can lead to:

- Destructive reset
- Functional reset
- Standby mode entry

Destructive reset and functional reset requests from MC_ME are non-retractable transitions. After it is initiated, the other MC_ME functionality is rendered unusable and bus errors are provided for upcoming access to the MC_ME register until a reset sequence is executed by MC_RGM. Hence, it is vital that MC_RGM should never ignore or gate the reset requests from MC_ME.

For transition into the standby mode, the software should ensure that required IP blocks such as clock sources and I/O communication are in their respective inactive states before initiating a standby mode transition to MC_ME. After MC_ME initiates a power down sequence request, it cannot be retracted. The SoC enters a standby power down sequence and then reenter power-up sequence even for cases where the standby wakeup happens right at the time of initiating a power-down sequence request.

Steps for initiating the MC_ME mode transition:

1. Setup the MODE_CONF register with the corresponding target mode bit set to logic-1.
2. Perform the same update as done in the MODE_CONF register on the CONF_UPD register.
3. Write the valid control key (0x5AF0) on the CTL_KEY register.
4. Write the valid invert control key (0xA50F) on the CTL_KEY register.

Mode transition to MC_ME is initiated, after the sequence mentioned above is completed.

In step 1, if both FUNC_RST and DEST_RST in Mode Configuration Register (MODE_CONF) are 1:

- After step 4 is complete, MC_ME initiates a mode transition to a destructive (not functional) reset.
- After the chip exits reset, MC_RGM records that both MC_ME's destructive reset and MC_ME's functional reset were the reset source.

NOTE

Any hardware partition processes setup, along with mode transition, is executed in parallel to the mode transition of MC_ME.

32.7 Standby entry

MC_ME provides hardware processes that implement shutdown sequencing of on-chip resources, such as cores and COFBs. The standby entry sequencing can be achieved or implemented using these hardware processes. The order of the hardware process is determined by the software and MC_ME. It requires no restriction in sequencing of the operation. Following is an example sequence for initiating a power-down sequence for entering the standby mode for SoC. The standby entry sequence should include (but not limited to) the following steps:

1. Setting up wakeup lines
2. Shutting down cores and COFBs
3. Switching all MC_CGM muxes to clk_src_0 with PCFS
4. Powering down all clock sources except clk_src_0
5. Setting up MC_ME using the main core and initiating a standby mode transition
6. Executing WFI instruction on the main core (per Arm specification)

32.7.1 Application core shutdown

This section describes a mechanism for shutting down an application core. The sequence proposed here is extendible with the housekeeping tasks required for other IPs. Each of the tasks mentioned in the following sequence, can be further integrated with an SoC-specific task.

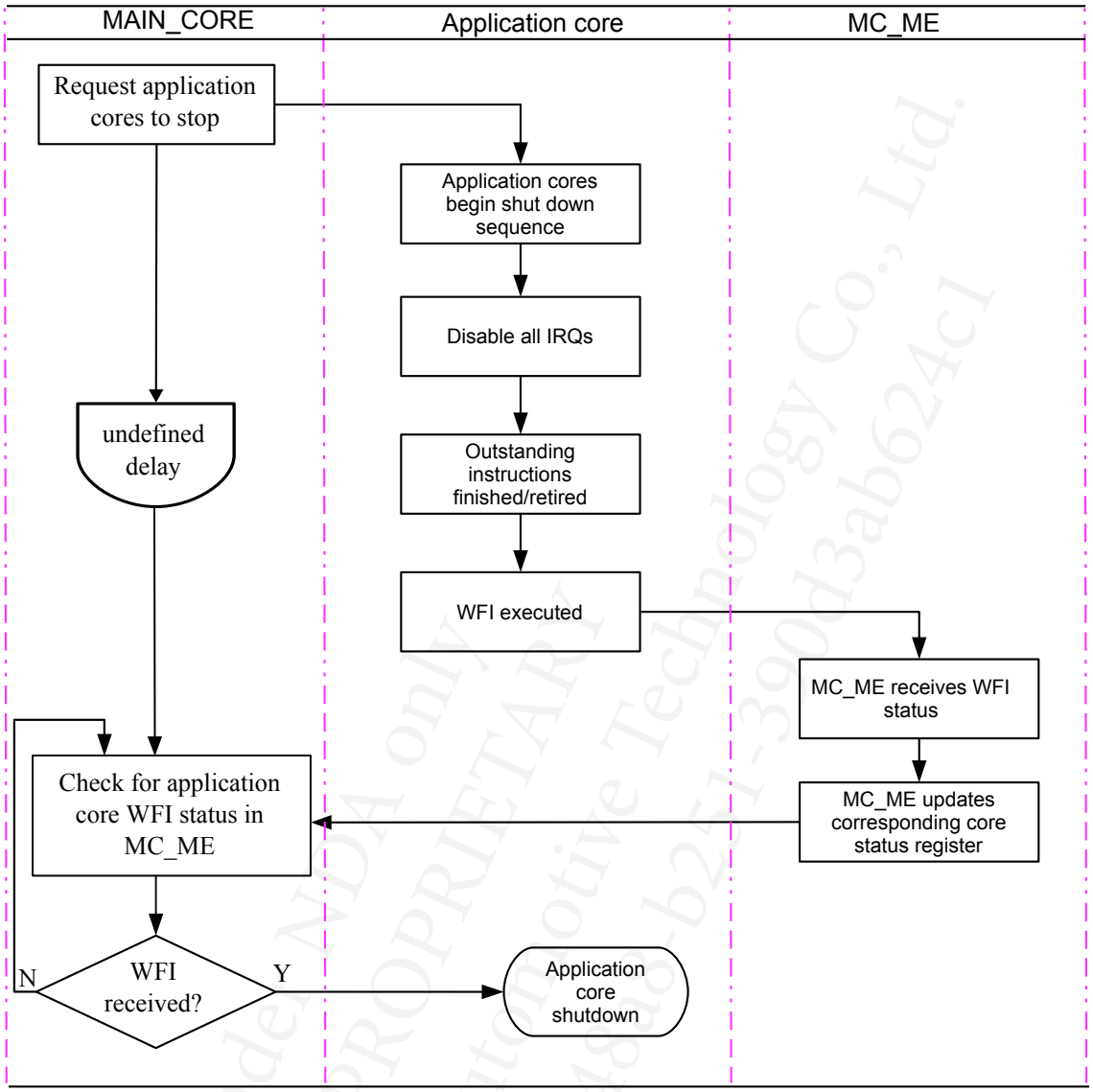


Figure 177. Application core shutdown

After the application core is shutdown, the main core can optionally decide to gate the respective core clock using the corresponding core clock hardware process.

32.7.2 Application cluster shutdown

This section describes a mechanism for shutting down an application cluster. The sequence proposed here is extendible with the housekeeping tasks required for other IPs. Each of the tasks mentioned in the following sequence can be further integrated with an SoC- specific task and can also be skipped when not required, as shown in Figure 178.

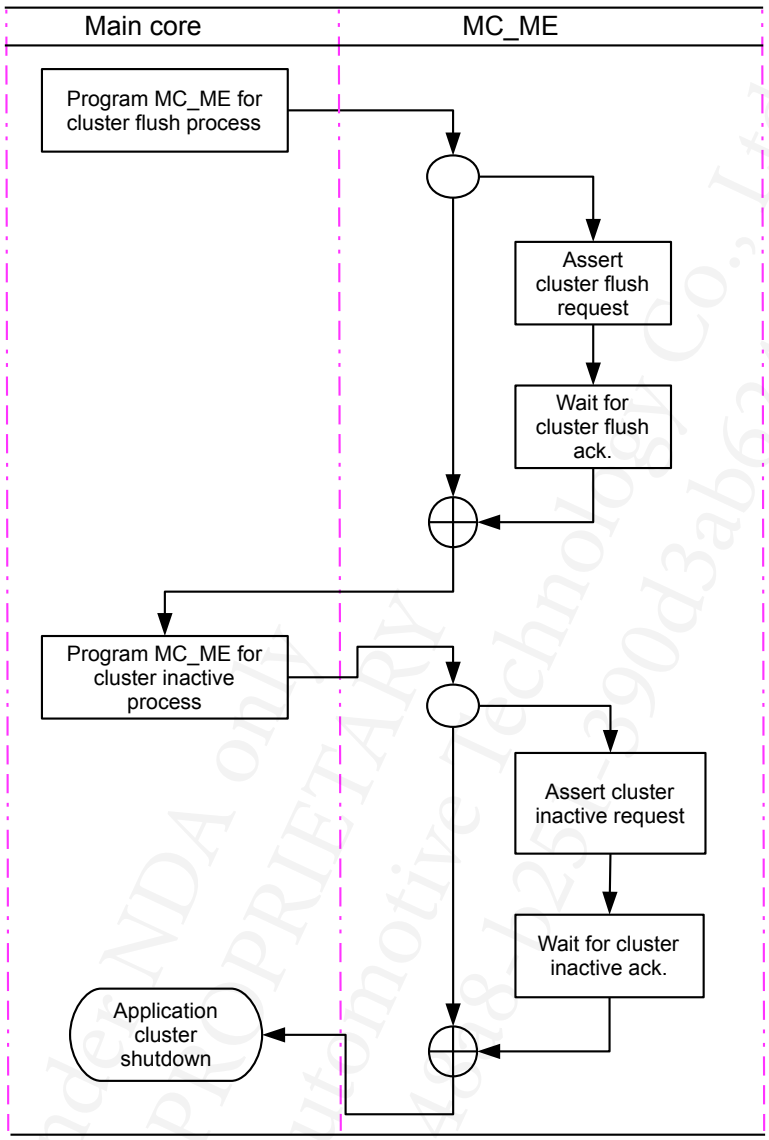


Figure 178. Application cluster shutdown

After the application cluster is shutdown, the main core can optionally decide to gate the respective core clock using the corresponding core clock hardware process.

32.7.3 Main core shutdown and standby entry

This section describes standby entry sequence along with the main core shutdown. This sequence should only be initiated after SoC is ready for entering standby and has completed all the housekeeping activities. It is necessary that the main core has completed all the operations pertaining to other (application cores) and is the last active core before initiating the standby entry sequence. See Figure 179.

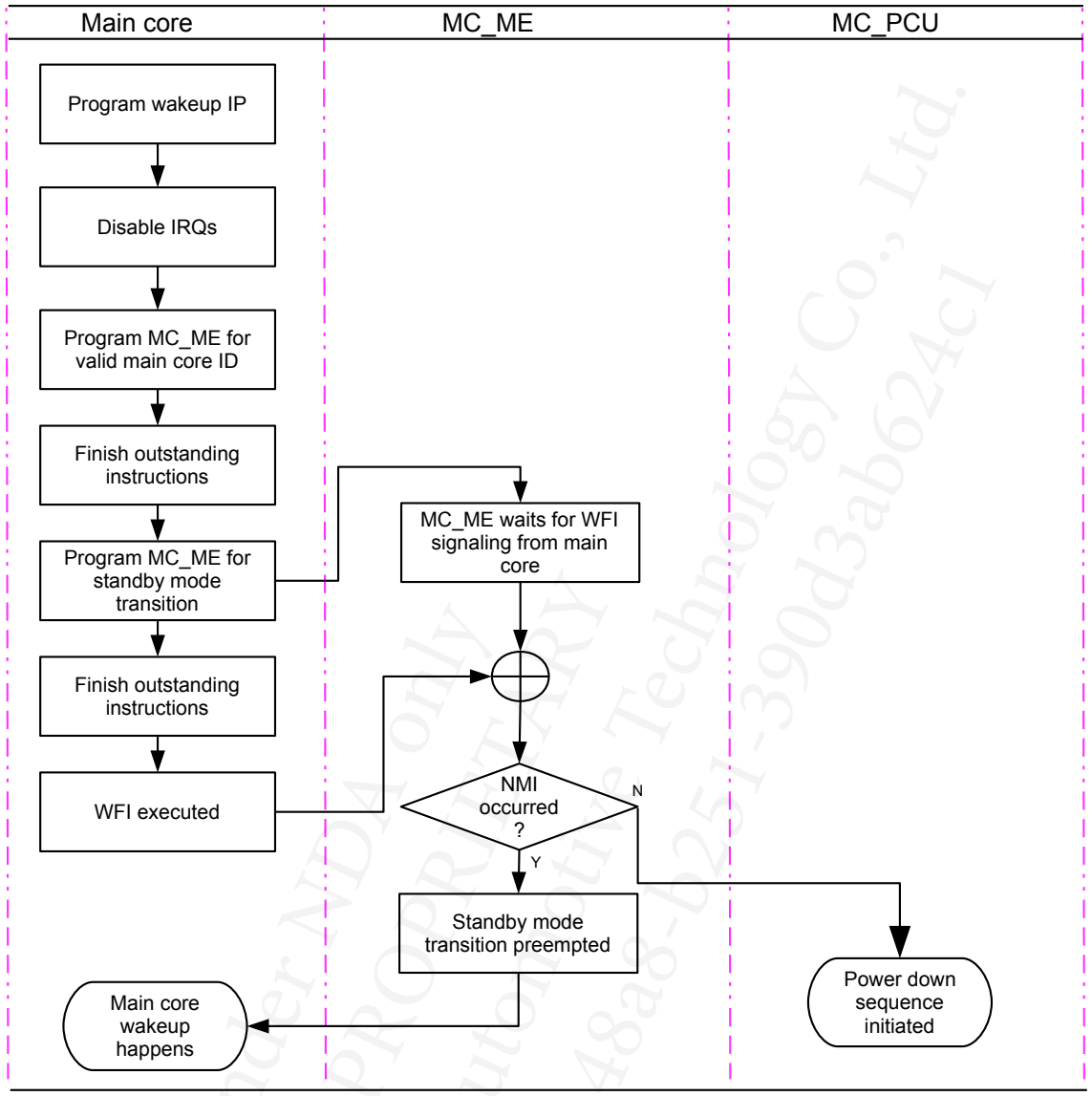


Figure 179. Standby entry sequence along with main core shutdown

NOTE

- After MC_ME initiates the power sequence to MC_PCU, it also gates the NMI signal to the main core with its clock still running. This enables the main core to remain inactive (WFI state) until it is reset and power-up again at standby exit. Therefore, it is necessary that gated NMI signal to the core should not be lost by the wakeup unit.

32.8 Glossary

WFI	Wait for interrupt
COFB	Collection of functional blocks also referred as number of peripherals

Provided under NDA only
COMPANY PROPRIETARY
Beijing HJ Automotive Technology Co., Ltd.
c8462e51-0214-48a8-b251-390d3ab624c1

Chapter 33
Power Control Unit (MC_PCU)

33.1 Introduction

The power control unit (MC_PCU) is used for initiating a Standby mode entry that reduces the overall chip power consumption. Power can be saved by disconnecting parts of the chip from the power supply. The blocks inside the chip are grouped into multiple parts having this capability, which are called "power domains".

When a power domain is disconnected from the supply, the power consumption is reduced to zero in that domain. Any status information, such as, power domain is lost. When you reconnect a power domain to the supply voltage, the domain draws an increased current until the power domain reaches its operational voltage. Maximum power saving is achieved by entering the Standby mode.

After the MC_ME asserts a standby entry request, MC_PCU initiates the power sequence, which is non-retractable and includes the handshake with the chip power management controller. The power-up/down sequences are handled by FSMs to ensure a smooth and safe transition into and out of the Standby mode. Exiting the Standby mode can only be done through a system wakeup event, power-on reset, destructive reset, or a functional reset.

33.2 Power sequence FSM

MC_PCU implements an FSM to initiate the power sequencing of the Standby mode entry/exit sequence for the chip.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3a6

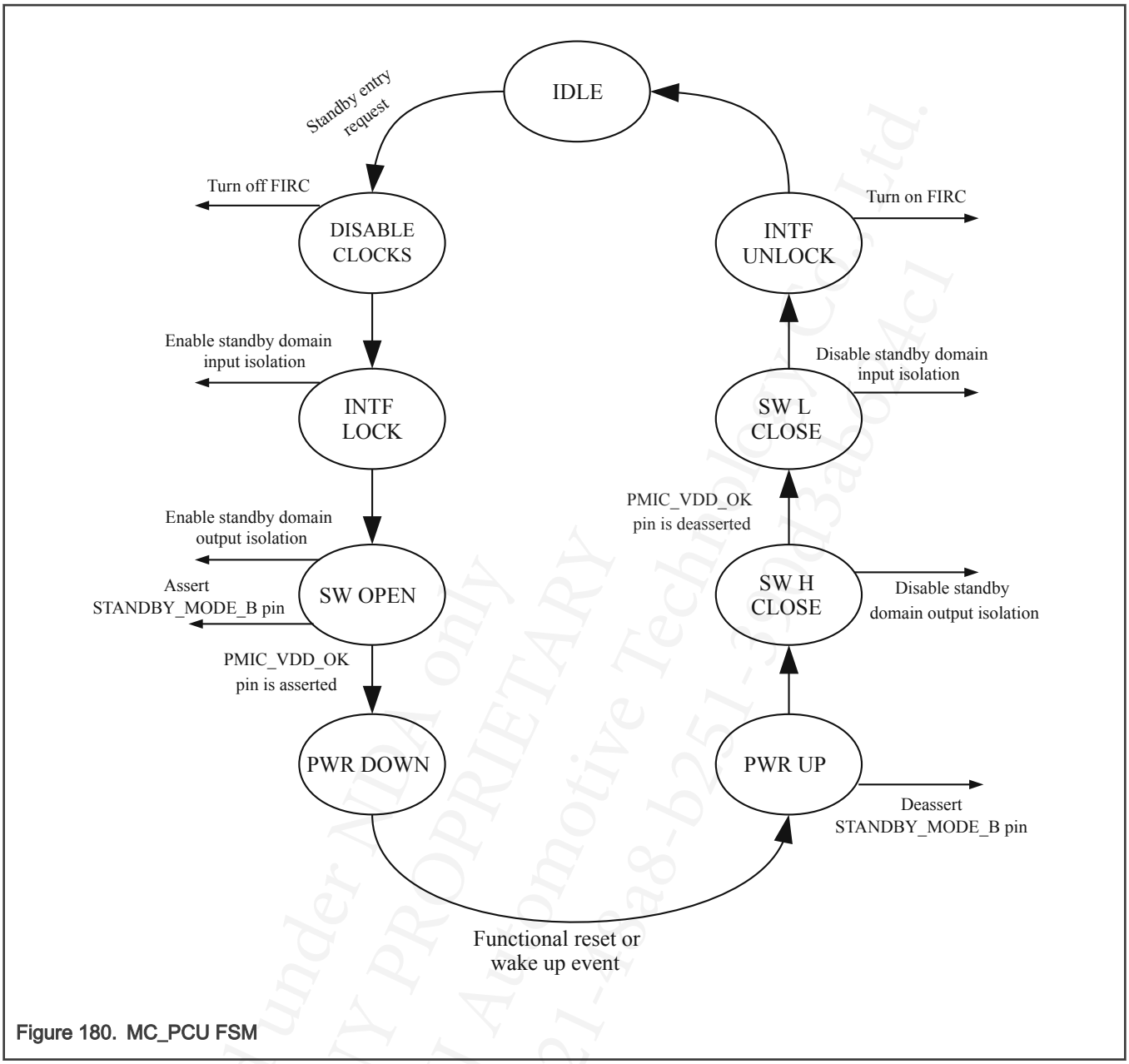


Figure 180. MC_PCU FSM

NOTE

When destructive reset is asserted from MC_RGM, MC_PCU FSM moves to the IDLE state immediately. Indication of this is not shown in the above figure.

Table 193. MC_PCU FSM transition description

State	Name	Exit condition
IDLE	Idle state	Standby entry request received from MC_ME
DISABLE CLOCKS	Disable clock state	Standby entry acknowledgment from MC_RGM and FIRC PD acknowledgment
INTF LOCK	Interface lock state	Input isolation active

Table continues on the next page...

Table 193. MC_PCU FSM transition description (continued)

State	Name	Exit condition
SW OPEN	Switch open state	Output isolation active, PMIC_VDD_OK input pin has been deasserted by the PMIC
PWR DOWN	Power down state	Wakeup or functional reset event occurrence
PWR UP	Power up state	One clock cycle of SIRC
SW H CLOSE	Switch H close state	PMIC_VDD_OK input pin has been asserted by the PMIC
SW L CLOSE	Switch L close state	Output isolation inactive
INTF UNLOCK	Interface unlock	Input isolation inactive

33.3 Glossary

FSM Finite state machine

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 34
Power Management Controller (PMC)

34.1 Chip-specific PMC information

34.1.1 Device Status Flag (DSF*n*) status signal mapping

The following sections provide details about the captured status signals through the PMC device status flag registers.

34.1.1.1 Device Status Flag (DSF0) register status signals

The following table provides details about the captured status signals through the DSF0 (POR_WDOG_EVENT_CAPTURE0) Register.

Table 194. PMC_DSF0[DSFBIT0] signal mapping

Bit field	Bit name	Bit field description
0	RGM_DES_0	Destructive Reset Event Status 0
1	RGM_DES_1	Destructive Reset Event Status 1
2	Reserved	Reserved
3	RGM_DES_3	Destructive Reset Event Status 3
4	RGM_DES_4	Destructive Reset Event Status 4
5	SYS_FUSE_SCAN_DONE	System fuse OCOTP fuse scanning done The description is valid only when POR_WDOG_EVENT_CAPTURE2[bit 19] is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
6	RGM_DES_6	Destructive Reset Event Status 6
7	SIRC_TRIM_LATCH	The description is valid only when POR_WDOG_EVENT_CAPTURE2[bit20] is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
8	RGM_DES_8	Destructive Reset Event Status 8
9	RGM_DES_9	Destructive Reset Event Status 9
10	RGM_DES_10	Destructive Reset Event Status 10
11	RGM_DES_11	Destructive Reset Event Status 11
12	RGM_DES_12	Destructive Reset Event Status 12
13	RGM_DES_13	Destructive Reset Event Status 13
14	PMC_TRIM_LATCH	The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit20]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received

Table continues on the next page...

Table 194. PMC_DSF0[DSFBIT0] signal mapping (continued)

Bit field	Bit name	Bit field description
		0: Ack awaited
15	FIRC_TRIM_LATCH	The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit20]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
16	RGM_DES_16	Destructive Reset Event Status 16
17	RGM_DES_17	Destructive Reset Event Status 17
18	RGM_DES_18	Destructive Reset Event Status 18
19	Reserved	Reserved
20	OSC_PD_ACK	Power down ack from FXOSC
21	CKEN_SLFTST_ACK	Acknowledgment of completion of the sync reset stage for destructive reset modules having sync reset requirement on selftest exit reset sequence.The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit29]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
22:25	Reserved	Reserved
26:29	Reserved	Reserved
30	RGM_DES_30	Destructive Reset Event Status 30
31	RGM_DES_31	Destructive Reset Event Status 31

34.1.1.2 Device Status Flag (DSF1) register status signals

The following table provides details about the captured status signals through the DSF1 (POR_WDOG_EVENT_CAPTURE1) Register.

Table 195. PMC_DSF1[DSFBIT1] signal mapping

Bit field	Bit name	Bit field description
0	RGM_FES_0	Functional Reset Event Status 0
1	SLFTST_ABORT_ACK	FUNC_RST_ENTRY_PROC_DONE[0] - selftest is aborted and cannot be further triggered until the reset sequence completion. The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit22]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence . 1: Ack received 0: Ack awaited
2	PERIPH_NOC_IDLE	IDLE from periph noc

Table continues on the next page...

Table 195. PMC_DSF1[DSFBIT1] signal mapping (continued)

Bit field	Bit name	Bit field description
		The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit23]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
3	RGM_FES_3	Functional Reset Event Status 3
4	RGM_FES_4	Functional Reset Event Status 4
5	Reserved	Reserved
6	RGM_FES_6	Functional Reset Event Status 6
7	OFF_CC_NOC_1_IDLE	IDLE from off-CC NoC_1 The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit23]" is 1 and it is identified as the "stuck-in-reset" stage of Reset sequence (For reset sequence stages, see Reset chapter). 1: Ack received 0: Ack awaited
8	OFF_CC_NOC_2_IDLE	Idle from off-CC NoC_2 The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit23]" is 1 and it is identified as the "stuck-in-reset" stage of Reset sequence (For reset sequence stages, see Reset chapter). 1: Ack received 0: Ack awaited
9	CLK_SW_ACK_CGM_0	Clock switch ack from MC_CGM_0 The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit24]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
10	CLK_SW_ACK_CGM_1	Clock switch ack from MC_CGM_1 The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit24]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
11	CLK_SW_ACK_CGM_5	Clock switch ack from MC_CGM_5 The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit24]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited

Table continues on the next page...

Table 195. PMC_DSF1[DSFBIT1] signal mapping (continued)

Bit field	Bit name	Bit field description
12	CLK_SW_ACK_CGM_2	Clock switch ack from off-CC MC_CGM_2 The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit24]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
13	CLK_SW_ACK_CGM_3	Clock switch ack from off-CC MC_CGM_3 The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit23]" is 1 and it is identified as the "stuck-in-reset" stage of Reset sequence (For reset sequence stages, see Reset chapter). 1: Ack received 0: Ack awaited
14	CLK_SW_ACK_CGM_4	Clock switch ack from off-CC MC_CGM_4 The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit23]" is 1 and it is identified as the "stuck-in-reset" stage of Reset sequence (For reset sequence stages, see Reset chapter). 1: Ack received 0: Ack awaited
15	DIV_INIT_ACK_CGM_0	Divider default configuration ack from MC_CGM_0 The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit25]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
16	DIV_INIT_ACK_CGM_1	Divider default configuration ack from MC_CGM_1 The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit25]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
17	DIV_INIT_ACK_CGM_5	Divider default configuration ack from MC_CGM_5 The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit25]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
18	RGM_FES_18	Functional Reset Event Status 18
19	DIV_INIT_ACK_CGM_2	Divider default configuration ack from off-CC/MC_CGM_2

Table continues on the next page...

Table 195. PMC_DSF1[DSFBIT1] signal mapping (continued)

Bit field	Bit name	Bit field description
		The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit25] is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
20	RGM_FES_20	Functional Reset Event Status 20
21	RGM_FES_21	Functional Reset Event Status 21
22	DIV_INIT_ACK_CGM_3	Divider default configuration ack from off-CC MC_CGM_3 The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit25]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
23	DIV_INIT_ACK_CGM_4	Divider default configuration ack from off-CC MC_CGM_4 The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit25]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence 1: Ack received 0: Ack awaited
24	PD_ACK_PERIPH_DFS	Power down ack from PERIPH_DFS The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit26]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
25	PD_ACK_CORE_DFS	Power down ack from CORE_DFS The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit26]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
26	PD_ACK_CORE_PLL	Power down ack from CORE_PLL The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit26]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
27	PD_ACK_PERIPH_PLL	Power down ack from PERIPH_PLL The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit26]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence.

Table continues on the next page...

Table 195. PMC_DSF1[DSFBIT1] signal mapping (continued)

Bit field	Bit name	Bit field description
		1: Ack received 0: Ack awaited
28	PD_ACK_DDR_PLL	Power down ack from DDR_PLL The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit26]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
29	PD_ACK_ACCEL_PLL	Power down ack from ACCEL_PLL The description is valid only when "POR_WDOG_EVENT_CAPTURE2[bit26]" is 1 and it is identified as the "stuck-in-reset" stage of reset sequence. 1: Ack received 0: Ack awaited
30	RGM_FES_30	Functional Reset Event Status 30
31	RGM_FES_31	Functional Reset Event Status 31

34.1.1.3 Device Status Flag (DSF2) register status signals

The following table provides details about the captured status signals through the DSF2 (POR_WDOG_EVENT_CAPTURE2) Register.

Table 196. PMC_DSF2[DSFBIT2] signal mapping

Bit field	Bit name	Bit field description
0-31	-	Reserved Do not access this register. Accesses do not result in a transfer error.

34.1.2 Supply naming reference

The table below shows the signal names used in this Reference Manual and how they are mapped to the actual names as shown in the IOMUX file attached to this document.

Table 197. Supply naming reference

Supply name in RM	Actual supply name from IOMUX file
Core supply	VDD
PMC supply	VDD_VREF

34.2 Introduction

The PMC monitors the different power domains of the chip. It receives power domain status from different points in the chip and combines them into two outputs:

- Power-on reset event (POR event)
- Non-critical supply presence detector event (NCSPD event)

Use both outputs as a system reset placing an application into a safe state. A low-power mode exists that reduces the power consumption of the PMC by turning off some features.

34.2.1 Features

- CSPD monitoring:
  - Combines the status of all CSPDs into one POR event signal, used as a system reset source
- NCSPD monitoring:
  - Monitors the NCSPD signals and creates a combined NCSPD event based on user configuration.
- POR watchdog (POR_WDOG):
  - Causes a POR event if the counter expires.

34.2.2 Block diagram

Figure 181 shows the functional structure of the PMC.

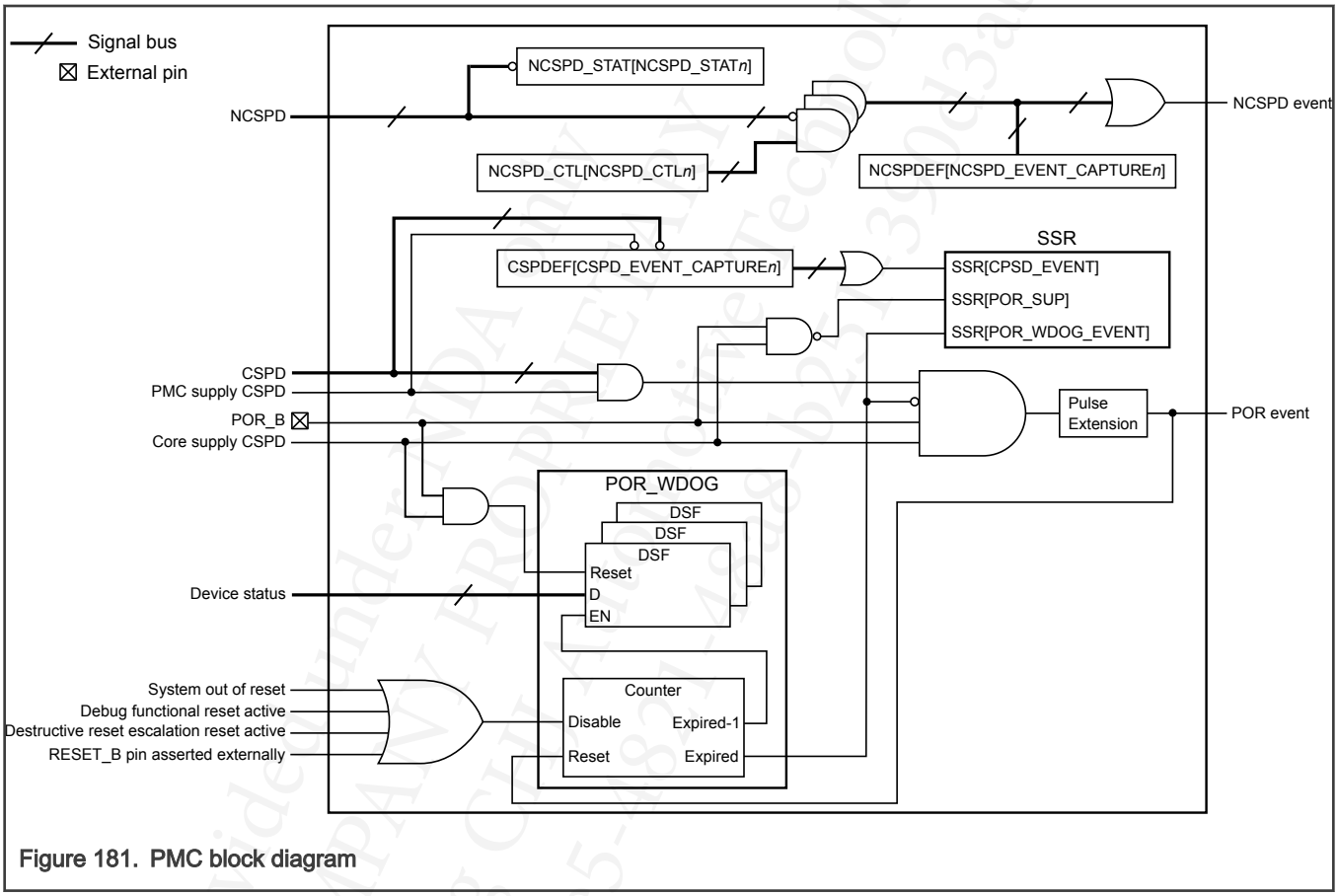


Figure 181. PMC block diagram

Guide to the diagram:

- The following input signals are active low. When those signals are logic high, the supply monitored by its SPD is operating normally.
  - NCSPD
  - CSPD
  - PMC supply CSPD
  - Core CSPD

- The output signal POR event is an active low signal.

The block diagram shows only general functionality. Standard logic, needed for power isolation, is not shown. This logic insures that the PMC supply CSPD signal is logic high when the PMC is in low-power mode.

34.2.2.1 SPD monitoring

There are two SPD types defined as follows:

- NCSPD - Non-critical SPDs are not available after reset and need to be configured before use.
  - Configure which NCSPD you need to use for you application.
  - Configure the NCSPD reaction.
  - Use a status register to show current state of an NCSPD (asynchronous).
  - Event capture register showing any event that caused previous event (synchronous).
- CSPD - Critical SPDs are always on. There is no user configuration available.
  - CSPDs cause a POR event.
  - Event capture register showing any event that caused previous POR (synchronous).

All sources that can modify one of the two output events are determined by reading registers, these register types are:

- Synchronously set registers (status registers) to observe the current state of a signal.
- Asynchronously set registers (event registers) keeping track of any event that has occurred.

The software can read the current state of the signal of interest from the respective field in the status registers, they are only for non-critical events.

Hardware writes one to the event flag register fields (CSPDEF[CSPD_EVENT_CAPTURE*n*] and NCSPDEF[NCSPD_EVENT_CAPTURE*n*]) when their corresponding signals have been in its active state. All signals involved in the generation of events are monitored by such event flag registers.

34.2.2.2 POR and NCSPD event generation

The PMC generates one POR event used as a system reset. The event is extended by the pulse extension so it stays asserted for a specified minimum time (see Data Sheet for details). It is generated by the logical combination of the following sources:

- An external power-on reset input signal (POR_B)
- A CSPD monitoring the Core supply
- A CSPD monitoring the PMC supply
- The combination of all CSPD
- The POR_WDOG counter expires (see POR watchdog for details)

In addition to the CSPD signals, the PMC receives NCSPD signals . When the NCSPD signals are individually enabled by the respective field in the Non-Critical Supply Presence Detector Control (NCSPD_CTL), they are combined into a single NCSPD event .

34.2.3 Modes of operation

The PMC can operate in different power modes:

- Full-power mode with all features available
- Standby mode

The PMC enters Standby mode by removing the PMC supply. The output of the PMC supply CSPD is isolated from the POR event generation during Standby mode. This reset cannot set the POR flag during Standby mode. All other features of the PMC stay functional in all modes.

34.3 Memory map and register definition

34.3.1 Transfer error description

The following actions cause a transfer error, but register contents does not change:

- Accessing an unused register address
- Accessing read-only registers

34.3.2 PMC register descriptions

34.3.2.1 PMC memory map

PMC base address: 4008_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Source of System Reset (SSR)	32	W1C	0000_0001h
4h	Non-Critical Supply Presence Detector Control (NCSPD_CTL)	32	RW	0000_0000h
Ch	NCSPD Status Register (NCSPD_STAT)	32	RO	0000_0000h
10h	CSPD Event Capture (CSPDEF) (CSPD_EVENT_CAPTURE)	32	W1C	See description
14h	NCSPD Event Flag (NCSPDEF) (NCSPD_EVENT_CAPTURE)	32	W1C	See description
1Ch	Device Status Flag (DSF0) (POR_WDOG_EVENT_CAPTURE0)	32	W1C	0000_0000h
20h	Device Status Flag (DSF1) (POR_WDOG_EVENT_CAPTURE1)	32	W1C	0000_0000h
24h	Device Status Flag (DSF2) (POR_WDOG_EVENT_CAPTURE2)	32	W1C	0000_0000h

34.3.2.2 Source of System Reset (SSR)

Offset

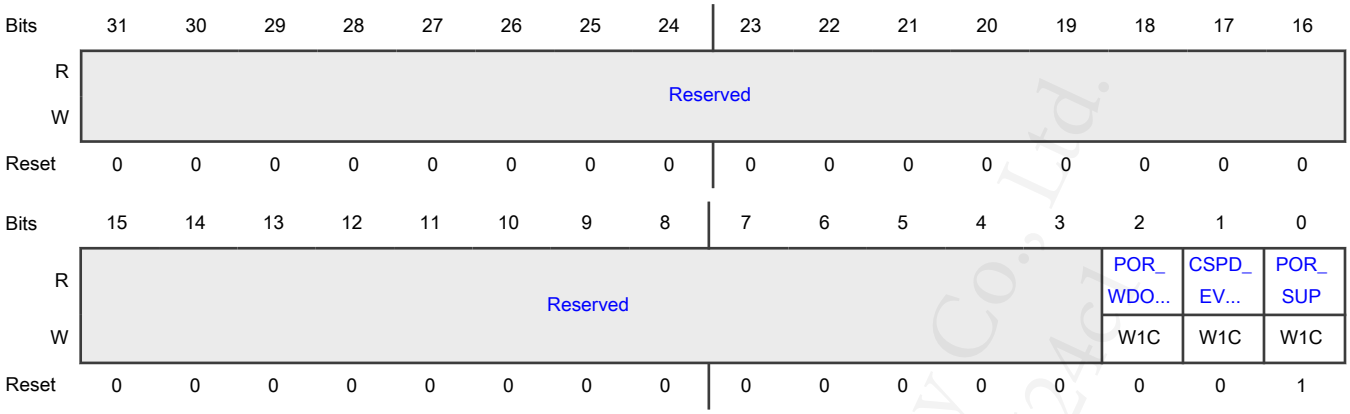
Register	Offset
SSR	0h

Function

The fields capture an event on any of the sources that may set the POR event.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a0-9051-39003e624c1

Diagram



Fields

Field	Function
31-3 —	Reserved
2 POR_WDOG_EVENT	POR watchdog event flag Captures a POR_WDOG timeout event (see the 'Expired' output from the POR_WDOG in Figure 181). 0b - No event detected after the last clearing of the bit 1b - An event was detected
1 CSPD_EVENT	Critical supply presence detector event flag Captures an event on any CSPD failure by monitoring CSPD Event Capture (CSPDEF) (CSPD_EVENT_CAPTURE) (see the CSPD valid and PMC supply CSPD in Figure 181). **NOTE** This field can only be cleared when the CSPDEF register does not show any CSPD event. 0b - No event detected after last clearing of the bit 1b - An event was detected
0 POR_SUP	POR on core supply CSPD event flag Captures an event on the core supply CSPD valid signal or external POR_B . **NOTE** When the value of this field equals 1, you must clear all other PMC registers before they can be used. 0b - No event detected after the last time this field was cleared 1b - An event was detected

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHIPAUTO Automotive Technology Co., Ltd.
c8a02e5-821 4ab b251 390d3ab624c1

34.3.2.3 Non-Critical Supply Presence Detector Control (NCSPD_CTL)

Offset

Register	Offset
NCSPD_CTL	4h

Function

Use these fields to enable capturing NCSPD valid signal status in the NCSPDEF register (see the "NCSPD configuration" section in the "Power Management" chapter for NCSPD mapping).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	NCSPD_C...	NCSPD_C...	NCSPD_C...	NCSPD_C...	NCSPD_C...	NCSPD_C...	NCSPD_C...	NCSPD_C...	NCSPD_C...	NCSPD_C...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
10 —	Reserved
9 NCSPD_CTL9	NCSPD9 control Enables capturing the NCSPD9 valid signal in NCSPDEF[NCSPD_EVENT_CAPTURE9]. 0b - NCSPD9 cannot set NCSPDEF[NCSPD_EVENT_CAPTURE9] 1b - NCSPD9 can set NCSPDEF[NCSPD_EVENT_CAPTURE9]
8 NCSPD_CTL8	NCSPD8 control Enables capturing the NCSPD8 valid signal in NCSPDEF[NCSPD_EVENT_CAPTURE8]. 0b - NCSPD8 cannot set NCSPDEF[NCSPD_EVENT_CAPTURE8] 1b - NCSPD8 can set NCSPDEF[NCSPD_EVENT_CAPTURE8]
7 NCSPD_CTL7	NCSPD7 control Enables capturing the NCSPD7 valid signal in NCSPDEF[NCSPD_EVENT_CAPTURE7]. 0b - NCSPD7 cannot set NCSPDEF[NCSPD_EVENT_CAPTURE7] 1b - NCSPD7 can set NCSPDEF[NCSPD_EVENT_CAPTURE7]
6 NCSPD_CTL6	NCSPD6 control Enables capturing the NCSPD6 valid signal in NCSPDEF[NCSPD_EVENT_CAPTURE6]. 0b - NCSPD6 cannot set NCSPDEF[NCSPD_EVENT_CAPTURE6] 1b - NCSPD6 can set NCSPDEF[NCSPD_EVENT_CAPTURE6]
5 NCSPD_CTL5	NCSPD5 control Enables capturing the NCSPD5 valid signal in NCSPDEF[NCSPD_EVENT_CAPTURE5]. 0b - NCSPD5 cannot set NCSPDEF[NCSPD_EVENT_CAPTURE5] 1b - NCSPD5 can set NCSPDEF[NCSPD_EVENT_CAPTURE5]
4 NCSPD_CTL4	NCSPD4 control Enables capturing the NCSPD4 valid signal in NCSPDEF[NCSPD_EVENT_CAPTURE4]. 0b - NCSPD4 cannot set NCSPDEF[NCSPD_EVENT_CAPTURE4] 1b - NCSPD4 can set NCSPDEF[NCSPD_EVENT_CAPTURE4]
3 NCSPD_CTL3	NCSPD3 control Enables capturing the NCSPD3 valid signal in NCSPDEF[NCSPD_EVENT_CAPTURE3]. 0b - NCSPD3 cannot set NCSPDEF[NCSPD_EVENT_CAPTURE3] 1b - NCSPD3 can set NCSPDEF[NCSPD_EVENT_CAPTURE3]

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 NCSPD_CTL2	NCSPD2 control Enables capturing the NCSPD2 valid signal in NCSPDEF[NCSPD_EVENT_CAPTURE2]. 0b - NCSPD2 cannot set NCSPDEF[NCSPD_EVENT_CAPTURE2] 1b - NCSPD2 can set NCSPDEF[NCSPD_EVENT_CAPTURE2]
1 NCSPD_CTL1	NCSPD1 control Enables capturing the NCSPD1 valid signal in NCSPDEF[NCSPD_EVENT_CAPTURE1]. 0b - NCSPD1 cannot set NCSPDEF[NCSPD_EVENT_CAPTURE1] 1b - NCSPD1 can set NCSPDEF[NCSPD_EVENT_CAPTURE1]
0 NCSPD_CTL0	NCSPD0 control Enables capturing the NCSPD0 valid signal in NCSPDEF[NCSPD_EVENT_CAPTURE0]. 0b - NCSPD0 cannot set NCSPDEF[NCSPD_EVENT_CAPTURE0] 1b - NCSPD0 can set NCSPDEF[NCSPD_EVENT_CAPTURE0]

34.3.2.4 NCSPD Status Register (NCSPD_STAT)

Offset

Register	Offset
NCSPD_STAT	Ch

Function

Captures the current status of an NCSPD.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	NCSPD_S...	NCSPD_S...	NCSPD_S...	NCSPD_S...	NCSPD_S...	NCSPD_S...	NCSPD_S...	NCSPD_S...	NCSPD_S...	NCSPD_S...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEC Automotive Technologies Co., Ltd.
c84e65a2-4621-48a8-b251-3907ab624c1

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 NCSPD_STAT9	NCSPD9 status Shows the current status of the NCSPD9 valid signal. 0b - NCSPD9 has not detected an error condition. 1b - NCPSD9 has detected an error condition.
8 NCSPD_STAT8	NCSPD8 status Shows the current status of the NCSPD8 valid signal. 0b - NCSPD8 has not detected an error condition. 1b - NCPSD8 has detected an error condition.
7 NCSPD_STAT7	NCSPD7 status Shows the current status of the NCSPD7 valid signal. 0b - NCSPD7 has not detected an error condition. 1b - NCPSD7 has detected an error condition.
6 NCSPD_STAT6	NCSPD6 status Shows the current status of the NCSPD6 valid signal.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - NCSPD6 has not detected an error condition. 1b - NCPSD6 has detected an error condition.
5 NCSPD_STAT5	NCSPD5 status Shows the current status of the NCSPD5 valid signal. 0b - NCSPD5 has not detected an error condition. 1b - NCPSD5 has detected an error condition.
4 NCSPD_STAT4	NCSPD4 status Shows the current status of the NCSPD4 valid signal. 0b - NCSPD4 has not detected an error condition. 1b - NCPSD4 has detected an error condition.
3 NCSPD_STAT3	NCSPD3 status Shows the current status of the NCSPD3 valid signal. 0b - NCSPD3 has not detected an error condition. 1b - NCPSD3 has detected an error condition.
2 NCSPD_STAT2	NCSPD2 status Shows the current status of the NCSPD2 valid signal. 0b - NCSPD2 has not detected an error condition. 1b - NCPSD2 has detected an error condition.
1 NCSPD_STAT1	NCSPD1 status Shows the current status of the NCSPD1 valid signal. 0b - NCSPD1 has not detected an error condition. 1b - NCPSD1 has detected an error condition.
0 NCSPD_STAT0	NCSPD0 status Shows the current status of the NCSPD0 valid signal. 0b - NCSPD0 has not detected an error condition. 1b - NCPSD0 has detected an error condition.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-9251-390d3ab624c1

34.3.2.5 CSPD Event Capture (CSPDEF) (CSPD_EVENT_CAPTURE)

Offset

Register	Offset
CSPD_EVENT_CAPTURE	10h

Function

Read these fields to determine if a CSPD monitored supply failed . These fields are unaffected by any reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	CSPD_EV...	CSPD_EV...	HVCPOREF
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 CSPD_EVENT_CAPTURE1	CSPD1 event flag Captures the CSPD1 status. 0b - No CSPD1 error condition detected after last clearing of this field. 1b - CSPD1 error condition detected.
1 CSPD_EVENT_CAPTURE0	CSPD0 event flag Captures the CSPD0 status. 0b - No CSPD0 error condition detected after last clearing of this field. 1b - CSPD0 error condition detected.
0 HVCPOREF	PMC supply CSPD event flag Captures the status of the PMC supply CSPD. 0b - PMC Supply did not cross the lower voltage limit 1b - PMC Supply was below the lower voltage limit

Provided under NDA only
COMPANY PROPRIETARY
Renesas Automotive Technology Co., Ltd.
c5c62e25-482f-48a8-b251-390d3ab624c1

34.3.2.6 NCSPD Event Flag (NCSPDEF) (NCSPD_EVENT_CAPTURE)

Offset

Register	Offset
NCSPD_EVENT_CAPTURE	14h

Function

Captures an NCSPD *n* error if it was enabled by writing one to NCSPD_CTL[NCSPD_CTL *n*]. These fields are unaffected by any reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	NCSPD_E...	NCSPD_E...	NCSPD_E...	NCSPD_E...	NCSPD_E...	NCSPD_E...	NCSPD_E...	NCSPD_E...	NCSPD_E...	NCSPD_E...
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31 —	Reserved
30 —	Reserved
29 —	Reserved
28 —	Reserved
27 —	Reserved
26	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 —	Reserved
10 —	Reserved
9 NCSPD_EVENT_CAPTURE9	NCSPD9 event flag Captures an NCSPD9 error condition. 0b - No NCSPD9 error condition occurred after last clearing of this field. 1b - NCSPD9 error condition occurred.
8 NCSPD_EVENT_CAPTURE8	NCSPD8 event flag Captures an NCSPD8 error condition. 0b - No NCSPD8 error condition occurred after last clearing of this field. 1b - NCSPD8 error condition occurred.
7 NCSPD_EVENT_CAPTURE7	NCSPD7 event flag Captures an NCSPD7 error condition. 0b - No NCSPD7 error condition occurred after last clearing of this field. 1b - NCSPD7 error condition occurred.
6 NCSPD_EVENT_CAPTURE6	NCSPD6 event flag Captures an NCSPD6 error condition. 0b - No NCSPD6 error condition occurred after last clearing of this field. 1b - NCSPD6 error condition occurred.
5 NCSPD_EVENT_CAPTURE5	NCSPD5 event flag Captures an NCSPD5 error condition. 0b - No NCSPD5 error condition occurred after last clearing of this field. 1b - NCSPD5 error condition occurred.
4 NCSPD_EVENT_CAPTURE4	NCSPD4 event flag Captures an NCSPD4 error condition. 0b - No NCSPD4 error condition occurred after last clearing of this field. 1b - NCSPD4 error condition occurred.
3 NCSPD_EVENT_CAPTURE3	NCSPD3 event flag Captures an NCSPD3 error condition.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No NCSPD3 error condition occurred after last clearing of this field. 1b - NCSPD3 error condition occurred.
2 NCSPD_EVENT_CAPTURE2	NCSPD2 event flag Captures an NCSPD2 error condition. 0b - No NCSPD2 error condition occurred after last clearing of this field. 1b - NCSPD2 error condition occurred.
1 NCSPD_EVENT_CAPTURE1	NCSPD1 event flag Captures an NCSPD1 error condition. 0b - No NCSPD1 error condition occurred after last clearing of this field. 1b - NCSPD1 error condition occurred.
0 NCSPD_EVENT_CAPTURE0	NCSPD0 event flag Captures an NCSPD0 error condition. 0b - No NCSPD0 error condition occurred after last clearing of this field. 1b - NCSPD0 error condition occurred.

34.3.2.7 Device Status Flag (DSF0) (POR_WDOG_EVENT_CAPTURE0)

Offset

Register	Offset
POR_WDOG_EVENT_CAPTURE0	1Ch

Function

These fields capture the status of the device just before a POR_WDOG timeout.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DSFBIT0															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DSFBIT0															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e54-8218-48a8-b251-90d3ab624c1

Fields

Field	Function
31-0 DSFBIT0	See the "Chip-specific PMC information" at the beginning of this chapter for the mapping between these fields and the "Device status" signals of the chip (see Figure 181).

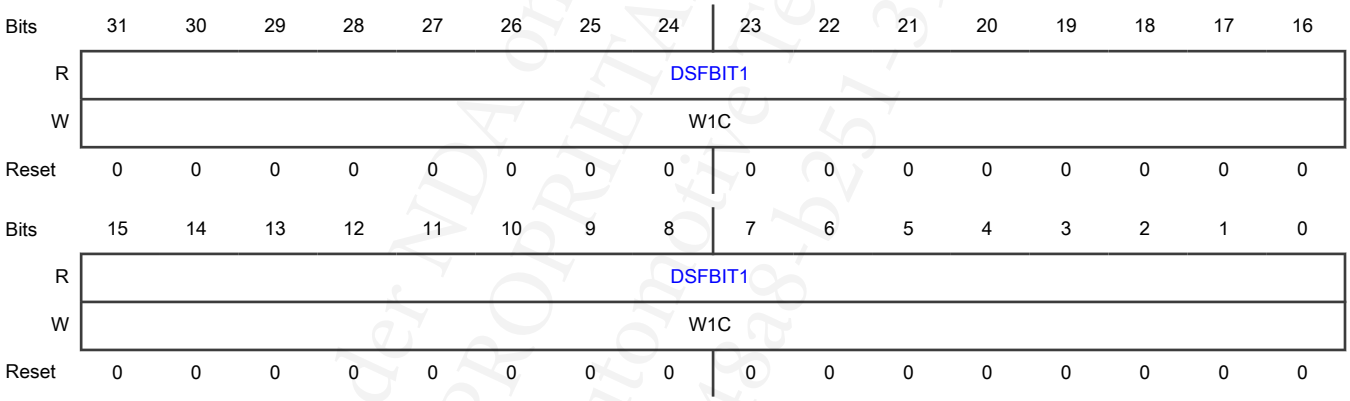
34.3.2.8 Device Status Flag (DSF1) (POR_WDOG_EVENT_CAPTURE1)

Offset

Register	Offset
POR_WDOG_EVENT_CAPTURE1	20h

Function
These fields capture the status of the device just before a POR_WDOG timeout.

Diagram



Fields

Field	Function
31-0 DSFBIT1	See the "Chip-specific PMC information" at the beginning of this chapter for the mapping between these fields and the "Device status" signals of the chip (see Figure 181).

34.3.2.9 Device Status Flag (DSF2) (POR_WDOG_EVENT_CAPTURE2)

Offset

Register	Offset
POR_WDOG_EVENT_CAPTURE2	24h

Function

These fields capture the status of the device just before a POR_WDOG timeout.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DSFBIT2															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DSFBIT2															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DSFBIT2	See the "Chip-specific PMC information" at the beginning of this chapter for the mapping between these fields and the "Device status" signals of the chip (see Figure 181).

34.4 Functional description

34.4.1 Reset

Different kinds of digital logic exist in the PMC reset scheme. Table 198 defines which function asynchronous controls a PMC function.

Table 198. Reset schemes

Asynchronous source	Function impacted	Explanation
POR_B or core supply CSPD (CSPDEF[HVCPOREF])	The entire PMC except for the event flag registers (CSPDEF, NCSPDEF). CSPD and NCSPD signals set CSPDEF and NCPDEF asynchronously. SSR[CSPD_EVENT] equals zero only when CSPDEF is clear.	Reset the logic by power cycling. All other reset sources are active together with or activated by this reset (CSPD and NCSPD signals may not be active with this reset).
Module reset	NCSPD_CTL NCSPD_STAT	Reset these registers with the module reset.
Events on CSPD or NCSPD signal inputs	NCSPDEF CSPDEF SSR[CSPD_EVENT] equals 1	Set the registers asynchronously. Clear the registers by writing a one to each field.
POR_WDOG event	POR_WDOG	Reset the POR_WDOG state machine when a POR_WDOG event occurs. The DSF registers are unaffected by a POR_WDOG event.

34.4.2 Event flag registers

As described in POR and NCSPD event generation, the logical combination of signals from different sources generates the POR and the NCSPD events. Read Source of System Reset (SSR), CSPD Event Capture (CSPDEF) (CSPD_EVENT_CAPTURE), and NCSPD Event Flag (NCSPDEF) (NCSPD_EVENT_CAPTURE) to identify which of the sources activated the event. When an event occurs due to any CSPD signal or any enabled NCSPD signal, the corresponding field in the register equals one until software writes a one.

34.4.3 POR watchdog

The POR watchdog (POR_WDOG) is one of the sources of the POR event. The POR_WDOG is a simple counter. This counter starts immediately when its supply is available and the clock signal is active. When the counter exceeds it non-configurable limit, the POR_WDOG triggers the POR event. To avoid the POR event setting when the system is working correctly, the system must stop the counter before it times-out.

One clock cycle before the counter exceeds its limit ("Expired-1" sets as shown in Figure 181), the device status inputs are captured inside Device Status Flag (DSF0) (POR_WDOG_EVENT_CAPTURE0), Device Status Flag (DSF1) (POR_WDOG_EVENT_CAPTURE1), and POR_WDOG_EVENT_CAPTURE2 registers. This provides some debug capabilities showing why the POR_WDOG counter did not stop in time. Those registers are not reset when the POR_WDOG counter expires.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3a2e2

Chapter 35
SRAM Controller (SRAMC)

35.1 Standby SRAM

The device contains a 32 KB SRAM, called the Standby SRAM. This SRAM is separate from the system SRAM and is powered by the STANDBY power domain. This means it will retain its contents through standby entry and exit unlike the system SRAM. The Standby SRAM is located at the address 0x2400_0000. The RAM is controlled by a single SRAM controller, this is simply another instance of the system ocp2sram controller, but dedicated to the 32KB standby SRAM. The SRAM controller supports ECC protection (single bit correction, double bit detection). Single bit errors are automatically corrected. Double bit errors will cause an FCCU fault to assert. The number of single bit errors that have occurred are counted and are readable in the ERM_STDBY module (a functional reset will clear the count). See the ERM section of the spec for details on that module, the ERM_STDBY base address is 0x4404_7000. Note that the SRAM controller is NOT located in the standby power domain (only the memory array itself is in the STANDBY domain), and thus the controller will not retain it's state during standby mode. The Standby RAM controller's configuration registers are at base address 0x4402_8000.

35.2 SRAMC registers

Here are some important notes about the SRAMC registers. For detailed register descriptions, see the SRAMC register descriptions section.

- The SRAMC register map is not accessible in User Mode.
- Always configure the registers PRAMIAS and PRAMIAE for the selected memory ranges prior to utilization of the initialization feature.

35.3 Introduction

35.3.1 Overview

The SRAMC acts as an interface between the bus request from an interconnect fabric and the integrated system RAM. It converts the protocols between the bus and the dedicated RAM array interface. The primary bus request port provides a connection to the platform module interconnect for direct RAM access from various masters. It uses a simple non-blocking interface, with a pipeline of multiple outstanding transactions, to maximize the bandwidth utilization.

A device can use a single channel or multiple identical channels. Each channel supports one pipeline. The concept is to create a block of SRAM arrays with a channel that can be distributed in the floorplan in an optimal way for each device. This model provides a way to arrange the accesses in an interleaving fashion on a certain address boundary—such as a cache line—so that a line fill or write back occupies only one channel at a time, freeing other channels to respond to other requests.

A global programming model provides centralized control and status, including the initialization setup and error reports, via a common set of memory-mapped registers for all channels.

The following figure shows the block diagram for SRAMC.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4b251-9003abe

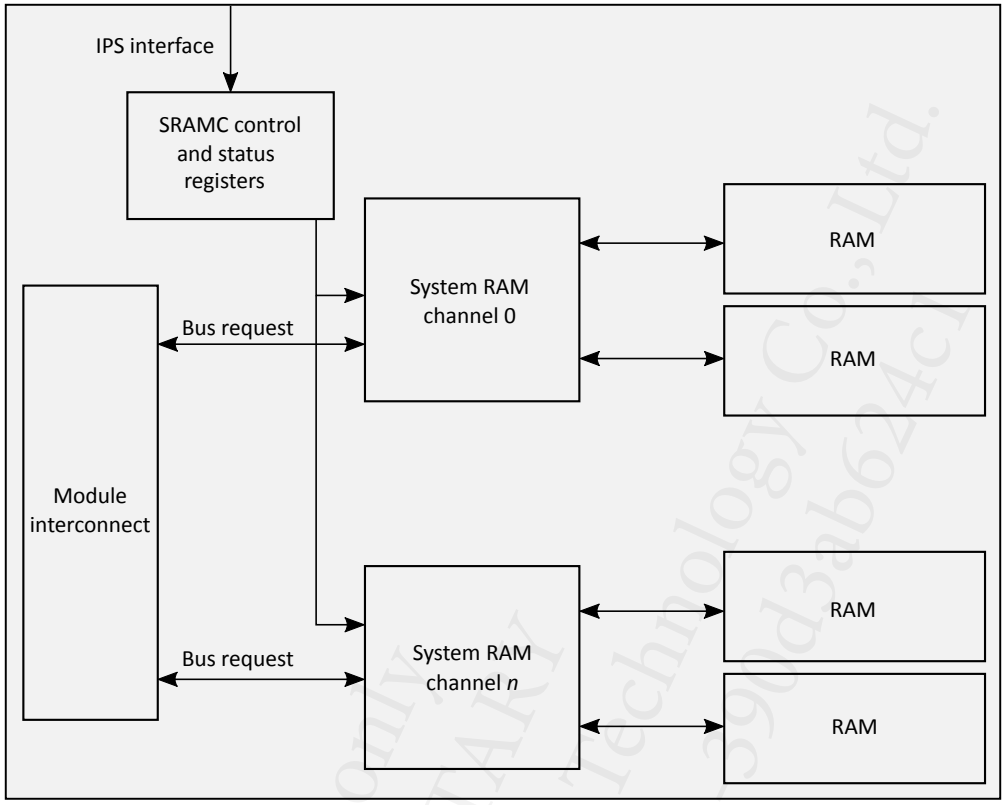


Figure 182. Block diagram

35.3.2 Features

For the RAM size and configuration implemented in this device, see the chip-specific SRAMC information.

Key features of SRAMC include:

- Each channel supports:
  - Single-beat transactions
  - Normal reads
  - Posted writes
  - Multiple outstanding transactions
- Each channel has low access latency for normal reads and posted writes:
  - 4-stage pipeline and a one-transaction latewrite buffer allowing up to four outstanding transactions
  - 3-cycle response for reads
  - 1-cycle response for 64-bit and 128-bit aligned writes
  - 3-cycle response for partial writes with read-modify-write operation for each 64-bit data
- SRAM data is covered by 8-bit ECC on a 64-bit basis using the H-matrix that includes address in the syndrome calculation, which enables single-error correction, double-error detection (SECDED).
- A global IPS interface to 4 K space of memory-mapped control and status registers for all channels
- Support for initialization of the SRAM arrays to a valid ECC state for each entry:
  - Initialization can be triggered by software

  - Initialization sequence with optional wait states, which can be used to limit power surge
  - SRAM arrays are initialized in parallel
  - Software programmable portion of the SRAM for initialization
- Support for an option to block access and suppress any error indicator for that access
- Error injection on memory read data
- Safety logic to detect a fault in the pipeline and the latewrite buffer MBIST detection to yield the array portions under test for MBIST without stalling the normal accesses for the whole MBIST operation

35.3.3 Modes of operation

The current controller only supports normal running mode. It allows MBIST to run concurrently with the normal accesses.

35.4 SRAMC register descriptions

The memory-mapped registers of SRAMC are in an address space of 4 KB and are accessible in Supervisor mode with granularity of word size. Any read or write access to the addresses in the SRAMC memory space not satisfying the above leads to transfer errors. These registers are shared by all SRAMC instances.

35.4.1 SRAMC memory map

SRAMC base address: 4019_C000h

STDBY_SRAM_CFG base address: 4402_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Platform RAM Control Register (PRAMCR)	32	RW	0000_0000h
4h	Platform RAM Initialization Address Register Start (PRAMIAS)	32	RW	0000_0000h
8h	Platform RAM Initialization Address Register End (PRAMIAE)	32	RW	0001_FFFFh
Ch	Platform RAM Status Register (PRAMSR)	32	W1C	0000_0000h
10h	Platform RAM ECC Address (PRAMECCA)	32	RO	0000_0000h

35.4.2 Platform RAM Control Register (PRAMCR)

Offset

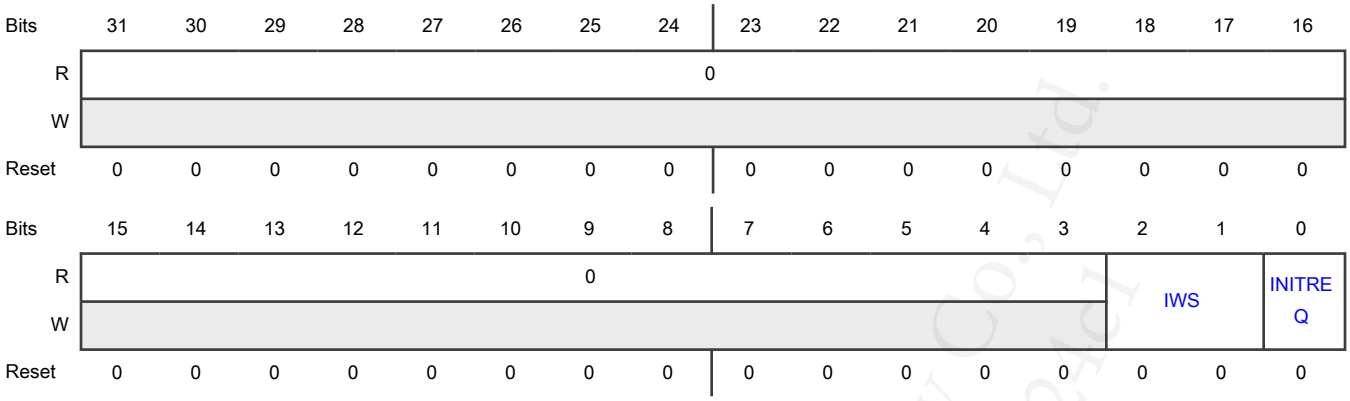
Register	Offset
PRAMCR	0h

Function

Use this register to request an initialization and specify the number of wait cycles for each memory write during the initialization.

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c6225-4821-48ab-b251-390d3a062fc1

Diagram



Fields

Field	Function
31-3 —	Reserved
2-1 IWS	Initialization Wait Cycles Write to this field to specify the number of wait cycles for each memory write during the initialization. The default value of 0b00 enables back to back writes with no idle cycle in between. This feature can be used to reduce power due to several SRAM banks being written to in the same cycle. 00b - No wait cycles between memory writes 01b - One wait cycle between memory writes 10b - Two wait cycles between memory writes 11b - Three wait cycles between memory writes
0 INITREQ	Initialization Request Write 1 to this field to request an initialization of the SRAM for the range specified in the PRAMIAS and PRAMIAE registers. Each time this field is written with a 1, it causes one initialization. To start another initialization, clear the field first before writing 1 to start a new initialization. Hardware clears this bit if the status bit for initialization done (PRAMSR[IDONE]) is cleared by the software. If software writes 0 to this bit prior to completion, the initialization is aborted. 0b - No initialization is requested. 1b - An initialization is requested.

35.4.3 Platform RAM Initialization Address Register Start (PRAMIAS)

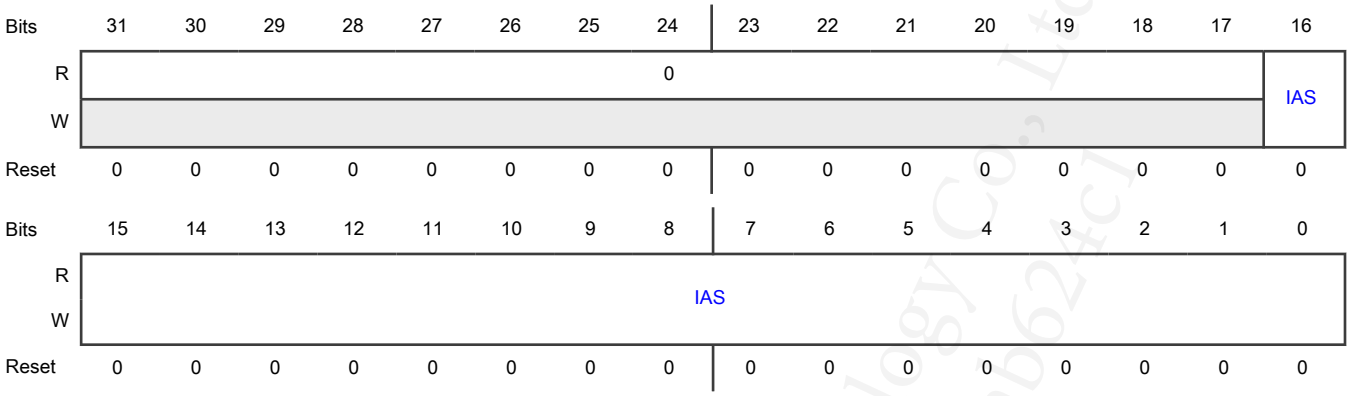
Offset

Register	Offset
PRAMIAS	4h

Function

Use this register to specify the start address for initialization.

Diagram



Fields

Field	Function
31-17 —	Reserved
16-0 IAS	Initialization Start Address Specify the RAM start address for initialization in this field. The address is of the RAM address and not the bus address.

35.4.4 Platform RAM Initialization Address Register End (PRAMIAE)

Offset

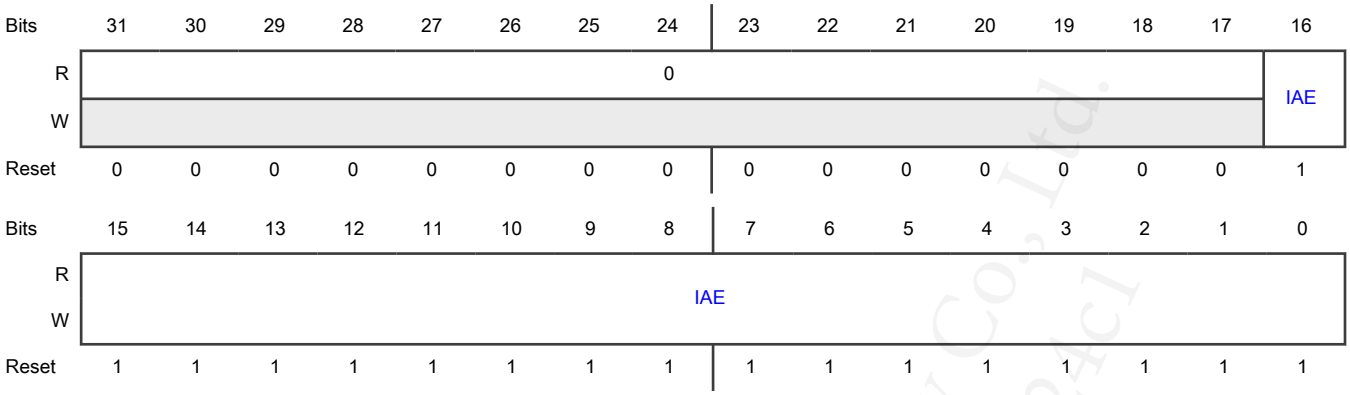
Register	Offset
PRAMIAE	8h

Function

Use this register to specify the end address for initialization. The value of the end address must be greater than that of the start address.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48ad-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-17 —	Reserved
16-0 IAE	Initialization End Address Specify the RAM end address for initialization in this field. The address is of the RAM address and not the bus address. The end address specified here must be greater than the start address specified in PRAMIAS[IAS].

35.4.5 Platform RAM Status Register (PRAMSR)

Offset

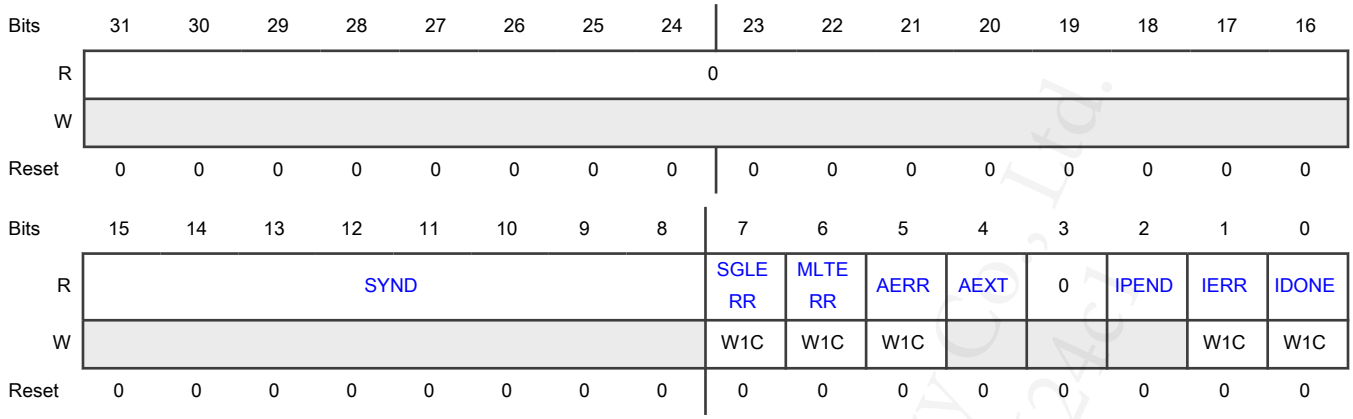
Register	Offset
PRAMSR	Ch

Function

PRAMSR is a global status register that reports the initialization progress as well as different ECC error types for the whole SRAM system.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b281-390d3ab624c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 SYND	ECC Syndrome Value This field is associated with the ECC error in the error fields. After an error is set, this field does not change. The chip updates it with a new value after all the errors are cleared by the software.
7 SGLERR	ECC Single-bit Error Indicates if a single-bit error occurred on a previous access 0b - No single-bit error 1b - A single-bit error
6 MLTERR	ECC Multi-bit Error Indicates if a multi-bit error occurred on a pervious access 0b - No multi-bit error 1b - A multi-bit error
5 AERR	ECC Address Error Indicates an address error on a previous access 0b - No address error 1b - An address error
4 AEXT	Indicates whether address bits used in the ECC HMatrix were from external or local RAM address 0b - Address bits used in the ECC HMatrix were the local RAM address 1b - Address bits used in the ECC HMatrix were from external RAM address
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 IPEND	Initialization Progress Status Indicates whether initialization is in progress The software should not try to clear this bit for it will get undefined behavior. To abort the initialization, it should write to the PRAMCR[INITREQ]. 0b - Initialization is not in progress. 1b - Initialization is in progress.
1 IERR	Initialization Error Indicates whether an error that is not of an ECC error type occurred during initialization or MBIST sequence, or an access outside of the valid range 0b - No non-ECC error occurred on a previous access 1b - A non-ECC error occurred on a previous access
0 IDONE	Initialization Done Indicates whether initialization is complete 0b - Initialization is not complete. 1b - Initialization completed successfully.

35.4.6 Platform RAM ECC Address (PRAMECCA)

Offset

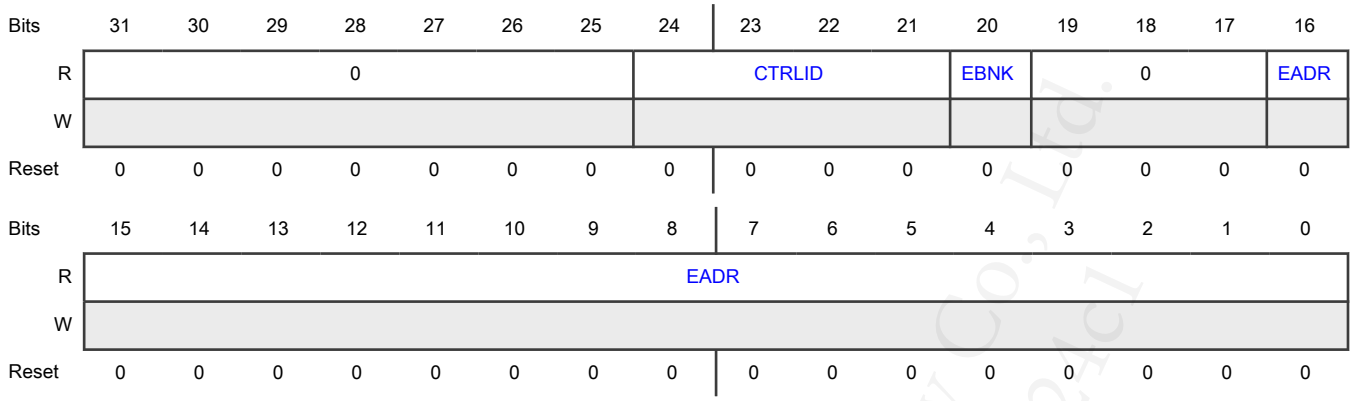
Register	Offset
PRAMECCA	10h

Function

Address associated with an ECC error during initialization. This is a read-only register, and any write to this register leads to transfer error.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4a21-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-25 —	Reserved
24-21 CTRLID	Indicates the controller ID of the reported error. 0000b - The RAM with the error is in controller 0. ... 1111b - The RAM with the error is in controller 15.
20 EBNK	Indicates the RAM bank that has the ECC error indicated in the error fields 0b - Bank 0 has the error 1b - Bank 1 has the error
19-17 —	Reserved
16-0 EADR	The chip writes to report ECC address associated with the error indicated in the error field in PRAMSR. After an error is set, this field does not change. Chip only updates it with a new value if all the errors in the status register PRAMSR have been cleared by the software. Only the software can read this field.

35.5 Functional description

This information describes the functionality of each SRAMC channel.

35.5.1 Latency of read and write accesses

The following tables show pipeline cycles for the read, full-width write, and partial write accesses. This chip implements 3-cycle response for read and partial write. For full-width write, 1-cycle response is achieved because it can avoid read-modify-write operation.

Licensed under NXP CONFIDENTIAL PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e3e5-48a4-48a8-b251-390d3ab624c1

Table 199. Pipeline for bus read with 3-cycle response

Read access pipeline						
	cyc0	cyc1	cyc2	cyc3	cyc4	cyc5
C0: Bus read request	<A>	<B>	<C>	<D>	<E>	
C1: SRAM rdata		<A>	<B>	<C>	<D>	<E>
C3: 3-cycle Resp				<A>	<B>	<C>
SRAM_CE	<A>	<B>	<C>	<D>	<E>	

Table 200. Pipeline for bus full-width write with 1-cycle response

Write access pipeline - full width (No SRAM read is required)						
	cyc0	cyc1	cyc2	cyc3	cyc4	cyc5
C0: Bus write request	<A>	<B>	<C>	<D>	<E>	<F>
C1: 1-cycle Resp		<A>	<B>	<C>	<D>	<E>
C3: 3-cycle Resp				<A>	<B>	<C>
(C2) Latewrite Buffer (lwr)			<A>	<B>	<C>	<D>
SRAM_CE: lwr moves to RAM			lwr A	lwr B	lwr C	lwr D

Table 201. Pipeline for bus partial width write with 3-cycle response

Write access pipeline - partial width (SRAM read is required for RMW)												
	cyc0	cyc1	cyc2	cyc3	cyc4	cyc5	cyc6	cyc7	cyc8	cyc9	cyc10	cyc11
C0: Bus write request	<A>	<B>	<C>	<D>	<E>		<F>		<G>		<H>	
C1: SRAM rdata		<A>	<B>	<C>	<D>		<E>		<F>		<G>	
C3: 3-cycle Resp				<A>	<B>	<C>		<D>		<E>		<F>
Latewrite Buffer				<A>		<B>		<C>		<D>		<E>

Table continues on the next page...

Table 201. Pipeline for bus partial width write with 3-cycle response (continued)

Write access pipeline - partial width (SRAM read is required for RMW)												
	cyc0	cyc1	cyc2	cyc3	cyc4	cyc5	cyc6	cyc7	cyc8	cyc9	cyc10	cyc11
SRAM_CE	<A> rd	<B> rd	<C> rd	<D> rd	Lwr A	<E> rd	Lwr B	<F> rd	Lwr C	<G>rd	Lwr D	<H> rd

35.5.2 Error report

When an ECC error occurs, the SRAMC reports the following: address-bit error and multi-bit error or single-bit error as well as the syndrome bus and the location of the RAM bank. Note that multi-bit error is loosely defined, which is an uncorrectable error including the address-bit error. The information is captured by the PRAMSR register, and a set of error signals are sent as module outputs.

The hardware does not change the error report until the software invalidates it. In case of a new error occurring while the current report for the previous error is still valid, a new report replaces the current report only if the current report is a single-bit error. Address-bit error and multi-bit error in the current report block subsequent errors from being reported.

There will be a single report for each error cycle for all the channel instances. Because only one location is reported, in case of multiple errors, it must select one over the others. The selection is based on the following priority:

1. It looks to see which channel has a highest priority error:
    a. Uncorrectable ECC error: This error type includes address-bit error or double-bit error. This error is not correctable, and is given the highest priority.
    b. Single-bit error: This error has the lowest priority.
2. If multiple channels have the same error types, the priority shifts from channels with lower ID to higher ID values. For a system having 16 channels, channel #0 has the highest priority whereas channel #15 has the lowest priority.
3. Within the selected channel, it selects one of the two RAM banks. If both RAM banks have errors at the same time, the bank with a higher priority error type is selected for reporting. In case of both banks having the same type of error, bank0, which holds lower doubleword (MData[63:0]), is selected for reporting.

The address that is reported is the local RAM address, along with the RAM bank. The information is captured in the PRAMECCA register.

35.6 Safety considerations

35.6.1 ECC algorithm

The ECC scheme implements a SECDED coding using the Hsiao odd-weight column criteria to produce 8-bit ECC checkbits for each SRAM banks:

- 64-bit Data: Bus_Data[63:0] for SRAM bank0
- 29-bit Address of the local SRAM address [16:0] unless specified by chip-specific SRAMC section.

Table 202. ECC HMATRIX

Chec	Data bit[63:32]																													
	Byte 7								Byte 6								Byte 5								Byte 4							
	63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48	47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32

Table continues on the next page...

Table 202. ECC HMATRIX (continued)

kbits [7:0]																															
7	*				*					*			*		*	*		*	*	*	*				*	*			*	*	
6			*	*			*	*					*		*		*				*				*		*	*			*
5	*	*	*	*					*				*	*		*		*	*				*	*	*			*			
4	*		*			*		*					*		*		*			*			*	*			*			*	
3									*	*	*	*	*		*				*		*				*	*	*	*				
2					*		*					*		*			*	*				*	*						*		*
1		*		*		*		*	*	*	*	*						*		*		*		*						*		*
0		*			*	*	*				*			*	*	*		*				*				*		*	*		*	
Check bits [7:0]	Byte 3								Byte 2								Byte 1								Byte 0							
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
7					*		*			*		*					*	*			*	*			*				*	*		*
6	*	*		*	*				*			*					*	*		*	*					*	*				*	*
5			*	*			*	*	*			*	*	*					*		*	*		*								
4	*		*	*				*		*	*				*	*	*			*			*	*				*		*		
3	*	*	*			*						*	*		*	*			*				*		*	*	*			*		
2		*		*	*	*	*	*	*	*	*				*				*				*		*		*	*		*		
1		*						*		*			*	*	*			*				*				*			*	*	*	
0		*		*		*		*				*				*				*				*					*	*		*	*
Check bits	Address bit[31:3]																															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3			

Table continues on the next page...

Table 202. ECC HMATRIX (continued)

[7:0]																															
7		*		*		*		*	*	*		*	*	*		*	*	*		*		*		*	*	*		*				
6		*		*	*	*	*			*	*			*	*	*	*		*		*	*		*	*	*		*				
5		*	*	*			*	*			*	*		*		*		*	*			*	*	*		*	*	*	*			
4	*	*	*		*	*		*		*	*	*	*		*					*	*		*	*	*		*	*				
3	*		*		*	*	*		*		*	*	*			*		*		*	*		*		*	*	*		*			
2	*	*			*		*	*	*	*				*	*	*	*	*	*	*		*	*		*		*		*			
1	*		*	*		*	*		*	*		*	*		*		*	*	*		*	*	*					*	*			
0	*		*	*	*			*	*		*		*	*	*		*		*	*	*			*		*	*		*			

35.6.2 Safety logic

Safety logic was implemented to detect a fault in the pipeline and the latewrite buffer.

- Check the integrity of the address and attributes being transported from the first stage of the pipeline to the last stage. A set of 8-bit parity signals is used, which include address and byte strobes and pipeline hit signals.[end]
- Check the integrity of the write data being transported from the first stage to the latewrite buffer. Sixteen 8-bit parity signals are used to cover two 64-bit data.
- Keep track of the write request from the bus and monitor request from the latewrite buffer to the RAM arrays. Detect any missing or any extra write to the RAM.

35.7 Initialization

The SRAMC can initialize all the RAM banks of all channels that the SRAMC controls. To initialize the banks:

- Program Platform RAM Control Register (PRAMCR), Platform RAM Initialization Address Register Start (PRAMIAS), and Platform RAM Initialization Address Register End (PRAMIAE)

All RAM banks are written in parallel from the lowest address entry to the highest. When the process is complete, a set of status signals is written in Platform RAM Status Register (PRAMSR). This register's fields reflect the status information. The information should be sufficient to determine:

- If the RAM has been initialized successfully
- If the RAM initialization has failed

NOTE

During initialization, access from any master must be prohibited.

Perform the following procedure to initialize the RAM banks:

1. Determine the bus address range to be initialized, and compute the equivalent range in the local memory address. This must identify the lower and upper bound addresses in local memory to set up Platform RAM Initialization Address Register Start (PRAMIAS) and Platform RAM Initialization Address Register End (PRAMIAE).
2. Write the lower bound local memory address to PRAMIAS[IAS].
3. Write the upper bound local memory address to PRAMIAE[IAE].
4. Start initialization by writing 1 to PRAMCR[INITREQ].

35.7.1 Initialization sequence examples

35.7.1.1 Example 1: Configuration of a single pipeline, non-interleaving system, 64-bit data (1 memory bank)

1. Considering the system base address (base_addr) as 4230_0000h and the bus address range to be initialized (bus_addr) as 4230_0040h to 4230_0080h, compute the equivalent range (lower and upper bound addresses) in local memory (mem_addr) by using the following equation:

   local_bus_addr[31:0] = bus_addr - base_addr

   mem_addr[16:0] = local_bus_addr[31:0] > 3

   Calculation result:

   Lower bound: for bus address 4230_0000h, local memory address = 8h

   Upper bound: for bus address 4230_0080h, local memory address = 10h
2. Write the lower bound local memory address to PRAMIAS[IAS].
3. Write the upper bound local memory address to PRAMIAE[IAE].
4. Start initialization by writing 1 to PRAMCR[INITREQ].

The software configured with this sequence writes to the addresses from 8h to 10h of a single RAM bank. It first writes to the address at the lower bound (that is, 8h), and increments it by one write for the next nine cycles, to end at the upper bound address (that is, 10h). That is, the software writes nine times to the RAM bank. For each write, the data is 0 and the address bits used in the ECC is the local memory address.

35.7.1.2 Example 2: Configuration of a single pipeline, non-interleaving system, 128-bit data (2 memory bank)

1. Considering the system base address (base_addr) as 3178_0000h and the bus address range to be initialized (bus_addr) as 3178_0080h to 3178_0110h, compute the equivalent range (lower and upper bound addresses) in local memory (mem_addr) by using the following equation:

   local_bus_addr[31:0] = bus_addr - base_addr

   mem_addr[16:0] = local_bus_addr[31:0] > 4

   Calculation result:

   Lower bound: for bus address 3178_0080h, local memory address = 8h

   Upper bound: for bus address 3178_0110h, local memory address = 11h
2. Write the lower bound local memory address to PRAMIAS[IAS].
3. Write the upper bound local memory address to PRAMIAE[IAE].
4. Start initialization by writing 1 to PRAMCR[INITREQ].

The software configured with this sequence writes to the addresses from 8h to 10h of both the RAM bank. It first writes to the address at the lower bound (that is, 8h), and increments it by one write for the next nine cycles, to end at the upper bound address (that is, 10h). That is, the software writes nine times to each of the two RAM banks (a total of 18 double-words). For each write, the data is 0 and the address bits used in the ECC is the local memory address, resulting in both banks having the same ECC check-bits at each location.

Chapter 36
DDR Subsystem

36.1 Chip-specific DDR information

36.1.1 Feature configuration

The DDR subsystem on this chip does not implement DDR4. Disregard its references in this document.

36.1.2 Mapping of masters and AXI ports of DDR controller

Table 203. Mapping of AXI ports and masters

AXI port	Master
AXI_0	All the non-core masters and HSE : 128 bit data width
AXI_1	All the Cortex-A53s : 128 bit data width
AXI_2	All the Cortex-M7s (except HSE) and DMA : 64 bit data width

36.1.3 Exclusive access monitors

DDR Controller supports eight exclusive access monitors based on the selected configuration. Here is the limitation per AXI port at NoC level:

Table 204. AXI ports and number of monitors

AXI Port 0	Number of monitors
AXI_0	6
AXI_1	3
AXI_2	3

In PCFGR_n register, rdwr_ordered_en field must be set to 1 when the software coherency is required. For details, see "Software coherency for AXI ports" section in DDR Subsystem.

36.1.4 AXI-urgent signals of DDR memory

AXI-urgent signals are disabled in this chip. Ignore any references to the same or its related content.

36.1.5 ECC and parity errors

All the DDR interrupts related to parity, ECC, or CRC errors are connected to FCCU.

36.1.6 Speculative access of DDR memory

Before you initialize DDR, speculative access of DDR memory can generate fault or unexpected behavior. Therefore, you must configure the core programming to disable speculative access of this region.

36.2 Overview

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

The DDR subsystem integrates several submodules to provide a complete memory interface solution for several high-performance memory types.

36.2.1 Supported DRAM types

The DDR subsystem supports the following DRAM types:

- DDR3L
- LPDDR4

36.2.2 Memory controller (MC)

The Memory Controller (MC) peforms the following functions:

- Accepts memory access requests from up to three slave application-side host ports connected to the MC through standard AMBA AXI4 bus interfaces (also know as AXI port interfaces or XPIs).
- Queues requested transactions internally and schedules them for access to the DRAM, satisfying the DRAM protocol timing requirements, transaction priorities, and dependencies between the transactions.
- Uses flexible address mapper logic to allow application-specific mapping of row, column, bank, bank group, and rank bits.

36.2.3 Physical interface (PHY)

The Physical Interface (PHY) launches and captures data to and from the external DRAM. It provides:

- Initialization
- DRAM interface training
- Delay line calibration and VT compensation
- Impedance calibration and VT compensation
- Programmable configuration controls

36.3 Architecture

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

This section contains a system diagram of an example DDR subsystem application, with descriptions of the main architectural components.

36.3.1 Block diagram

The following diagram illustrates the DDR subsystem.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-9251-390d3ab624c2

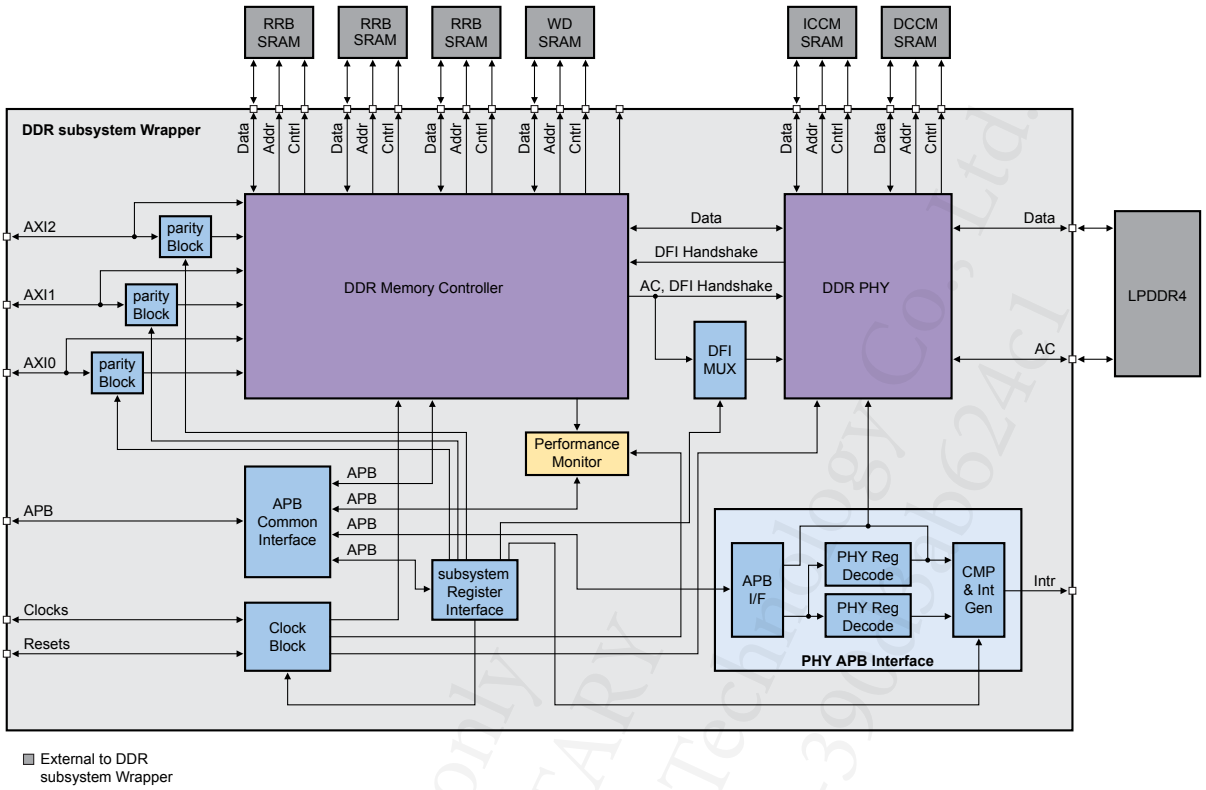


Figure 183. System diagram

36.3.2 Parity block

Each application-side host port (AXI port) connects to the MC through a parity block. Each parity block contains three parity generators to generate one bit of parity either per word or per byte for the AXI write data, write addresses, and read addresses of the corresponding port. OCPARCFG*n* controls parity enabling and parity type for all three parity blocks.

The MC generates parity for the write data, write addresses, and read addresses of each of the three AXI ports.

36.3.3 Performance monitor

The Performance Monitor consists of counter logic driven by various MC performance logging signals.

36.3.3.1 Performance signals/counters

There are no performance counters inside the uMCTL2 controller, so performance signals are brought out to the interface of the uMCTL2 controller within DDR-Subsystem, where a set of 4 x 32 bit counters are implemented. These counters are used to monitor the Performance log specific PERF signals from uMCTL2-Controller based on defined connectivity within DDR-Subsystem. A limited set of Performance log signals are utilized within these Performance monitors.

Monitored signals are:

36.3.3.1.1 Generic monitored signals

Table 205. Generic monitored signals

Signal	Description
WAQ_COUNT_0[3:0]	The number of used positions in the write queue FIFO (queue is 8 deep)
WAQ_POP_0	One write granted by port arbiter
RAQ_WCOUNT_0[3:0]	Number of used positions in read queue - depth is 8
RAQ_POP_0	One read granted by port arbiter
LPR_CREDIT_CNT[6:0]	Number of available low priority read CAM slots free
HPR_CREDIT_CNT[6:0]	Number of available high priority read CAM slots free
WR_CREDIT_CNT[6:0]	Number of available write CAM slots free
stat_ddrc_reg_selfref_type[1:0]	self-refresh status and type • 00 : not in self refresh • 11 : self-refresh by automatic means • 10 : caused by SW/HW interface

36.3.3.1.2 Perf specific signals monitored

These form two groups, selected by the CSV field below [3:0] one bit of which is fed to the counter.

Table 206. Group 0

CSV[3:0]	Signal	Description
0000	perf_hif_rd	asserts for every read command to the DDRC
0001	perf_hif_wr	assert for every write command to the DDRC
0010	perf_hif_rmw	assert for every read modify write to the DDRC
0011	perf_hif_hi_pri_rd	assert for every high priority read
0100	perf_hpr_req_with_nocredit	high performance request when full
0101	perf_hpr_xact_when_critical	high performance request when critical
0110	perf_lpr_req_with_nocredit	low performance request when full
0111	perf_lpr_xact_when_critical	low performance request when critical
1001	perf_wr_xact_when_critical	write request when critical
1010	perf_dfi_rd_data_cycles	assert for every read data cycle
1011	perf_dfi_wr_data_cycles	assert for every write data cycle

Provided under NDA only
COMPANY PROPRIETARY
Beijing Autonomotive Technology Co., Ltd.
c8a0e5e1-148a-b251-3903ab8ac1

Table 207. Group 1

CSV[3:0]	Signal	Description
0000	perf_rdwr_transitions	assert for every read->write or write->read transition
0001	perf_op_is_precharge	assert for every precharge issued
0010	perf_op_is_activate	assert for every activate issued
0011	perf_op_is_load_mode	assert for every MRW or MRR
0100	perf_op_is_mwr	
0101	perf_op_is_rd	
0110	perf_op_is_rd_activate	
0111	perf_op_is_refresh	
1000	perf_op_is_wr	
1001	perf_raw_hazard	RAW or WAW hazard

The actual performance function is performed by a set of counters which can be configured to count a number of the available signals described above. Selection of the value for each counter is done via the config registers. There is one register for each counter.

36.3.3.1.3 Counter control register

31:24 CSV : Count_value_select[7:0] : Used to select which performance signal is to be applied for counting, defined by the table below

23:16 CP : Count_parameter[7:0] : Used to pass a parameter to the counter to limit the count for counterN , see table below

Counter0 is special in that it always counts “time” and when expired causes a lock on itself.
Counter0 start counting, if counter0 is enabled by configuring COUNTER_0_CTRL_REG_ADDR[CNTR_EN]=1 and COUNTER_0_CTRL_REG_ADDR[CNTR_CLR] =1.

Counter0 & COUNTER_0_CTRL_REG_ADDR[CNTR_OFL] is cleared by setting COUNTER_0_CTRL_REG_ADDR[CNTR_CLR] =0.

Logically the counterN start counting if both Counter0 and the corresponding counterN are enabled. It stops counting when either the counterN overflows or counter0 overflows . But this counter functionality also depends on setting of COUNTER_0_CTRL_REG_ADDR[CNTR_CLR] =1, which can mask the enable (CNTR_EN) specific dependency. Therefore, this counterN start incrementing with the value of the “CNTR_INCR_N” selected based on CSV just after setting the COUNTER_0_CTRL_REG_ADDR[CNTR_CLR] =1

CNTR_INCR_N is generated by the following table

CSV[7:0]	Value	comment
8’h00	No increment (zero)	Counter will stop incrementing.
8’h01	stat_ddrc_reg_selfref_type = CP[1:0]	Counts when ram is running vs self refresh
8’h04	RAQ_POP_0 = 1	Number of reads[Not applicable]
8’h05	WAQ_POP_0 = 1	Number of writes

Table continues on the next page...

Table continued from the previous page...

CSV[7:0]	Value	comment
8'h08	RAQ_WCOUNT_0[3:0] = CP[3:0]	Read Queue depth is fixed number X (Number of used positions) [Not applicable]
8'h09	WAQ_WCOUNT_0[3:0] = CP[3:0]	Write Queue depth is fixed number X (Number of used positions)
8'h10	LPR_CREDIT_CNT[6:0] = CP[6:0]	LP read cam has X entries (Number of available CAM free slots)
8'h11	HPR_CREDIT_CNT[6:0] = CP[6:0]	HP read cam has X entries(Number of available CAM free slots)
8'h12	WR_CREDIT_CNT[6:0] = CP[6:0]	write cam has X entries (Number of available CAM free slots)
8'h2X	Group 0 counts from perf signals	See above
8'h3X	Group 1 counts from perf signals	

Note that the intent here is to take samples for the period from resetting the counter0 until it expires again, causing an interrupt, once that happens the values can be read from counterN and then a new set of sample can be taken.

NOTE

We only take EXACT queue depths so we have to repeat to gain a histogram of values.

36.3.3.1.4 MRR read data register

This is a separate read only register which collects status from the controller directly – it's not really related to the perf counters. Its function is

If hif_mrr_data_valid Then

MRR[31:0] := hif_mrr_data[31:0]

Else

MRR[31:0] := MRR[31:0]

In effect it holds the value last supplied from the DRAM controller, it's a simple status read register

This acts as a single register repeated 16 times in the upper part of the perf counter space. These are all pointing to the same register to ensure that the address map is filled up.

36.3.4 APB common interface

This is a custom logic block designed to provide a common APB interface to all internal slaves within the DDR subsystem, thus unifying the register space for software access to DDR subsystem. APB common interface module directs the APB traffic from the single host side APB port to the correct APB interface on either the controller or the PHY.

36.4 Features

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

36.4.1 DDR MC feature list

The MC supports the following features:

- Support for DDR4 protocol up to DDR4-3200 speed grade
- Support for LPDDR4 protocol up to LPDDR4 3200 speed grade
- 1:2 frequency ratio architecture
- Up to three (03) host ports using AMBA AXI4 protocol
  - Support for AXI burst types: Incremental and wrap
  - AXI clock synchronous to the controller clock
  - Exclusive access support
  - Read reorder buffer with reduced latency options (for example, bypass)
- Direct software request control or programmable internal control for ZQ short calibration cycles for DDR3 and DDR4 modes
- Support for ZQ long calibration after self-refresh exit for DDR3 and DDR4 modes.
- Direct software request control or programmable internal control for ZQ calibration cycles for LPDDR4 mode.
- Support for ZQ calibration after SR-Power down exit for LPDDR4 mode.
- Support for ZQ Reset feature through software for LPDDR4 mode.
- Dynamic scheduling to optimize bandwidth and latency
- Read and write buffers in fully associative CAMs, 32 Reads and 32 Writes.
- Delayed writes for optimum performance on DRAM data bus
- Support "Out of order" command execution to maximizes the DRAM efficiency
- Programmable DRAM parameters
- Supports BL16 burst length
- Supports 1,2 memory ranks
- Control options to avoid starvation of lower priorities
- Guaranteed coherency for write-after-read (WAR) and read-after-write (RAW) hazards
- Write combine to allow multiple writes to the same address to be combined into a single write to DRAM; supported for same starting address
- Paging policy selectable by configuration registers as any of the following:
  - Leave pages open after accesses, or
  - Close page when there are no further accesses available in the controller for that page, or
  - Auto-Precharge with each access, with an optimization for page-close mode which leaves the page open after a flush for read-write and write-read collision cases
- Supports automatic DRAM power-down entry and exit caused by lack of transaction arrival for programmable time
- Supports automatic Clock Stop entry and exit caused by lack of transaction arrival
- Supports self-refresh entry and exit as follows:
  - Supports automatic self-refresh entry and exit caused by lack of transaction arrival for programmable time
- Support for self-refresh entry and exit under software control
- Support for explicit DRAM mode register updates under software control
- Flexible address mapper logic to allow application specific mapping of row, column,bank, and rank bits
- Programmable support for 1T or 2T timing
- User-selectable refresh control options:

  - Controller-generated auto-refreshes at programmable average intervals
  - In multi-rank designs, an offset can be applied to each rank's refresh timer to allow rank refreshes to expire at different times (this can increase efficiency by allowing traffic to continue to other ranks while a given rank is being refreshed)
- Ability to group up to 8 controller-generated refreshes together to be issued consecutively (this reduces the frequency of page closings, increases overall efficiency)
  - When LPDDR4 is used, per-bank refreshes are scheduled for banks with no traffic in the CAM to minimize impact of refreshes on throughput.
  - When controller-generated refreshes are grouped, some refreshes can be issued speculatively when the controller is idle for a programmable period of time
  - Ability to disable controller-generated auto-refreshes
  - Ability to issue a refresh through direct software request
  - When LPDDR4 is used, user-selectable ability to perform per-bank refreshes rather than all-banks refreshes
  - When DDR4 is used, fine granularity refresh can be selected
  - When controller-generated refreshes are grouped, some refreshes can be issued speculatively when the controller is idle and does not have any HIF transactions to the DRAM rank/bank address for a programmable period of time
- INLINE-ECC support, when controller is configured for 32-bit/16-bit DRAM data width, including the following:
  - Support for single error correction/double error detection (SEC/DED)
- Automatic data scrubbing of correctable errors
- Support for highly efficient read-modify-write when byte enables are used with ECC enabled
- Automatic logging of both correctable and uncorrectable errors
- Ability to "poison" the write data by adding uncorrectable errors, for use in testing ECC error handling

NOTE

1. The DDR4 CRC-Retry feature is disabled based on hardware configuration of DDR subsystem.

36.4.2 PHY features

The PHY supports the following features:

- 8-bit, 16-bit and 32-bit wide SDRAMs
- Support 8-bit and 16-bit DDR3 and DDR4 DRAM devices
- Support 16-bit per channel LPDDR4 DRAM devices
- Support VT compensated delay lines for DQS centering, read/write leveling, and per bit de-skew
- PHY independent, firmware-based training using an embedded calibration processor
  - Utilizes a specialized hardware acceleration engine
  - Automatic periodic retraining through PHY MASTER Interface
  - Support for (LPDDR4) Command Bus eye training relative to CK
  - Support for Write Leveling to compensate for CK-DQS timing skew
  - Support for Write Training: DQS to DQ: Two modes of training are available in LPDDR4:
    - Command-based FIFO WR/RD with user patterns
    - An internal DQS clock-tree oscillator, to determine the need for, and the magnitude of, required periodic training
- Data bus VREFDQ training

- Read training:
  - DQ bit deskew training using DRAM mode registers
  - DQS to DQ eye centering training using DRAM array
- LVSTL IO calibration and ODT calibration
- Three inactive idle states:
  - DFI_LP Mode: most clocks and delay lines gated
  - PHY Inactive: leakage only
  - PHY Retention: Core power removed, most I/Os powered down, SDRAMs held in self-refresh
- Voltage and temperature compensated delay lines used for:
  - Centering the clock in the address/command window
  - Centering the strobes in the data eyes
  - Read/write leveling
  - Per-bit deskew
- Includes a low-jitter PLL for both PHY clock generation and DRAM clock generation
  - Only one PLL is required per DDR PHY
  - Integrated PLL bypass to support 50 to 666 MT/s
- Flexible pre and postamble
  - Support for 1 or 2 clock write preamble
  - Support for static or toggling read preamble of 1 or 2 tCK
  - Support for 0.5 or 1.5 tCK read postamble
- Support for a SW controllable DQ bit and AC bit swizzling
- Includes the PHY utility block (PUB)
  - Soft IP Verilog design that includes PHY control features, such as read/write leveling and data eye training
  - APB and JTAG interfaces for register access
- JTAG register interface for easy test access
- Direct override programming available for all VREF, ODT, drive strength, and timing delays to facilitate debug and characterization
- DDR3 Burst Chop
- DRAM addressing up to 4Gb
- MRR reads and MPC write/read
- Data bus inversion
- Mode register readout
- Support Programmable input on-die termination (ODT) for IOs
- Support Programmable output impedance for IOs
- Support PVT-compensated ODT and output impedance for IOs
- Support Retention feature keeps SDRAMs in self-refresh during VDD power down
- Support Separate driver and receiver I/O power-down control
- Support PAD and internal loopback modes

Licensed under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c4c2e5-4821-48a2-b251-390d3ab624c1

36.5 Functional description

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

The MC converts system bus transactions into memory commands on the DFI interface that are compliant with the DDR protocols.

36.5.1 Exclusive access

The MC controller supports the AXI Exclusive Access feature.

All exclusive read transactions have an EXOKAY response (except in the case of an ECC uncorrectable error). Successful exclusive write accesses have an EXOKAY response, unsuccessful exclusive write accesses return an OKAY response. If an exclusive write fails, the data mask for the exclusive write is forced low so the data is not written. Masked writes or RMW must be supported when using exclusive access.

The exclusive access monitor compares the exclusive write transaction address, size, length, ID and port number against the exclusive read transaction address, size, length, ID and port number, and only accepts an exclusive write when these parameters match. Otherwise, the exclusive write is considered as fail. In some cases SLVERR response may be returned for exclusive access transactions. For more information about this, see “Read Data and Response Channel” and “Write Response Channel” description.

NOTE

- The read response SLVERR is generated for an exclusive read transaction for ECC configurations, when ECC is enabled and an uncorrectable error is detected. The master, upon receiving a SLVERR to an exclusive read command should not complete the exclusive operation by sending an exclusive write command. If it does, the MC monitors return EXOKAY if there is no violation.
- The maximum number of bytes that are monitored as a region per address cannot exceed 128 bytes. The AXI exclusive transaction length must always be a power of two.
- When all the monitors are active, further exclusive read should not be issued. If this happens, one of the active monitors is evicted using a round robin control and new exclusive read is accepted. The selection of the monitor to be evicted is based on a pointer which rotates across all the locations when a new exclusive read is received.

36.5.2 Software coherency for AXI ports

For information about how the MC handles in-order execution of commands to the same address, see Address collision handling. For the commands issued at the AXI port, the logic in the MC protects against all types of software coherency hazards when the AXI master waits for write (read) response before sending the next same address read (same address write) or vice versa.

Some special AXI masters expect an ordering relationship between the reads and writes when the opposite direction transactions are sent without waiting for the previous response. For these special cases, additional logic should be instantiated in the AXI port interface (XPI) to enforce the order of acceptance within the reads and writes between the AXI and HIF interfaces at the MC.

This logic is included by setting of the hardware parameter UMCTL2_RDWR_ORDERED_n (Enabled for all three AXI ports in Hardware configuration) and enabled by the register PCFGR_n.rdwr_ordered_en. When this feature is enabled and read and write transactions are presented at the same cycle, the pre-arbiter gives precedence to read transactions, incrementing the write transaction order token by 1 with respect to the read transaction order token.

For example, if the last order token issued was equal to 0, then the read order token will be equal to 1 and the write order token will be equal to 2.

36.5.3 Host interface (HIF)

HIF is an internal interface when the MC is configured to include a multi-port arbiter (the hardware parameter UMCTL2_INCL_ARB is set to 1). In this case, the AXI Port Interface (XPI) and/or Port Arbiter handle all signals on this interface. Read, write, and RMW requests as well as write data are received through the HIF. Following the receipt and acceptance of a write or RMW request, the MC requests write data from the interface.

The interface subsequently provides the independently timed write data in the requested order (matches the order in which requests are made to the MC). Read, write, and RMW requests are provided through a common command bus if UMCTL2_DUAL_HIF = 0.

Both read and write requests can be reordered later, before being output on the DFI interface. Requests to the MC are throttled in two ways:

- A credit mechanism that ensures that buffer space is available for any request to the MC, prior to the request being made.
- An independent stall mechanism that throttles requests when address collisions take place or if entering self-refresh through the software self-refresh or hardware low power self-refresh.

Data is handled separately. The MC throttles data by potentially waiting after it receives a write or RMW request before requesting the associated data. If there is collision, then the throttling lasts until the collision is cleared in the MC.

Once write data is requested, it is always accepted by the MC. However, if RMW requests are present, the write data can be throttled using the 'hif_wdata_stall' signal.

36.5.4 Partial writes

A Partial Write is where the number of write data beats is less than the number required for a normal (full) write for the MEMC_BURST_LENGTH.

When UMCTL2_PARTIAL_WR = 1 (as set in hardware configuration), the MC issues the minimum number of DRAM bursts on the DDR interface required, dependent on the number of HIF write data beats and the HIF address alignment with respect to the DRAM Column address - as DRAM Writes must be sent BL-aligned.

36.5.5 XPI queues information

The controller interface provides the information about the address queues in the XPI. The information involves address queues for both read and write, and for reads separate information for blue/red queues in case dual queue is selected for that specific port (UMCTL2_XPI_USE2RAQ_$ = 1).

In some cases, an AXI WRAP command at the input may be split into two internal INCR commands and are pushed separately into the queue. This can be recognized when split is asserted. When split and push are high, it means that the first generated INCR command has been pushed into the queue, and the following push means that the second generated INCR command has been pushed into the queue.

36.5.6 Overview of DFI interface

The MC contains a DFI interface, that connects the MC to a DFI-compliant PHY, and to transfer address, control, and data to the PHY. In case of DDR PHY, the MC interfaces to a PUB block, which acts as the DFI PHY interface. The MC, including its DFI Interface, runs at:

Half data rate (HDR) clock (referred to as 1:2 frequency ratio mode in the DFI Specification) You can get more information about the DFI interface from the following specifications:

- DDR PHY Interface (DFI) Specification, Revision 3.1 (Preliminary), 21 March 2014 and DDR PHY Interface (DFI) Specification, Preliminary DFI 4.0 Specification - Addendum to DFI 3.1, 2 April 2014

The DFI interface is sub-divided into the following interface groups:

- **Control interface:** The Control interface is a reflection of the DRAM control interface including address, bank, chip select, row strobe, column strobe, write enable, clock enable and ODT control, as applicable for the memory technology
- **Write data interface:**The Write data interface handles the transmission of write data across the DFI interface. The DFI Specification defines signals and timing relationships. The timing of this interface is user-configurable, to support all DFI-compliant PHYs through the DFITMG0.dfi_tphy_wrlat and DFITMG0.dfi_tphy_wrdata registers.
- **Read data interface:**The Read data interface handles the return of read data across the DFI interface. The DFI Specification defines signals and timing relationships. The timing of this interface is user-configurable, to support all DFI-compliant PHYs through the DFITMG0.t_rddata_en register.

- **Update interface:**During system operation, the system may require updates to the internal settings to compensate for environmental conditions. To ensure that updates do not interfere with signals on the DRAM interface, the DFI interface supports update modes where the DFI read, write, and Control interface is suspended from the normal activity.
- **Status interface:** The DFI interface requires status information for initialization and clock control to the DRAM devices. These signals are used to convey information between the MC and PHY.
- **Training** (PHY-Independent mode supported by the MC): Write and read leveling function performed in a DFI-compliant PHY with no MC interaction.
- **Low power control interface:** This interface consists of signals that are used to inform the PHY of a low power mode opportunity, as well as how quickly the MC requires the PHY to resume normal operation.

1:2 Frequency ratio mode considerations:

The MC, including its DFI interface runs at the half data rate (HDR) clock (referred to as 1:2 frequency ratio mode in the DFI Specification), so all DFI command and address signals are twice the width of the equivalent DDR DRAM signals. DFI data signals are four times the width of the equivalent DDR DRAM signals. The MC operates with an HDR clock. The PHY handles HDR-to-SDR conversion on the address bus to memory and HDR/DDR conversion on the data buses. In 1:2 frequency ratio mode, core_ddrc_core_clk runs at half the frequency of CK/CK#.

For DFI command and address signals in DFI 1:2, the DFI protocol refers to phase 0 (_p0)and phase 1 (_p1) but there are no such *_p0 and *_p1 signals in the MC. Instead, the lower half of the bits corresponds to phase 0 and the upper half of the bits corresponds to phase 1.

Consider the following example:

- In system that has 32 bits of DRAM in DFI 1:2, dfi_wrdata[127:0] is the case. dfi_wrdata[63:0] corresponds to dfi_wrdata_p0 and dfi_wrdata[127:64] corresponds to dfi_wrdata_p1. Similarly, dfi_wr-data_en[7:0] is the case. dfi_wrdata_en[3:0] corresponds to dfi_wrdata_en_p0 and dfi_wrdata_en[7:4] corresponds to dfi_wrdata_en_p1. Similarly, dfi_wrdata_mask[15:0] is the case. dfi_wrdata_mask[7:0] corresponds to dfi_wrdata_mask_p0 and dfi_wrdata_mask[15:8] corresponds to dfi_wr-data_mask_p1.

  For dfi_rddata_valid/dfi_rddata/dfi_rddata_dbi signals in DFI 1:2, the DFI protocol refers to word 0 (_w0) and word 1 (_w1). But, there are no such *_w0 and *_w1 signals in the MC. Instead, the lower half of the bits correspond to word 0 and the upper half of the bits correspond to word 1.
- In system that has 32 bits of DRAM in DFI 1:2, dfi_rddata[127:0] is the case. dfi_rddata[63:0] corresponds to dfi_rddata_w0 and dfi_rddata[127:64] corresponds to dfi_rddata_w1. Similarly, dfi_rd-data_valid[7:0] is the case. dfi_rddata_valid[3:0] corresponds to dfi_rddata_valid_w0 and dfi_rddata_valid[7:4] corresponds to dfi_rddata_valid_w1. Similarly, dfi_rddata_dbi[15:0] is the case. dfi_rddata_dbi[7:0] corresponds to dfi_rddata_dbi_w0 and dfi_rddata_dbi[15:8] corresponds to dfi_rddata_dbi_w1.

36.5.7 Low power software interface

Low Power Interface In a DDR memory subsystem, it is advantageous to place the PHY in a low power state when the MC has knowledge that the memory subsystem remains idle for a period of time. Depending on the state of the system, the MC communicates state information to the PHY allowing the PHY to enter the appropriate power saving state.

Software control is provided in DFILPCFG0/DFILPCFG1 registers (see UMCTL2_REGS) for the following:

- This is an optional interface for a DFI-compliant PHY. The interface signals always exist in the MC. But the software can enable this interface in self-refresh and/or power-down and/or deep power-down mode and/or maximum power saving mode - DFILPCFG0.dfi_lp_en_pd, DFILP-CFG0.dfi_lp_en_sr, DFILPCFG0.dfi_lp_en_dpd and DFILPCFG1.dfi_lp_en_mpsm respectively.
- Software control over what is driven on dfi_lp_wakeup signal (and its associated timing) in self-refresh or power-down or deep power-down or maximum power saving mode - DFILP-CFG0.dfi_lp_wakeup_pd, DFILPCFG0.dfi_lp_wakeup_sr, DFILPCFG0.dfi_lp_wakeup_dpd and DFILPCFG1.dfi_lp_wakeup_mpsm respectively.
- The timing of this DFI low power interface - DFILPCFG0.dfi_tlp_resp.

- ctl_idle output pin is a sideband signal (not specified in DFI) supported by PHY to trigger the PHY's Anti-Aging feature. This is driven the same as dfi_lp_req if enabled through soft-ware (DFIMISC.ctl_idle_en = 1 and DFI Low Power Interface is also enabled through DFILP-CFG0/DFILPCFG1).

36.5.8 Transaction service control

The transaction service control allows you to carefully manage the following:

- Costly read/write bus turn-arounds
- Priorities of read requests to generally favor high priority traffic while also preventing starvation of low priority traffic

This functionality is implemented in a simple 2-state machine for each traffic type with completely configurable controls. The states of the 2-state machine determine when reads/writes are serviced and the relative priority (high priority versus low priority reads) at any given moment.

Transactions in the MC are separated into five categories:

- Low Priority Reads (LPR)
- Variable Priority Reads (VPR)
- High priority reads (HPR)
- Normal priority writes (NPW)
- Variable priority writes (VPW)

36.5.9 Page policy

36.5.9.1 Explicit auto-precharge (Per Command)

The explicit auto-precharge feature can enable auto-precharge on a per-command basis. If the HIF signal hif_cmd_autopre is set during a valid command, then the auto-precharge bit for that command is set when it is sent to the DRAM. If you do not want to use the per-command auto-precharge feature, then this bit can be tied to 1'b1 or 1'b0. Tying it to 1'b1 causes all the commands to be executed with auto-precharge, and tying it to 1'b0 causes no commands to be executed with auto-precharge. For AXI configurations, the HIF signal hif_cmd_autopre is tied to 0 internally, and is not accessible to the user.

36.5.9.2 Intelligent precharges

The SCHED.pageclose register (see UMCTL2_REGS Registers) enables precharge commands to be issued smartly through auto-precharges or explicit precharges. The exact functionality depends on the value programmed in SCHED1.pageclose_timer:

- If SCHED.pageclose is set to 1 and SCHED1.pageclose_timer = 0, a bank is kept open while there are page hit transactions available in the CAM to that bank. The last read or write command in the CAM with a bank and page hit is executed with auto-precharge. The intelligent precharge logic considers only the CAM or CAM region that is currently active, i.e. while executing reads it looks at the LPR or HPR region of the read CAM (whichever is active at the time) and does not look at transactions in the other region of the read CAM or in the write CAM; while executing writes the logic does not look at transactions in the LPR or HPR regions of the read CAM.

  Whenever a mode switch occurs between LPR, HPR and writes, the page context is lost and therefore, a redundant auto-precharge may be issued followed by an explicit activate to the same bank or only an explicit precharge may be issued instead of an auto-precharge. The read and write commands that are executed as part of the ECC scrub requests are also executed with explicit precharge if the page needs to be closed.
- SCHED.pageclose is set to 1 and SCHED1.pageclose_timer >0, a bank is kept open while there are page hit transactions available in the CAM to that bank. The last read or write command in the CAM with a bank and page hit is not executed with auto-precharge. Instead, a timer is started with page-close_timer as the initial value

  There is a timer on a per bank basis. The timer decrements unless the next read or write in the CAM to a bank is a page hit. It resets to pageclose_timer value if the next read or write in the CAM to a bank is a page hit. Once the timer reaches zero, an explicit precharge is scheduled. Prior to a bank's timer expiring, an explicit precharge may also be scheduled if the transaction scheduler chooses a CAM entry that is a row miss to the same bank.

- If SCHED.pageclose is set to 0, the bank remains open only until there is a need to close it (to open a different page or for page timeout or refresh timeout). This is also known as open page policy. The open page policy can be overridden by setting the per command auto pre bit on the HIF interface (hif_cmd_autopre) as described in “Explicit Auto-Precharge (Per Command. When the multi-port arbiter is configured, the HIF signal hif_cmd_autopre is tied to 0 and you do not have control over it.

36.5.9.3 Address collision handling

The MC can execute transactions out-of-order while ensuring that all transactions appear as if they are executed in the order in which they are received. Every transaction that requires a response from the MC arrives with a token number which is provided back to the core as part of the response. Since the MC queues transactions prior to execution, it is possible that multiple transactions to the same DRAM address can arrive before the first transaction to that address is issued. For address collision, two HIF addresses are considered the same address if all of the HIF address bits (except for the LSB column values) are the same. That is, the “collision” ignores the LSB column values when comparing addresses.

To enforce ordering of accesses to the same address, the MC uses the following algorithm:

1. **New read colliding with queued read:** This collision causes no problems. The two reads can end up being executed out-of-order.
2. **New write colliding with queued write:** If write combine is enabled, the MC overwrites the data for the old write with that from the new write and only performs one write transaction (write combine). For more information about this, see “Write Combine” section.
3. **New read (or write) colliding with queued write (or read) respectively:** In this case, the MC performs the following sequence:
    a. Holds the new transactions in a temporary buffer
    b. Applies flow control back to the SoC core to prevent more transactions from arriving.
    c. Flushes the internal queue holding the colliding transaction until that transaction is serviced.
    d. Accepts the new transaction and removes flow control.
4. **New read colliding with both read and write:** This can happen when a read collides with an RMW command. In this case, the reads are flushed until the read collision is cleared, then the writes are flushed in the same manner as described in the Step 3.
5. New write colliding with both read and write: This can happen when a write collides with an RMW command.
    - If write combine is enabled, the new write is combined with write part of queued RMW.
    - If write combine is disabled, the new write is held in a temporary buffer until the RMW has completed.
6. **New RMW colliding with queued write:** In this case, the new RMW is stored in a temporary buffer until the queued write is completed.

36.5.9.4 Write combine

The write combine feature can combine multiple writes to the same address into a single write to DRAM. When a new write collides with a queued write in the CAM:

- If write combine is enabled, the MC overwrites the data for the old write with that from the new write and only performs one write transaction (write combine).
- If write combine is disabled, the MC performs the following sequence:
    1. Holds the new write transaction in a temporary buffer.
    2. Applies flow control back to the SoC core to prevent more transactions from arriving.

36.5.9.4.1 Registers related to transaction service control

The following are the registers related to the Transaction Service Control:

- Low priority read transaction store
  - PERFLPR1.lpr_max_starve
  - PERFLPR1.lpr_xact_run_length
- High priority read transaction store
  - PERFHPR1.hpr_max_starve
  - PERFHPR1.hpr_xact_run_length
- Write transaction store
  - PERFWR1.w_max_starve
  - PERFWR1.w_xact_run_length
- Bus turn-around control
  - SCHED.prefer_write
  - SCHED.rdwr_idle_gap

36.5.10 Overview of quality of service

In a SoC based sub-systems, Quality of Service (QoS) are extremely important in distributing the available resources (bandwidth or latency) amongst the different devices, in a fair mechanism to meet the overall performance requirements.

In any SoC based on-chip communication system, there will be multiple Masters trying to communicate with different Slaves through Interconnect. Interconnect is used to route the traffic between the Masters and Slaves. There are potential chances of bandwidth sensitive Master starving for data and latency sensitive Master requirements are not met. This impacts the overall performance of the sub-system. QoS addresses these kinds of performance issues in sub-system.

Traffic classes: The MC supports different traffic classes that are distinguished from each other by the arqos signal.

36.5.10.1 Read classes

- **Low priority read (LPR):** It is also called as the best effort traffic. There is no resource allocation and it shares the resources with the other traffic types. LPR is always treated as low priority in the port abiter (PA) and in the MC. There are timeout mechanisms that can be used to prevent starvation for both the PA and the MC. When there is a timeout in the PA, that port becomes the highest priority (priority0). When there is a timeout in the LPR store in the MC (in other words, LPR store becomes critical), the LPR entries are served before HPR entries.
- **Variable priority read (VPR):** It is also called as the maximum latency bound traffic for meeting real time deadlines in video or audio applications. VPR traffic shares the same resources with LPR in the MC. But, based on the configuration, it may have a dedicated queue (red or blue) in the XPI. It has the same priority as LPR, but lower priority than HPR for the PA. For the MC, VPR has the same initial priority as LPR. Each command tagged as VPR has an associated latency timer. When expired, VPR transactions have the highest priority in the controller, both in the PA and in the MC. The purpose of this traffic class is to limit the maximum latency, where this latency bound is expected to be a few hundred clock cycles.
- **High priority read (HPR):** It has allocated resources. If the XPIs are configured to have dual read address queues, then the red queue can be allocated to HPR. For the MC, HPR traffic goes to its own dedicated store. HPR traffic has higher priority than LPR. HPR is meant for latency critical but not real time applications such as CPU.

36.5.10.2 Write classes

- **Normal priority write (NPW):** It is also called as the best effort traffic. There is no resource allocation and it shares the resources with the other traffic types. It is always treated as normal priority in the PA and MC. There are timeout mechanisms that can be used to prevent starvation for the PA. When there is a timeout in the PA, that port becomes the highest priority (priority0).
- **Variable priority write (VPW):** It is also called as the maximum latency bound traffic to meet real time deadlines in video or audio applications. VPW traffic shares the same resources with NPW in the MC. For the MC, VPW has the same initial priority as NPW. Each command tagged as VPW has an associated latency timer. When expired, VPW transactions have

the highest priority in the controller, both in the PA and in the MC. The purpose of this traffic class is to limit the maximum latency, where this latency bound is expected to be a few hundred clock cycles.

36.5.11 Significance of urgent signal on QoS

The application master can indicate to the uMCTL2 Port Arbiter that it is going to starve soon by asserting the AXI sideband signals like arurgent and awurgent. These sideband signals can be asserted anytime and are not associated with any particular command. When an arurgnet or wurgent signal is asserted, the priority of that port becomes the highest (Priority0) and that causes the read/write direction to switch immediately at the Port Arbiter level if enabled by PCFGR_n[rd_port_urgent_en] and PCFGW_n[wr_port_urgent_en]. These AXI-Urgent signals also assert the hif_go2critical_wr/ hif_go2critical_lpr/_hpr signals at the HIF interface level, which forces the read/write direction switch within DDRC, if enabled by PCCFG[go2critical_en].

36.6 ECC scrubber

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

ECC Scrubber (SBR) is a block which initiates periodic background read commands to the MC. The purpose behind periodic scrubbing is to increase the reliability of the system by correcting single-bit ECC errors in time, before they turn into uncorrectable 2-bit errors. With Inline ECC, scrub feature of the MC is not supported, for this reason RMW command is initiated by the Scrubber itself for every single-bit ECC error detected. Scrub command is a read of one memory burst (for example, BL8) which is sent to the MC periodically with the lowest priority. The SBR consumes one of the ports of the Port Arbiter (PA) as a requester. The SBR terminates and dumps the read data. For Inline ECC the Scrubber itself initiates the required RMW commands as necessary to fix single-bit ECC errors.

- The controller does not scrub automatically for each correctable read error (for Inline ECC, always ECCCFG0.dis_scrub = 1).
- ECC Scrubber (SBR) block is enhanced to issue fully masked (all bytes disabled) single read-modify-write command after detecting a correctable read error (a new HIF signal hif_rdata_corr_ecc_err is provided).
- For arbiter configurations only.

36.6.1 Scrub period

Scrub interval (tSCRUBI) is defined as number of clock cycles between two scrub read commands. Scrub period is defined as the time needed to read every memory location of the entire DRAM address range. The period ranges from 2 hours to 12 hours depending on the tSCRUBI setting, memory size, and clock frequency. The register SBRCTL.scrub_interval[12:0] counts in multiples of 512 cycles. With inline-ECC, whenever SBRCTL.scrub_intrval is programmed to 0, the scrubber will not issue any RMW command to the MC if a single bit ECC errors are detected.

36.6.2 Normal operation

For Inline ECC a minimum of 8 bursts (1 block) is issued to the MC each time; the amount of burst sent is set by programming SBRCTL[scrub_burst] [2:0] register. Each time tSCRUBI counter times out, a scrub command (or a burst of commands for Inline ECC) is pushed to a scrub queue (8-deep), which is then immediately issued to the MC.The SBR counters halt if more than 8 scrub commands are collected and pending inside the scrub queue due to the MC being busy. The SBR starts the reads from address 0 up to the maximum address as defined by some of the controller parameters and address map register settings.

The address range is automatically calculated. However, for debugging purpose (for example, to speed up testing of it), the start address can be overridden by the external signal sbr_address_start_mask, while the maximum address range can be overridden by the external signal, Scrubber address range mask. If there are outstanding read commands on flight issued by the SBR, SBRSTAT[scrub_busy] register is asserted.

For inline ECC configurations that use an ECC scrubber, if the Scrubber address range mask signal is set to any of these values: unprotected region, ECC region, or zero, the scrubber loops back to the starting address even after it completes writing or scrubbing the last memory location. SBRSTAT[scrub_done] is not always asserted.

Therefore, the end of the initialization write operation cannot be determined, and a software timer is required to determine the period to initialize or scrub the memory space. This is done using the scrubber. You also need to stop the scrubber by writing 0 to SBRCTL[scrub_en], after the time is over.

NOTE

In Inline ECC configurations , the 'ECC Scrub' feature is not supported and is always disabled by setting the ECCCFG0[dis_scrub] = 1'b1.

36.6.3 Low power operation -software controlled

If the MC is in one of the software controlled low power modes such as self-refresh, deep power-down (DPD), or maximum power saving mode (MPSM), it is recommended to follow the procedure mentioned below:

- Before entering into any of the software controlled low power modes, disable the SBR by setting SBRCTL.scrub_en to 0.
- If scrubbing is required in this low power mode, the whole address range can be quickly scrubbed by issuing back-to-back scrub read commands.
  - Software must wake up the MC and initialize the DRAM/PHY as necessary.
  - To start the SBR in an accelerated rate, write SBRCTL[scrub_interval] to 0 and enable the SBR.
  - Once the SBR scrubs the entire DRAM range, sbr_done_intr interrupt signal is asserted. In addition, SBRSTAT.scrub_done software status bit is set which can be polled by the software. The scrubbing operation continues until SBR is disabled.

NOTE

Note that sbr_done_intr signal and scrub_done register are never set if the SBR is operating in its normal mode (SBRCTL[scrub_interval] > 0).

  - If the software does not want to wait for the complete scrubbing operation to finish, SBRCTL[scrub_interval] can be set to a non-zero value anytime. The SBR will re-start the scrub commands from the start address.
  - Disabling the SBR or setting the SBRCTL[scrub_interval] back to its normal operating value (tSCRUBI) which is greater than 0 clears both the interrupt signal and the status bit.

NOTE

When you use scrubber in the Inline ECC mode, the SBR counter reaches the configured SBRCTL[scrub_interval] and the controller is in the LP mode. In this case the next set of SBR reads is not generated.

Therefore, disable SBR during the Low-Power mode by writing 0 to SBRCTL[scrub_during_lowpower].

36.6.4 Initialization writes

The above sections refer to scrub read commands. The SBR block can be used to initialize the memories with a defined pattern. SBRCTL.scrub_mode should be programmed to 1. SBRWDATA0 can be programmed with the desired pattern.

After PHY and DRAM initialization, perform the following steps:

- If MEMC_INLINE_ECC is defined and ECC is enabled, set ECCCFG1.ecc_parity_region_lock to 1.
  1. Ensure that PCTRL_n.port_en are set to 0, block the AXI ports from taking the transaction.
  2. Set SBRCTL.scrub_mode = 1.
  3. Set SBRCTL.scrub_interval = 0.
  4. Set the desired pattern through SBRWDATA0 register.
  5. Enable the SBR by programming SBRCTL.scrub_en = 1.
  6. Poll SBRSTAT.scrub_done = 1 (all scrub writes commands have been sent).
  7. Poll SBRSTAT.scrub_busy = 0 (all scrub writes data have been sent).
  8. Disable SBR by programming SBRCTL.scrub_en = 0.
  9. Prepare for normal scrub operation (Reads) by programming SBRCTL.scrub_mode = 0 and SBRCTL.scrub_interval to tSCRUBI.

10. Enable the SBR by programming SBRCTL.scrub_en = 1. 11. Enable AXI ports by programming PCTRL_n.port_en = 1.

36.7 Overview of inline ECC support

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

The main features of Inline ECC :

- ECC parity (code) is stored together with the data without using a dedicated sideband memory device.
- Inline ECC is a necessity for LPDDR4 due to device characteristics and device topology.
  - Inline ECC is supported with LPDDR4, DDR4 (BG rotation address mapping), and DDR3 protocols.
- The supported memory data widths are 16, and 32.
- For inline ECC, 64/8 SECDED Hamming code is used:
  - Data to ECC ratio is 8/1

36.7.1 Address map

There are few definitions mentioned below:

- **Data block:** 8 burst accesses of data (a burst can be DRAM BL8 or BL16).
- **Super-block:** Block with ECC parity (9 burst accesses).
- **Hole:** ECC parity + wasted region of the memory.
- **Usable memory size:** All the data blocks that fits.
- **Usable size ratio:** Usable memory size/Full memory size.
- **Utilization percentage:** The percentage of memory used for Data and ECC.

When Inline ECC is enabled, the highest three column bits must be mapped to the highest address map position possible. The controller flexible address mapping scheme is constrained so that the highest system address space is reserved for ECC parity and waste as one region as shown in Figure below. For the normal data in all remaining regions, the system address is linear and continuous. The reason for the waste is due to how the memory is divided for ease of implementation. Data to ECC ratio is 8 to 1 (divide by 9), but the memory is mapped 7 to 1 (divide by 8 which is a power of 2).

- Usable size ratio is 7/8 = 87.5%.
- Utilization percentage is (7/8 + (1/8*7/8)) = 63/64 = 98.4% and the remaining becomes the waste. The unprotected wasted area can still be accessible by the system, but not in linear and continuous fashion.

Provided under NDA
COMPANY CONFIDENTIAL
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a8c-9251-390d3ab6246



Figure 184.

For LPDDR4 and DDR3, for best Inline ECC performance, it is recommended to have address mapping such that Data and ECC to be mapped to the same bank and page.

For DDR4, it is recommended to have bank group rotation address mapping. In this case ECC parity is in a different bank and bank group than the data

36.7.2 Selectable protected regions

The memory is divided into eight regions. To maintain uninterrupted linear system address space, ECC hole maps to the top of the system address.The ECC hole size, including the waste, is 1/8 of the memory size. Except for the last region, which is the ECC hole, the remaining seven regions can be set to be either ECC protected or not. This feature allows ECC protection only on the critical safe areas of the memory. Remaining unprotected addresses can be accessed with no performance overhead since ECC commands are not injected.

The register ECCCFG0.ecc_region_map[6:0] indicates whether each region is ECC protected or not. To protect a region, set the corresponding bit to 1, otherwise set to 0.

SYS Addr	A24	A23	A22	A21	A20	A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
SDRAM Addr	C9	C8	C7	R13	R12	R11	R10	R9	R8	R7	R6	R5	R4	R3	R2	R1	R0	C6	C5	C4	C3	C2	C1	C0	
Unprotected	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
Unprotected	0	0	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
Protected	0	1	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
Protected	0	1	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
Protected	1	0	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
Unprotected	1	0	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
Unprotected	1	1	0	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	
ECC Hole	1	1	1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	

Figure 185. ECC Regions Example

36.7.3 Locking of ECC region

The ECC region comprises two separate sub-regions:

1. Used sub-region:
   - ECC parity area for ECC protected regions.

2. Unused sub-region:
    - ECC parity area for unprotected regions (reserved but not used by inline ECC logic).
    - Waste region, which is never used by inline ECC logic.

The Used sub-region is locked/unlocked by the register ECCCFG1.ecc_region_parity_lock. The Unused sub-region is locked/unlocked by the register ECCCFG1.ecc_region_waste_lock.

In normal conditions, the user software should not access the Used region of the ECC hole and therefore should be kept locked. However, for debugging or error injection purposes into ECC parity, it can be unlocked by the ecc_region_parity_lock register. If required, for normal transfers, the user software can access the Unused region of the ECC hole and therefore can be kept unlocked using the ecc_region_waste_lock register. However, it should be noted that the Unused region is not consecutive in terms of system addresses and it is interleaved with the Used region.

Locking the ECC hole sub-regions from access means the following:

- Any write or RMW request to this region is discarded, while the HIF output hif_wdata_ptr_addr_err is asserted.
- Any read request to this region is executed to the memory, but without the ECC check and returns all zeros, while the output hif_rdata_addr_err is asserted.

For AXI configurations, any assertion of hif_wdata_ptr_addr_err or hif_rdata_addr_err also results in SLVERR/ERROR response. When Inline ECC is disabled in the software, all the memory space can be accessed.

36.7.4 Error injection through software (ECC data poisoning)

ECC error injection is useful for system-level software validation. There is no dedicated hardware support for it. However, errors can be easily injected through the software by unlocking the ECC region through the ecc_region_parity_lock register and overriding certain ECC parity bits. When the corresponding addresses are read from a protected memory region, ECC errors are generated as correctable or uncorrectable, depending on the type of error introduced.

36.7.5 Performance

The idle command latency of the controller increases by one core clock cycle with inline ECC.

- Full block write: No additional latency
    - ECC burst is appended after 8 data bursts, the write data response is generated normally after each data burst.
    - Back to back full write blocks can be streamed with 88.9% application port efficiency and 100% DDR efficiency (this assumes single block channel).
    - For writes, the overhead command is not appended until the block channel is overridden or flushed. For example, for four block channels, it is possible to achieve 100% utilization up to 8*4 sequential/interleaving write bursts.
- For block read: BL/2 additional latency (4 DDR cycles for BL8, eight DDR cycles for BL16)
    - First, read the ECC burst, then read the eight consecutive data bursts. The first read command of the block channel incurs the latency.
    - Back to back full read blocks can be streamed with 88.9% application port efficiency and 100% DDR efficiency.

36.7.6 Interrupts related to inline ECC

The following are the interrupts for Inline ECC:

- ecc_corrected_err_intr
- ecc_uncorrected_err_intr
- ecc_ap_err_intr

36.8 Refresh controls

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

36.8.1 Overview of refresh controls

Refresh can be issued using the auto-refresh feature or using the direct software request of the refresh command. The “RFSHCTL3.dis_auto_refresh” register bit selects the refresh method. When this bit is set to 1, the MC uses direct software request of refresh command. When it is set to 0, the internal auto-refresh feature is used.

The purpose of refresh control is to:

- Reduce the bandwidth impact of refresh cycles
- Increase the likelihood of refreshes being serviced during idle periods
- Provide fine-gain control of the trading-off the above benefits (to gather refreshes) versus the increased worst case latencies associated with gathering refreshes
- Allow traffic to flow to other ranks while a given rank is being refreshed (for multi-rank configurations of the MC only)

36.8.2 Refresh using direct software request of refresh command

Follow these steps to put the MC in direct software request of refresh command mode:

1. Set the RFSHCTL3.dis_auto_refresh bit to 1. When the register bit set, the MC checks for any pending refreshes. Any pending refreshes are issued right away using the ‘critical refresh’ feature inside the MC. After these refreshes are issued, all the refresh timers inside the MC are reset to 0. They are re-activated only when the auto-refresh feature is enabled.
2. The SoC core must keep track of the refresh requirements of the DRAM.
3. The refresh command can be issued by setting the register bits DBGCMD.rank*_refresh to 1. When the rank*_refresh request is stored in the MC, the corresponding register bit is automatically cleared. The SoC core can initiate a rank*_refresh operation only if DBGSTAT.rank*_refresh_busy is low. The MC issues refresh to the DRAM at the earliest.
4. Software-driven refresh commands for each rank are loaded into a 9-entry buffer, and are issued by the MC on the DFI as soon as it is legal to do so (the MC must wait tRFC(min) between each refresh request). If the buffer saturates, the DBGSTAT.rank*_refresh_busy remains asserted to prevent software from initiating further refreshes.

NOTE

- Application can give a burst of back-to-back refresh commands without polling the DBGSTAT.rank*_refresh_busy register field to reduce the time between the software refresh commands to a minimum with the following risk:
  - If the refresh buffer is not empty when the burst starts or the burst is longer than 9 refresh commands, the buffer saturates (DBGSTAT.rank*_refresh_busy is asserted) and some APB writes given in the burst are not added to the buffer. This means that the number of APB writes do not reflect the number of REF commands sent out to the memory.
- For LPDDR4 configuration manual refreshes (RFSHCTL3.dis_au-to_refresh = 1) may not be used if the PHY Master Interface is enabled (DFIPHYMSTR.dfi_phymstr_en = 1).

36.8.3 Refresh using auto refresh feature inside the MC

The MC provides advanced refresh controls. Besides fully-configurable refresh constraints (tRFC(min) and tREFI), the MC can also be programmed to gather refreshes to each rank of DRAM to reduce the bandwidth consumed by refreshes and to increase the likelihood that refreshes can be serviced during an idle period.

Fine-grain control of the refreshes ensures these benefits can be balanced against worst case latencies associated with servicing refreshes together. Staggered refresh timers for multi-rank configurations of the MC allow transactions to continue to other ranks while refreshes are taking place to just one rank.

To minimize the worst case impact of a forced refresh cycle, the MC can be programmed to issue single refreshes at a time by forcing RFSHCTL0.refresh_burst = 0. It can be programmed to burst up to 8 refreshes (RFSHCTL0.refresh_burst = 7) in non-DDR4 modes. In LPDDR4 mode, with per-bank refresh enabled (RFSHCTL0.per_bank_refresh = 1), the maximum refresh burst supported by MC is 64 (RFSHCTL0.refresh_burst = 63). In DDR4 mode, if using Fine Granularity Refresh feature, the number of refreshes that can be bursted is up to 16 refreshes in Fixed 2x mode and up to 32 in Fixed 4x mode. Burst refresh can be used to minimize the bandwidth lost to closing pages for refresh and to increase the likelihood that refreshes can be serviced during idle periods. It can also be programmed to any number in between to trade-off the benefits of each.

36.8.3.1 Single refresh

When using single refresh (RFSHCTL0.refresh_burst = 0), the MC issues refreshes every time the refresh timer (tREFI) expires. This is the optimal mode of operation for systems that minimize the maximum latency associated with refresh cycles.

36.8.3.2 Burst refresh

When burst refresh is enabled (RFSHCTL0.refresh_burst > 0), the MC issues refreshes in bursts of (RFSHCTL0.refresh_burst+1) refreshes at one time. Bursting refreshes reduces the total latency associated with those refreshes by reducing the number of pre-charges and activates required for refresh, as banks must be precharged only once to perform the entire group of refreshes, instead of once for each refresh.

36.8.3.3 Speculative refresh

When tREFI has expired at least once, the MC can also perform speculative refreshes. This is done by automatically inserting refreshes when there are no transactions pending in the CAM to a rank/bank address. The RFSHCTL0.refresh_to_x1_x32 register determines how long the transactions must not be stored in the CAM before considering inserting these speculative refreshes. Each time a speculative refresh is performed, the count of tREFI expirations is decremented, and thereby increasing the time before a critical refresh is required.

This also ensures that speculative refreshes never occur more often than is required to keep the DRAM properly refreshed. If a new read or write transaction is accepted by the MC during speculative refresh, the MC services it as soon as legally possible. Most often it entails waiting for the required NOP cycles after a refresh before performing an activate and then servicing the read or write. If the MC has begun closing pages for a speculative refresh but has not yet issued the refresh when the new transaction arrives, the speculative refresh is canceled.

36.8.4 Per-bank Refresh (LPDDR4 only)

If RFSHCTL0.per_bank_refresh is set to 1 , the MC performs per-bank refreshes instead of all-bank refreshes. In this case, RFSHTMG.t_rfc_nom_x1_x32 and RFSHTMG.t_rfc_min should be set to the appropriate values for per-bank refresh (tREFIpb and tRFCpb respectively). In this mode, the MC keeps track of which bank is being refreshed at any time, and is able to schedule commands to other banks immediately before and after the per-bank refresh commands, resulting in potential efficiency gains. To improve the accuracy of per-bank refresh timing, set RFSHTMG.t_rfc_nom_x1_sel to 1 and program RFSHTMG.t_rfc_nom_x1_x32 accordingly.

For LPDDR4, the per-bank refresh command can be issued in any bank order. The MC can send the per-bank refresh to banks as a priority, where no transactions pending in the CAM or the bank is precharged. Also, the MC does not send the per-bank refresh command to multiple ranks at the same time, as the command contains a bank address, and the MC cannot specify multiple bank addresses with one command.

36.8.5 Fine granularity refresh (DDR4 only)

This feature is supported only by DDR4. As the device density increases, tRFC gets longer and limits the performance. Consequently, frequent refresh with shorter tRFC may be advantageous in certain traffic scenarios. DDR4 allows the following types of fine granularity refresh modes. They are set by programming the DRAM mode register MR3 A[8:6] and RFSHCTL3.refresh_mode registers:

- 000: Fixed 1x (normal mode)
- 001: Fixed 2x

- 010: Fixed 4x
- 101: On-the-fly 1x/2x (not supported by the MC)
- 110: On-the-fly 1x/4x (not supported by the MC)

The MC only supports Fixed mode refresh modes. The mode of operation has to be programmed both in the DRAM mode register and RFSHCTL3.refresh_mode registers. This must happen before the deassertion of reset or when controller is in self-refresh mode, similar to other static registers. When updating refresh mode in self-refresh, INIT4.emr3 must be updated as well, because the value written to the DRAM through automatic MR3 command is taken from INIT4.emr3, not RFSHCTL3.refresh_mode. Fixed 2x and Fixed 4x modes have shorter tREFI and tRFC (min) values compared to Fixed 1x mode. The following restrictions apply when using Fine Granularity Refresh mode:

- On-the-fly refresh modes are not supported
- Only Fixed 1x refresh mode operation is allowed if Temperature Controlled Refresh (TCR) is enabled in the DRAM (MR4 A[3:2])
- Multi-purpose register (MPR) reads and writes are not allowed when the Controller is in Fixed 2x or Fixed 4x modes.
    1. Move the controller and DRAM into self-refresh
    2. Change the controller operation to Fixed 1x mode (RFSHCTL3.refresh_mode = 3'b000)
    3. Program the DRAM operation to Fixed 1x mode (INIT4.emr3 = 3'b000)
    4. Exit self-refresh On exiting self-refresh, the controller automatically issues MR3 write and puts the DRAM in Fixed 1x mode. It is now possible to perform MPR operation as the controller and DRAM are in Fixed 1x mode. If the system wants to put the controller and DRAM back into Fixed 2x or Fixed 4x mode, it should repeat the above steps by moving the controller back into self-refresh, changing the refresh mode and the exiting self-refresh.
- RFSHCTL0.refresh_burst has the following restrictions:
    - In Fixed 1x mode, this must be less than 8
    - In Fixed 2x mode or on-the-fly 2x mode, this must be less than 16
    - In Fixed 4x mode or on-the-fly 4x mode, this must be less than 32
- DDR4 protocol requires the following requirements before self-refresh entry:
    - Even number of refreshes required in Fixed 2x mode
    - Multiple of four refreshes required in Fixed 4x mode

The controller takes care of this requirement. Note that the controller enters self-refresh only after waiting for all ranks to meet the above requirements (especially the even/multiple of four requirement for Fixed 2x/Fixed 4x modes). Hence, larger the difference between refresh_timer*_start_value_x32 settings, longer the wait before self-refresh entry occurs.

36.8.6 Constraints on refresh_timerX_start_value_x32

The refresh_timerX_start_value_x32 register fields control the relative timing of refreshes to different ranks. The register fields must obey the following constraint: refresh_timerX_start_value_x32 + RoundUp(RFSHTMG.t_rfc_min/32) < RFSHTMG.t_rfc_nom_x1_x32

If RFSHTMG.t_rfc_nom_x1_sel is set, RoundDown(RFSHTMG.t_rfc_nom_x1_x32/32) should be used in the above constraint. Also, note that the register field refresh_timerX_start_value_x32 cannot be changed after initialization. So the value of the register RFSHTMG.t_rfc_nom_x1_x32 used above should be anticipated as the minimum value which will be used, considering frequency change, fine granularity refresh, etc.

36.9 ZQ calibration

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

36.9.1 Overview of ZQ calibration

This feature is applicable to DDR3/DDR4 and LPDDR4. The MC uses ZQ calibration command to calibrate DRAM RON (Resistor ON) and ODT (On-Die Termination) values over PVT (Process, Voltage, Temperature). DDR3/DDR4 and LPDDR4 SDRAMs need more time to calibrate RON and ODT at initialization and relatively less time to perform periodic calibrations.

36.9.2 DDR3/DDR4 devices

ZQCL (ZQ Calibration Long) command is used as follows:

- To perform the initial calibration during the power-up initialization sequence. This command is allowed a timing period of tZQinit determined by INIT5. dev_zqinit_x32 to perform full calibration and the transfer of values.
- To perform long ZQ Calibration after exiting from self-refresh mode (or after exiting from maximum power saving mode in DDR4 mode). This command is allowed a timing period of tZQOPER, deter-mined by ZQCTL0._t_zq_long_nop. This command is automatically issued when ZQCTL0.dis_srx- _zqcl (ZQCTL0.dis_mpsmx_zqcl) is set to 0. To disable issuing ZQCL after self-refresh exit (after maximum power saving mode exit in DDR4 mode), set ZQCTL0.dis_srx_zqcl (ZQCTL0.dis_mpsmx- _zqcl) to 1.

ZQCS (ZQ Calibration Short) command is used to perform periodic calibration to account for VT (Voltage/Temperature) variations. A shorter timing window is provided to perform calibration and transfer of values as defined by the timing parameter tZQCS (determined by ZQCTL0._t_zq_short_nop).

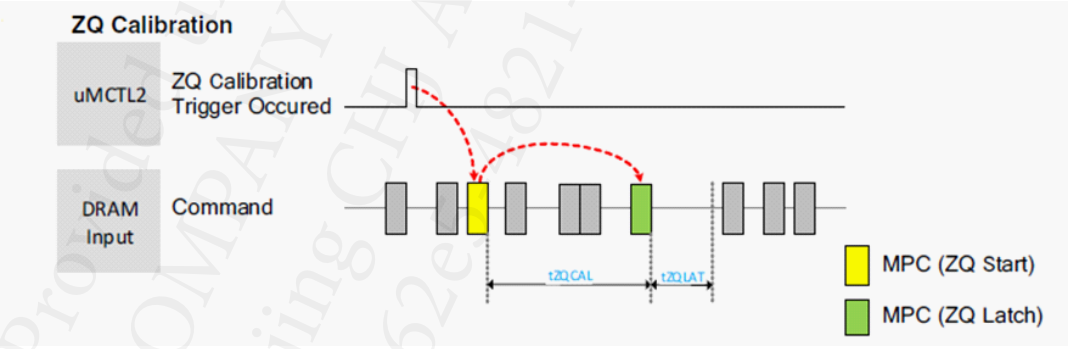
ZQCS can be performed automatically on a regular interval or through direct software request.. In DDR3/DDR4 mode, the ZQ Calibration commands are sent out encoded on the DFI command bus as mentioned in the JEDEC Specification. The MC performs no other activities for the duration of tZQinit, tZQOPER and tZQCS. The quiet time on the DRAM channel helps in accurate calibration of DRAM RON and ODT. All banks are precharged and tRP met before the MC issues the ZQ Calibration commands.

36.9.3 LPDDR4 devices

The MC supports all the ZQ calibration commands that are supported by the JEDEC Specification. ZQ calibration sequence uses the following MPC commands:

- ZQCal Start
- ZQCal Latch

The ZQCal Start command is used to initiate DRAM's calibration procedure. The ZQCal Latch command is used to capture the result and load it into DRAM's drivers. ZQCal Latch command can be issued after tZQCAL (determined by ZQCTL0._t_zq_long_nop) has expired and all DQ Bus operations have completed. If a ZQ calibration is triggered, those commands are sent out consecutively to the DRAM by the MC.



ZQCal Start and ZQCal Latch command can be performed automatically at a regular interval or through direct software request. Furthermore, both commands are issued automatically after SR-Powerdown exit if ZQCTL0. dis_srx_zqcl is set to 0. To disable issuing of both commands after SR-Powerdown exit, set ZQCTL0. dis_srx_zqcl to 1.

Do not change the following mode register fields following a ZQCal Start command and before tZQCAL has expired:

- PU-Cal (Pull-up Calibration VOH Point) - MR3:OP[0]

- PDDS (Pull Down Drive Strength and Rx Termination) - MR3:OP[5:3]
- DQ-ODT (DQ ODT Value) - MR11:OP[2:0]

36.9.4 Automatic and software Initiated ZQCS

The MC issues a ZQCS command in the following ways:

- **Automatic ZQCS by the MC:**

  In this case, the MC sends ZQCS commands to DRAM periodically. The interval is determined by ZQCTL1.t_zq_short_interval_x1024. This method is used if ZQCTL0.dis_auto_zq is set to 0. In LPDDR4 mode, the MC sends MPC-based ZQ calibration commands (ZQCal Start and ZQCal Latch) instead of ZQCS command.

- **ZQCS using direct software request:**

  In this case, the SoC core sends a ZQCS command through software by setting DBGCMD.zq_calib_short to 1. When the ZQCS request is stored in the MC, the register bit is automatically cleared. It is recommended not to set DBGCMD.zq_calib_short signal in init, self-refresh, deep power-down operating modes or maximum power saving mode (MPSM). The SoC core can initiate a ZQCS operation only if DBGSTAT.zq_calib_short_busy is low. The DBGSTAT.zq_calib_short_busy signal goes high in the clock after the MC accepts a ZQCS request. It goes low when the ZQCS operation is initiated in the MC. For proper DRAM operation, user/SoC core should schedule this command frequently. This method is used if ZQCTL0.dis_auto_zq is set to 1. In LPDDR4 mode, the MC sends MPC-based ZQ calibration commands (ZQCal Start and ZQCal Latch) instead of ZQCS command and it is recommended not to set DBGCMD.zq_calib_short signal in init or SR-Powerdown or Deep Power Down operating mode (this can be checked by observing STAT.operating_mode and STAT.selfref_state).

  For self-refresh (except LPDDR4) or SR-Powerdown(LPDDR4), command is scheduled after SR/SR-Powerdown is exited. For Deep Power Down or MPSM, command is not scheduled although ZQSTAT.zq_calib_short_busy is de-asserted.

36.9.5 LPDDR4 ZQ reset command

In LPDDR4 mode, it is recommended not to set this signal in init or SR-Powerdown or Deep Power Down operating modes (this can be checked by observing STAT.operating_mode and STAT.selfref_state). The SoC core can initiate a ZQ Reset operation only if ZQSTAT.zq_reset_busy is low. This signal goes high in the clock after the MC accepts a ZQ Reset request. It goes low when the ZQ reset command is issued to the DRAM and the associated NOP period is completed.

For self-refresh (except LPDDR4) or SR-Powerdown(LPDDR4), command is scheduled after SR/SR-Powerdown is exited. For Deep Power Down, command is not scheduled although ZQSTAT.zq_reset_busy is de-asserted.

36.10 MRR/MPR specific sequences

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

36.10.1 Mode register read and write operations

This section explains how to perform mode register reads and writes through software. Mode Register Reads (MRR) are applicable only to LPDDR4 and are used to read configuration and status data from mode registers in the SDRAM. Mode Register Writes (MRW or MRS) are applicable to all supported DDR protocols and are used to write configuration data to mode registers in the SDRAM. For DDR4 , the uMCTL2 also supports Multi-Purpose Register (MPR) reads and writes.

Access to the mode register is initiated by programming the MRCTRL0 and MRCTRL1 registers. It must be done in following steps:

1. Poll mr_wr_busy[mr_wr_busy] until it is 0. This checks that there is no outstanding MR transaction. No writes should be performed to MRCTRL0 and MRCTRL1, if mr_wr_busy[mr_wr_busy] =1.
2. Write the mr_type[mr_type], mr_addr[mr_addr], mr_rank[mr_rank] and (for MRWs) mr_data[mr_data] to define the MR transaction.
3. In a separate APB transaction, write the mr_wr[mr_rank] to 1. This bit is self-clearing and triggers the MR transaction. The uMCTL2 then asserts the mr_wr_busy[mr_wr_busy], while it performs the MR transaction to SDRAM and no further accesses can be initiated until it is de-asserted.

Note for Mode Register Write:

MRW transactions can be specified for either any single rank or any combination of serval ranks, by programming mr_rank[mr_rank]. If any part of the SDRAM's MR register is updated by software, it is also the responsibility of the software to update corresponding INIT*.*mr* so that it is aligned to the SDRAM's MR register.

Note for Mode Register Read:

MRR transactions should only be performed to one rank at a time, to avoid bus contention. When a MRR is performed, the mode register contents are available on hif_mrr_data, qualified by hif_mrr_data_valid, after the mode register read command is issued by the uMCTL2 to the SDRAM.

36.10.2 Multi-purpose register (MPR for DDR4 only)

This section describes how MPR Reads and MPR Writes are performed. The DDR4 SDRAMs contain four 8-bit programmable MPR which can be used for DQ training, CA parity log, MRS read out , or for vendor specific purposes. You can access MPRs when SDRAM is in MPR mode and the value of mpr_en[mpr_en] is 1.

Sequence of MPR Write operation:

1. Issue MRS command to SDRAM MR3 to put the SDRAM into MPR mode. The register mpr_en[mpr_en] must be 0. The MPR page selection, MR3[1:0] also be selected.
2. Wait until mr_wr_busy[mr_wr_busy] is 0. Write mr_data[mr_data].
3. Write MRCTRL0, where mr_addr[mr_addr] = MPR Location, mr_type[mr_type] = write , mr_wr[mr_wr] =1 and mpr_en[mpr_en] =1.
4. Issue MRS command to SDRAM MR3 to exit the SDRAM from MPR mode. The mpr_en[mpr_en] must be 0.

Sequence of MPR Read operation:

1. Issue MRS command to SDRAM MR3 to put the SDRAM into MPR mode. The register mpr_en[mpr_en] must be 0. The MPR page selection, MR3[1:0] and read format, MR3[12:11] must also be selected at this time.
2. Wait until mr_wr_busy[mr_wr_busy] is 0. Write mr_data[mr_data], where mr_data[mr_data][1:0]=00.
3. Write MRCTRL0, where mr_addr[mr_addr] = MPR Location, mr_type[mr_type] =read , mr_wr[mr_wr] =1 and mpr_en[mpr_en] =1. This causes the uMCTL2 to issue the MPR Read. The mode register contents are available on hif_mrr_data, qualified by hif_mrr_data_valid. When this data is received, it is safe to move to the next step.
4. Issue MRS command to SDRAM MR3 to exit the SDRAM from MPR mode. The register MRCTRL0.mpr_en must be 0.

36.11 ODT control

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

36.11.1 Overview of ODT control

By default, ODT to memories (dfi_odt) is driven to all zeros. The register inputs control the following:

- ODTMAP.rank*_wr_odt: The value desired for ODT following a write command.
- ODTCFG.wr_odt_delay: The number of cycles to delay following a write command before driving the programmed values for write ODT, which depends primarily on CAS latency.
- ODTCFG.wr_odt_hold: The number of cycles to hold the programmed write value after it is first driven.
- ODTMAP.rank*_rd_odt: The value desired for ODT following a read command.
- ODTCFG.rd_odt_delay: The number of cycles to delay following a read command before driving the programmed values for write ODT, which depends primarily on CAS latency.
- ODTCFG.rd_odt_hold: The number of cycles to hold the programmed read value after it is first driven.

NOTE

- ODT control for memories is not required for LPDDR4. It is a DDR3/DDR4 specific feature.
- For LPDDR4, set ODTMAP.rank*_wr/rd_odt to all zero because the ODT feature is automatically turned ON/OFF. In addition, the DRAMTMG13.odtloff register field must be appropriately programmed.
- DDR4 Dynamic ODT mode is not supported by the MC.

  All of these must be set by the SoC core before taking the MC out of reset. They are then applied to every read or write issued by the MC. To avoid overlap of ODT for reads and writes, the following constraints must be enforced:

1:2 Frequency Ratio Mode:

- ODTCFG.wr_odt_hold <= 2 * RANKCTL.diff_rank_wr_gap + MSTR.burst_rdwr
  - If burst chop is enabled (MSTR.burstchop = 1), use burst_rdwr/2 in the above equation
- ODTCFG.rd_odt_hold <= 2*RANKCTL.diff_rank_rd_gap + MSTR.burst_rdwr
  - If burst chop is enabled (MSTR.burstchop = 1), use burst_rdwr/2 in the above equation
- ODTCFG.wr_odt_delay + ODTCFG.wr_odt_hold <= ODTCFG.rd_odt_delay + 2 * DRAMTMG2.wr2rd
  - For DDR4, use wr2rd_s in the above equation
- ODTCFG.rd_odt_delay + ODTCFG.rd_odt_hold <= ODTCFG.wr_odt_delay + 2 * DRAMTMG2.rd2wr
- ODTCFG.wr_odt_delay >= DFITMG1.dfi_t_cmd_lat (DDR4 only)

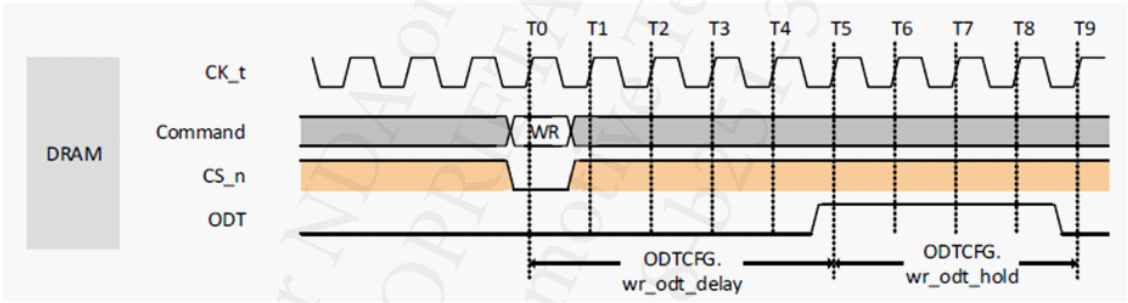


Figure 186. WR ODT timing diagram

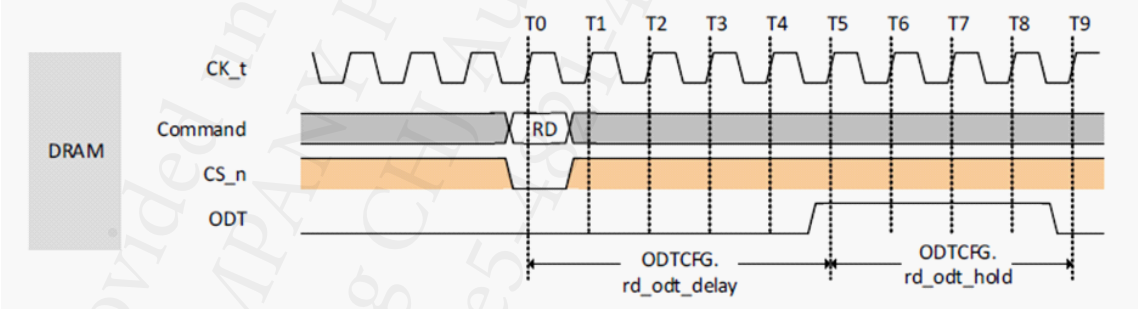


Figure 187. RD ODT timing diagram

36.12 Programming considerations

This section and all its sub-sections are Synopsys Proprietary. Used with permission.

This section provides information about how to program certain types of registers and fields.

36.12.1 Programming fields based on multiples of DFI clock cycles

The MC contains three timers that issue pulses at these multiples of the DFI clock:

- Every 32 clock cycles
- Every 1024 clock cycles
- Every 4096 clock cycles

Some timing register field values are stated based on these multiples. These fields have "x32", "x1024", or "x4096" in the field name. For example, an "x32" field counts pulses of the 32-clock cycle timer. The periods tracked by these fields start without any guaranteed relation to the phase of the associated timer. Therefore, for "x32" fields:

- A value of 0 gives no delay.
- A value of 1 gives a 1 to 32 cycle delay.
- A value of 2 gives a 33 to 64 cycle delay, and so on.

For "x1024" fields:

- A value of 0 gives no delay.
- A value of 1 gives a 1 to 1024 cycle delay.
- A value of 2 gives a 1025 to 2048 cycle delay, and so on.

For "x4096" fields:

- A value of 0 gives no delay.
- A value of 1 gives a 1 to 4096 cycle delay.
- A value of 2 gives a 4097 to 8192 cycle delay, and so on.

For a minimum delay, the field value must be set to the smallest number for which the resulting delay is ALWAYS greater than or equal to the required delay. For a maximum delay or nominal refresh, the value must ensure that the delay is ALWAYS less than or equal to the required delay.

36.12.2 Converting times into clock cycles

Most of the DRAM timing register fields call for values stated in clock cycles. But the value in an DRAM datasheet may be stated in time, such µs, ns, or ps. These datasheet values are primarily minimum timings. Before using them to calculate a value to program into a field, divide the specified time by the DRAM clock period (tCK) and round any fraction up to the next integer value.

The exceptions to this rule are maximum timings, where the result should be rounded down rather than up. These maximum timings include:

- RFSHTMG[t_rfc_nom_x32]
- DRAMTMG0[t_ras_max]

Certain DRAM timing register fields clariy this by using the following functions in their descriptions:

- RoundUp (X/Y), which means that if X divided by Y does not result in a whole integer, round the result up to the next integer value. For example: RoundUp(1000/150) = 7, and RoundUp(1050/150) = 7.
- RoundDown (X/Y), which means that if X divided by Y does not result in a whole integer, round the result down to the next integer value. For example: RoundUp(1000/150) = 6, and RoundUp(1050/150) = 7.

36.12.3 Programming mode

Each field description specifies the programming mode for that field. The programming mode describes when the field can be written to. The options are:

Programming mode	Description
Static	Can be written only when the MC is in reset.
Dynamic	Can be written at any time during operation.
Quasi-dynamic Group 1	Can be written when the MC is in reset and when no read/write traffic is present at the DFI.
Quasi-dynamic Group 2	Can be written when the MC is in reset and when the MC is in the Self Refresh, Deep Power Down, or Maximum Power Saving Mode states.
Quasi-dynamic Group 3	Can be written when the MC is in reset and when the MC is empty (idle).
Quasi-dynamic Group 4	Can be written when the MC is in reset and depending on the frequency mode (MSTR[frequency_mode]).

NOTE

Some fields are programmable under more than one quasi-dynamic group.

If OCCAPCFG[occap_en] = 1, you must poll SWSTAT[sw_done_ack] until it becomes 0 before modifying quasi-dynamic registers.

36.12.4 Standby power state (LP3/IO Retention)

The LP3/IO Retention power state is a non-operational low power state. The standby power state has the following properties:

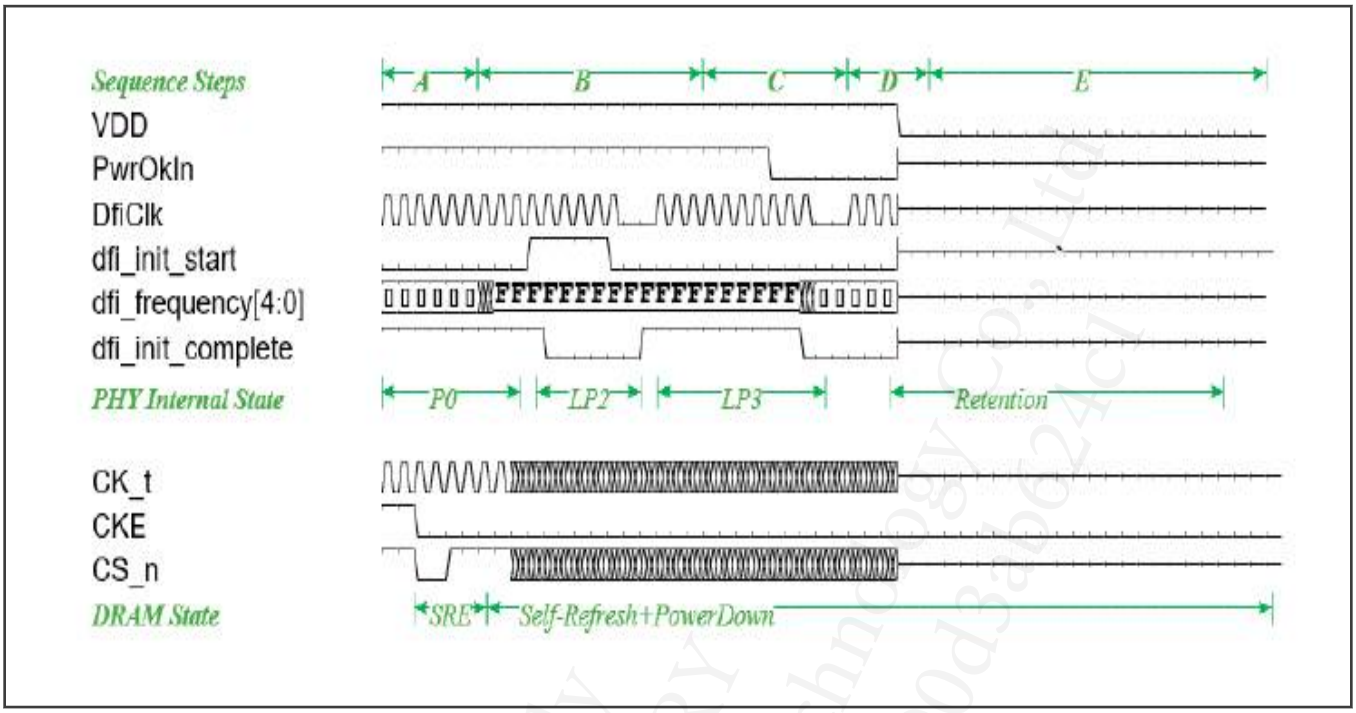
- This is the lowest possible power state for the PHY.
- In this power state, the DRAM may either be in self-refresh, or in Reset.
- The core VDD and VAA power supplies can be powered down in this state. If the core VDD power supply is shut off, power consumption is predominantly determined by the device leakage on VDDQ, the device leakage on VDDQLP, and the number of active CKE lanes.
- The state of CKE and MEMRESET pins will be pReserved at their last values when the core VDD is shut off. VDDQ must remain valid for the value of the pins to be Reserved.
- The Standby power state is entered via a Frequency Change sequence, using the dfi_init_start and dfi_freq[4:0] interface

NOTE

- When entering or exiting LP3/IO Retention, the DFI interface must be held in a valid state until PwrOkIn is set to 0. While in LP3/IO Retention, the DRAM signals required for proper IO Retention will maintain the appropriate state (including CKE and MEMRESET pins).
- The Memory controller may ignore the dfi_phyupd_req signals during LP3.
- When DfiClk stops(LP3 mode), then DFiCtlClk can also be clock gated or stopped.

The phy LP3 state is for deep sleep, in which the DFICLK can be gated; and additionally the PHY PLL is placed into maximum Powerdown mode. The power-saving of LP3 are only in the reduction of the standby current of the PHY PLL. However, the exit latency from this state is estimated at 35uS (since the PHY PLL requires a full initialization after entering Powerdown).

Preliminary Information
COMPANY CONFIDENTIAL
Beijing Autonomous Technology Co., Ltd.
c8c462_2021-04-26-10-51-39-0a1ab6a4a1



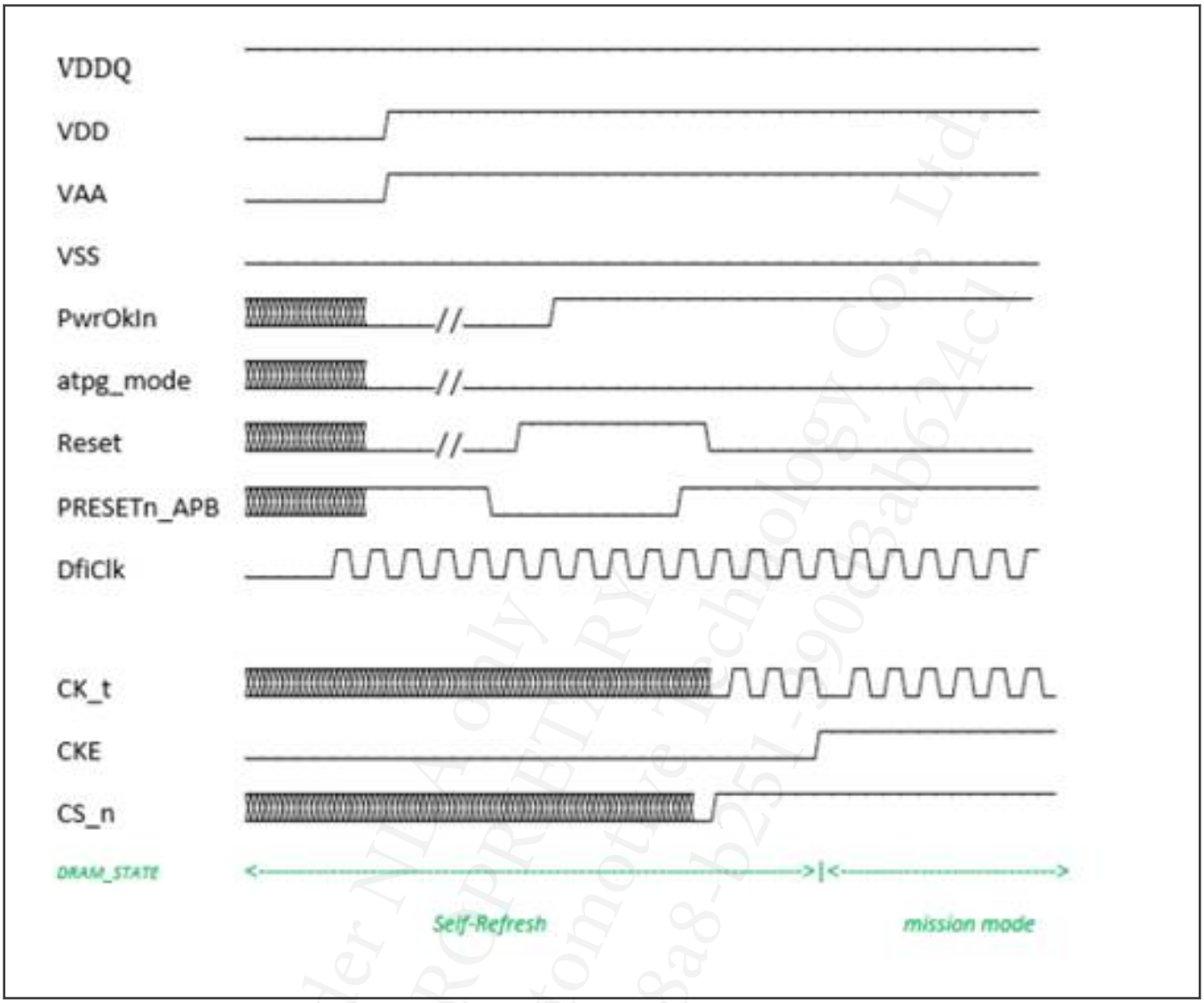
Sequence of LP3/ IO Retention Enter steps:

- Use the DFI interface to put the DRAM into Self-Refresh or Reset.
- Transition the PHY to the LP3 / IO Retention state by using a DFI Frequency Change operation.
- Set the PwrOkIn signal to 0. This will enable the data retention feature on the CKE and MEMRESET pins. These pins will hold the current values.
  - Remaining PHY input signals must be valid, inactive states for at least 10ns after the PwrOkIN de-asserting edge.
  - The PHY input clocks are not required to toggle after the de-asserting edge of PowerOkIN.
- Turn off the core VDD and VAA supplies. The VDDQ and VDDQLP power supply must remain active during LP3/IO Retention.
- All PHY input values are now don't cares and can go to X in simulations.

Sequence of LP3/IO Retention Exit steps:

- The VDDQ and VDDQLP power supply will already be active, restore the core VDD and VAA power supplies.
- Start clocks and reset the PHY
- Initialize the PHY configuration
- Reload the save training state
- Load the PHY Initialization Engine image
- Initialize the PHY to mission mode through a DFI initialization request
- The PHY is ready for mission-mode transactions.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e_482148812431390d8b624c1



The following requirements must also be addressed when exiting LP3/IO Retention mode:

- For retention exit, all rules regarding resetting the PHY and enabling the clocks apply a standard reset sequence.
- Since the CKE* and MEMRESET pin state must be protected , special care must be taken to ensure that the following signals:
  - Atpg_mode = 1’b0
  - PwrOkIn = 1’b0
- The (BypassModeEn*, WRSTN) signals may be defined at VDD power-on, but must be driven to Zero at least 10ns prior to the asserting edge of PwrOklin.

36.13 UMCTL2_MP register descriptions

36.13.1 UMCTL2_MP memory map

This section and all its sub-sections are Synopsys Proprietary. Used with permission. DDR MC Multi-Port Registers

UMCTL2_MP base address: 403C_03F8h

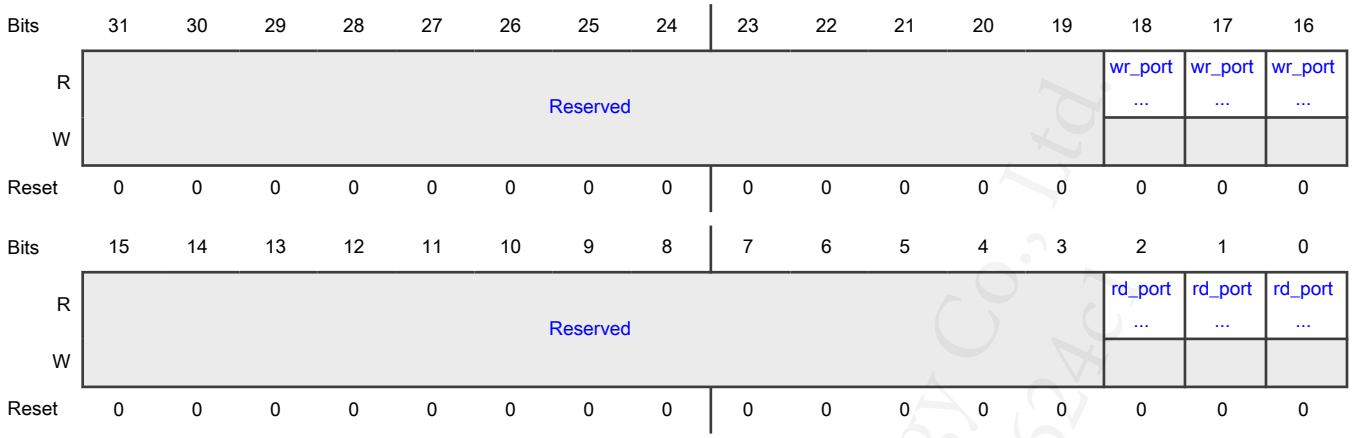
Offset	Register	Width (In bits)	Access	Reset value
4h	Port Status (PSTAT)	32	RO	0000_0000h
8h	Port Common Configuration (PCCFG)	32	RW	0000_0000h
Ch	Port n Configuration Read (PCFGR_0)	32	RW	0000_0000h
10h	Port n Configuration Write (PCFGW_0)	32	RW	0000_4000h
98h	Port n Control (PCTRL_0)	32	RW	0000_0000h
9Ch	Port n Read QoS Configuration Register 0 (PCFGQOS0_0)	32	RW	0200_0E00h
A0h	Port n Read QoS Configuration Register 1 (PCFGQOS1_0)	32	RW	0000_0000h
A4h	Port n Write QoS Configuration Register 0 (PCFGWQOS0_0)	32	RW	0000_0E00h
A8h	Port n Write QoS Configuration Register 1 (PCFGWQOS1_0)	32	RW	0000_0000h
BCh	Port n Configuration Read (PCFGR_1)	32	RW	0000_0000h
C0h	Port n Configuration Write (PCFGW_1)	32	RW	0000_4000h
148h	Port n Control (PCTRL_1)	32	RW	0000_0000h
14Ch	Port n Read QoS Configuration Register 0 (PCFGQOS0_1)	32	RW	0200_0E00h
150h	Port n Read QoS Configuration Register 1 (PCFGQOS1_1)	32	RW	0000_0000h
154h	Port n Write QoS Configuration Register 0 (PCFGWQOS0_1)	32	RW	0000_0E00h
158h	Port n Write QoS Configuration Register 1 (PCFGWQOS1_1)	32	RW	0000_0000h
16Ch	Port n Configuration Read (PCFGR_2)	32	RW	0000_0000h
170h	Port n Configuration Write (PCFGW_2)	32	RW	0000_4000h
1F8h	Port n Control (PCTRL_2)	32	RW	0000_0000h
1FCh	Port n Read QoS Configuration Register 0 (PCFGQOS0_2)	32	RW	0200_0E00h
200h	Port n Read QoS Configuration Register 1 (PCFGQOS1_2)	32	RW	0000_0000h
204h	Port n Write QoS Configuration Register 0 (PCFGWQOS0_2)	32	RW	0000_0E00h
208h	Port n Write QoS Configuration Register 1 (PCFGWQOS1_2)	32	RW	0000_0000h
B2Ch	Scrubber Control (SBRCTL)	32	RW	0000_FF10h
B30h	Scrubber Status (SBRSTAT)	32	RO	0000_0000h
B34h	Scrubber Write Data Pattern0 (SBRWDATA0)	32	RW	0000_0000h

36.13.2 Port Status (PSTAT)

Offset

Register	Offset
PSTAT	4h

Diagram



Fields

Field	Function
31-19 —	Reserved
18 wr_port_busy_2	Indicates if there are outstanding writes for AXI port 2. Programming Mode: Dynamic
17 wr_port_busy_1	Indicates if there are outstanding writes for AXI port 1. Programming Mode: Dynamic
16 wr_port_busy_0	Indicates if there are outstanding writes for AXI port 0. Programming Mode: Dynamic
15-3 —	Reserved
2 rd_port_busy_2	Indicates if there are outstanding reads for AXI port 2. Programming Mode: Dynamic
1 rd_port_busy_1	Indicates if there are outstanding reads for AXI port 1. Programming Mode: Dynamic
0 rd_port_busy_0	Indicates if there are outstanding reads for AXI port 0. Programming Mode: Dynamic

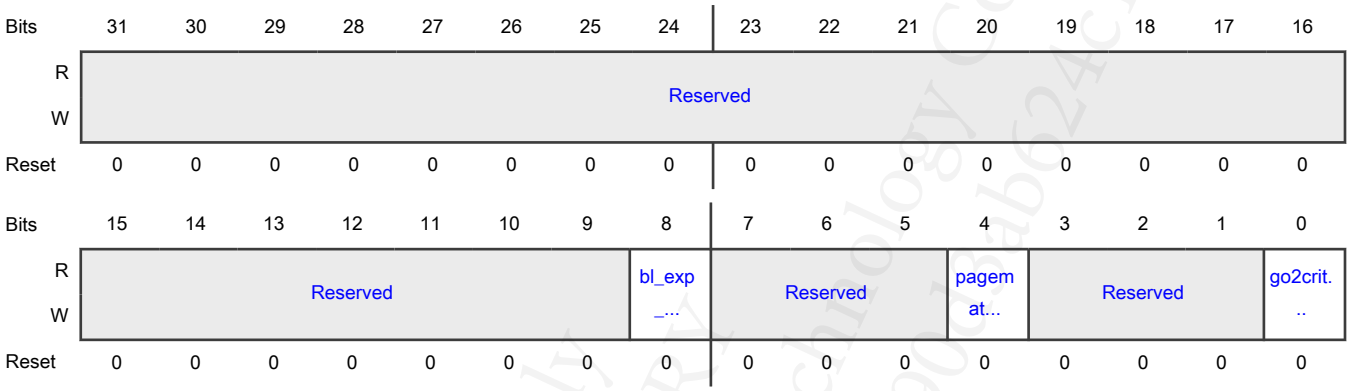
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e1-4212-48a8-b251-190d3ab624c

36.13.3 Port Common Configuration (PCCFG)

Offset

Register	Offset
PCCFG	8h

Diagram



Fields

Field	Function
31-9 —	Reserved
8 bl_exp_mode	Burst expansion mode Selects half of the memory burst length (BL) as the XPI burst expansion packet length. Applies to both read and write expansion. When using Partial Writes (UMCTL2_PARTIAL_WR == 1), half BL burst expansion avoids or minimizes the t_ccd_l penalty for DDR4 or the t_ccd_mw penalty for LPDDR4. Therefore, you should use it only for DDR4 or LPDDR4 DRAM. When not using data mask (DBICTL[dm_en] == 0), the MC does not support half BL burst expansion if any of these conditions are true: • Partial Writes disabled (UMCTL2_PARTIAL_WR == 0) • Partial Writes enabled (UMCTL2_PARTIAL_WR == 1) AND — For LPDDR4, all of these conditions are true: ◦ Half bus width (MSTR[data_bus_width] == 01) ◦ MC BL8 (MEMC_BURST_LENGTH == 8) ◦ DRAM BL16 (MSTR[burst_rdwr] == 1000) — For DDR4, all of these conditions are true:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	◦ Half bus width (MSTR[data_bus_width] == 01) ◦ MC BL4 (MEMC_BURST_LENGTH == 4) ◦ DRAM BL8 (MSTR[burst_rdwr] == 0100) ◦ Either no burst chop (MSTR[burstchop] == 0) OR CRC enabled (CRCPARCTL1[crc_enable] == 1) This field has no effect when using full data bus width (MSTR[data_bus_width] == 00) When using Data Channel Interleave, the MC does not support burst expansion. Programming Mode: Static 0b - Full BL 1b - Half BL
7-5 —	Reserved
4 pagematch_limit	Page match four limit If Page Match is enabled (PCFGR_0[rd_port_pagematch_en]), limits the number of consecutive same-page MC transactions granted by the Port Arbiter to four. Programming Mode: Static 0b - No limit 1b - Limit of four
3-1 —	Reserved
0 go2critical_en	Go to critical enable Enables go-to-critical signaling to MC based on urgent input from the AXI master. Programming Mode: Static 0b - Disable 1b - Enable

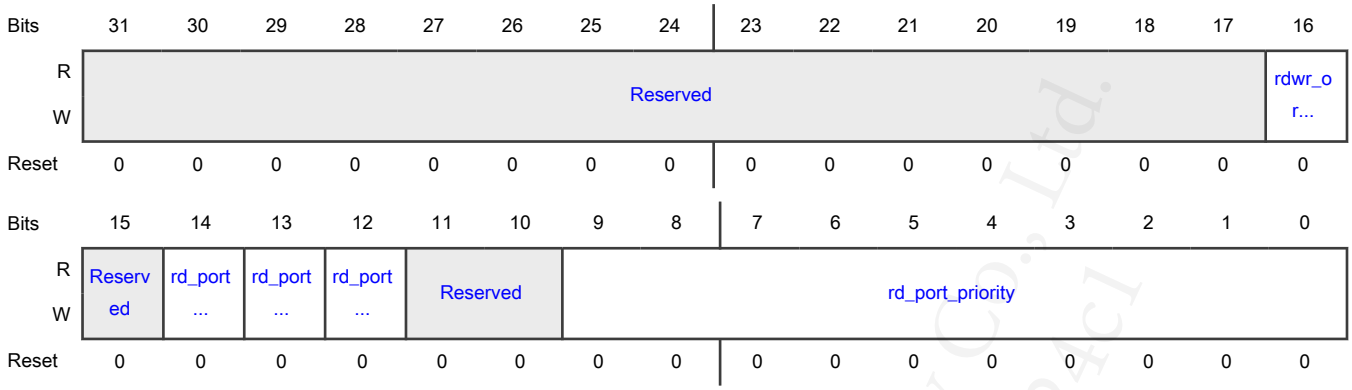
36.13.4 Port n Configuration Read (PCFGR_0)

Offset

Register	Offset
PCFGR_0	Ch

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-b251-890d3ab624c1

Diagram



Fields

Field	Function
31-17 —	Reserved
16 rdwr_ordered_en	Ordered read/writes enable Enables ordered read/writes. If enabled, the MC preserves the order of read and write transactions issued to the same address on a given port. In other words, the MC transports all same-address read and write commands from the application port interface to the DFI interface in the order of acceptance. This feature helps to maintain software coherency for masters that issue back-to-back read/write transactions without waiting for write/read responses. Programming Mode: Static> 0b - Disable 1b - Enable
15 —	Reserved
14 rd_port_pagematch_en	Read Page Match enable Enables read Page Match. If enabled, when the port arbiter grants a requesting port, that port remains granted if the following immediate commands are to the same memory page (same bank and row). See PCCFG[pagematch_limit] for information about limiting the number of consecutive transactions under Page Match. Programming Mode: Static 0b - Disable 1b - Enable
13 rd_port_urgent_en	Read AXI urgent sideband enable

Table continues on the next page...

Table continued from the previous page...

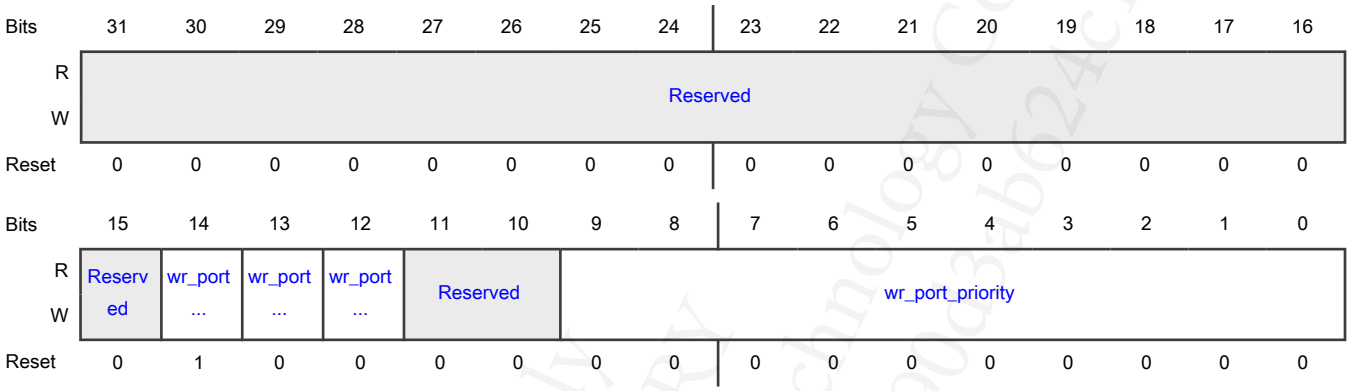
Field	Function
	Enables the AXI urgent sideband signal (arurgent). If enabled and the master asserts arurgent, that port becomes the highest priority port. Also, if enabled and go to critical is enabled (PCCFG[go2critical_en]), the MC asserts co_gs_go2critical_lpr or co_gs_go2critical_hpr to the MC. **NOTE** The master can assert arurgent any time and for as long as needed, independent of address handshaking. It is not associated with any particular command. Programming Mode: Static 0b - Disable 1b - Enable
12 rd_port_aging_en	Read aging enable Enables the aging function for the port read channel. Programming Mode: Static 0b - Disable 1b - Enable
11-10 —	Reserved
9-0 rd_port_priority	Read aging counter value Specifies the initial value to be loaded into the read aging counters. The MC loads the counters with this value after reset or after each grant to the corresponding port. The aging counters count down every clock cycle that a port is requested but not granted. The five most significant bits (msb) of the counter sets the priority of the read channel for a particular port. That port's priority increases as the five msb decrease. When the aging counter reaches zero, the corresponding port channel has the highest priority (timeout condition – Priority0). For multi-port configurations, if using external dynamic priority inputs (arqos), you cannot use the aging counters to set port priorities (timeout is still applicable). For single-port configurations, the aging counters only matter to force read-write direction switching when they timeout (become 0). In this case, you can still use the external dynamic priority input (arqos), for reads only, to set the MC read priority: low priority read (LPR) or high priority read (HPR) on a command-by-command basis. **NOTE** The two least-significant bits (lsb) of this field are tied low (00b) internally. Programming Mode: Static

36.13.5 Port n Configuration Write (PCFGW_0)

Offset

Register	Offset
PCFGW_0	10h

Diagram



Fields

Field	Function
31-15 —	Reserved
14 wr_port_pagematch_en	Write Page Match enable Enables write Page Match. If enabled, when the port arbiter grants a requesting port, that port remains granted if the following immediate commands are to the same memory page (same bank and row). See PCCFG[pagematch_limit] for information about limiting the number of consecutive transactions under Page Match. Programming Mode: Static 0b - Disable 1b - Enable
13 wr_port_urgent_en	Write AXI urgent sideband enable Enables the AXI urgent sideband signal (awurgent). If enabled and the master asserts awurgent, that port becomes the highest priority port. Also, if enabled and go to critical is enabled (PCCFG[go2critical_en]), the MC asserts co_gs_go2critical_wr to the MC. **NOTE** The master can assert awurgent any time and for as long as needed, independent of address handshaking. It is not associated with any particular command. Programming Mode: Static

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
12 wr_port_aging_en	Write aging enable Enables the aging function for the port write channel. Programming Mode: Static 0b - Disable 1b - Enable
11-10 —	Reserved
9-0 wr_port_priority	Write aging counter value Specifies the initial value to be loaded into the Write aging counters. The MC loads the counters with this value after reset or after each grant to the corresponding port. The aging counters count down every clock cycle that a port is requested but not granted. The five most significant bits (msb) of the counter sets the priority of the write channel for a particular port. That port's priority increases as the five msb decrease. When the aging counter reaches zero, the corresponding port channel has the highest priority (timeout condition – Priority0). For multi-port configurations, if using external dynamic priority inputs (awqos), you cannot use the aging counters to set port priorities (timeout is still applicable). For single-port configurations, the aging counters only matter to force read-write direction switching when they timeout (become 0). In this case, you can still use the external dynamic priority input (arqos), for reads only, to set the MC read priority: low priority read (LPR) or high priority read (HPR) on a command-by-command basis. **NOTE** The two least-significant bits (lsb) of this field are tied low (00b) internally. Programming Mode: Static

36.13.6 Port n Control (PCTRL_0)

Offset

Register	Offset
PCTRL_0	98h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															port_en
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 port_en	Enables AXI port n. Programming Mode: Dynamic

36.13.7 Port n Read QoS Configuration Register 0 (PCFGQOS0_0)

Offset

Register	Offset
PCFGQOS0_0	9Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved						rqos_map_region2		Reserved		rqos_map_region1		Reserved		rqos_map_region0	
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				rqos_map_level2				Reserved				rqos_map_level1			
Reset	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHX Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b213-90d3ab624c1

Fields

Field	Function
31-26 —	Reserved
25-24 rqos_map_region2	Region 2 traffic class Specifies the traffice class for region 2. For dual-address-queue configurations, region 1 maps to the red address queue. If VPR support is disabled (UMCTL2_VPR_EN == 0) and the region 2 traffic class is VPR, the MC aliases VPR traffic to LPR traffic. Programming Mode: Quasi-dynamic Group 3 00b - Invalid 01b - VPR 10b - HPR (not value for dual-address-queue configurations) 11b - Invalid
23-22 —	Reserved
21-20 rqos_map_region1	Region 1 traffic class Specifies the traffice class for region 1. For dual-address-queue configurations, region 1 maps to the blue address queue. If VPR support is disabled (UMCTL2_VPR_EN == 0) and the region 1 traffic class is VPR, the MC aliases VPR traffic to LPR traffic. Programming Mode: Quasi-dynamic Group 3 00b - LPR 01b - VPR 10b - HPR (not valid for dual-address-queue configurations) 11b - Invalid
19-18 —	Reserved
17-16 rqos_map_region0	Read region 0 traffic class Specifies the traffice class for region 0. For dual-address-queue configurations, region 0 maps to the blue address queue. If VPR support is disabled (UMCTL2_VPR_EN == 0) and the region 0 traffic class is VPR, the MC aliases VPR traffic to LPR traffic. Programming Mode: Quasi-dynamic Group 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - LPR 01b - VPR 10b - HPR (not valid for dual-address-queue configurations) 11b - Invalid
15-12 —	Reserved
11-8 rqos_map_level2	Read separation level 2 Specifies the separation level 2, which indicates the end of region1 mapping. Region2 starts from (level1 + 1) up to 15. Possible values are (level1 + 1) to 14, which corresponds to arqos. **NOTE** For PA, the MC uses arqos values directly as port priorities, where the higher the value, the higher the port priority. You must set all map_level* registers to distinct values. Programming Mode: Quasi-dynamic Group 3
7-4 —	Reserved
3-0 rqos_map_level1	Read separation level 1 Specifies the separateion level 1, which indicates the end of region0 mapping. Region0 starts at zero. Possible values are 0 to 13 for dual RAQ or 0 to 14 for single RAQ, which corresponds to arqos. **NOTE** For PA, the MC uses arqos values directly as port priorities, where the higher the value, the higher the port priority. You must set all map_level* registers to distinct values. Programming Mode: Quasi-dynamic Group 3

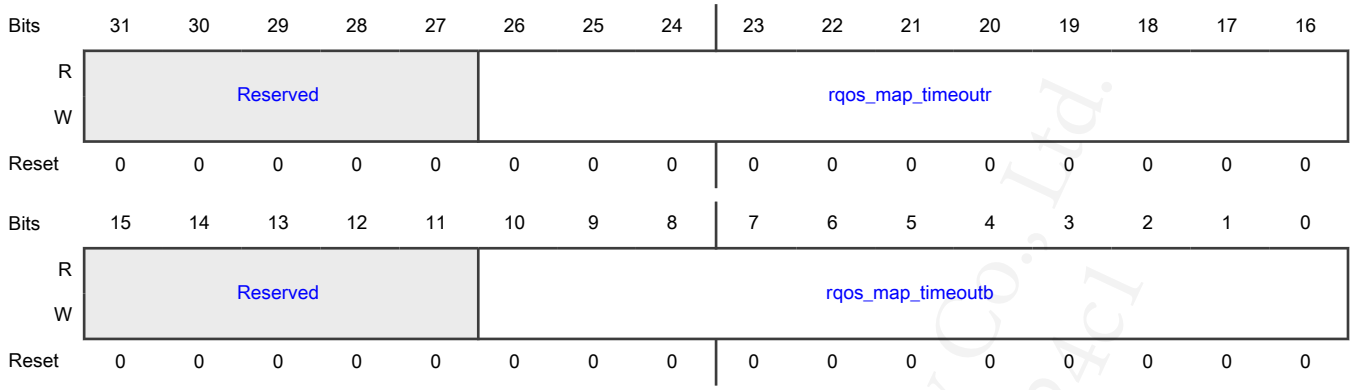
36.13.8 Port n Read QoS Configuration Register 1 (PCFGQOS1_0)

Offset

Register	Offset
PCFGQOS1_0	A0h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4221-4ae6-b251-39da3a1624c1

Diagram



Fields

Field	Function
31-27 —	Reserved
26-16 rqos_map_timeoutr	Read red address queue timeout interval Specifies the timeout interval for transactions mapped to the red address queue. Programming Mode: Quasi-dynamic Group 3
15-11 —	Reserved
10-0 rqos_map_timeoutb	Read blue address queue timeout interval Specifies the timeout interval for transactions mapped to the blue address queue. Programming Mode: Quasi-dynamic Group 3

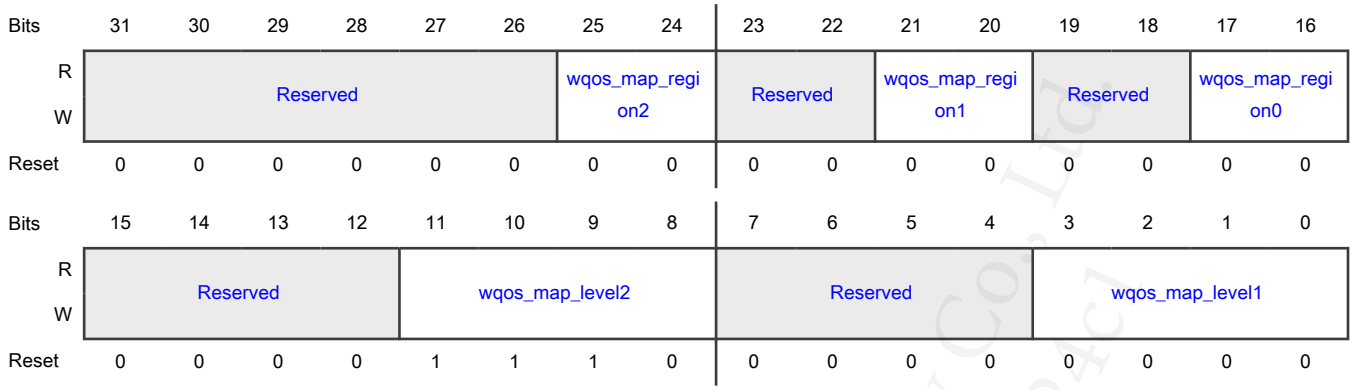
36.13.9 Port n Write QoS Configuration Register 0 (PCFGWQOS0_0)

Offset

Register	Offset
PCFGWQOS0_0	A4h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a2-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24 wqos_map_region2	Write region 2 traffic class Specifies the traffice class for region 2. If VPW support is disabled (UMCTL2_VPW_EN == 0) and the region 2 traffic class is VPW, the MC aliases VPW traffic to NPW traffic. Programming Mode: Quasi-dynamic Group 3 00b - LPR 01b - VPR 10b - Invalid 11b - Invalid
23-22 —	Reserved
21-20 wqos_map_region1	Write region 1 traffic class Specifies the traffice class for region 1. If VPW support is disabled (UMCTL2_VPW_EN == 0) and the region 1 traffic class is VPW, the MC aliases VPW traffic to NPW traffic. Programming Mode: Quasi-dynamic Group 3 00b - LPR 01b - VPR 10b - Invalid 11b - Invalid
19-18	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
17-16 wqos_map_region0	Write region 0 traffic class Specifies the traffice class for region 0. If VPW support is disabled (UMCTL2_VPW_EN == 0) and the region 0 traffic class is VPW, the MC aliases VPW traffic to NPW traffic. Programming Mode: Quasi-dynamic Group 3 00b - LPR 01b - VPR 10b - Invalid 11b - Invalid
15-12 —	Reserved
11-8 wqos_map_level2	Write separation level 2 Specifies the separation level 2, which indicates the end of region 1 mapping. Region2 starts from (level1 + 1) up to 15. Possible values are (level1 + 1) to 14, which corresponds to awqos. **NOTE** For PA, the MC uses awqos values directly as port priorities, where the higher the value, the higher the port priority. You must set all map_level* registers to distinct values. Programming Mode: Quasi-dynamic Group 3
7-4 —	Reserved
3-0 wqos_map_level1	Write separation level 1 Specifies the separateion level 1, which indicates the end of region 0 mapping. Region 0 starts at zero. Possible values are 0 to 13 for dual RAQ or 0 to 14 for single RAQ, which corresponds to awqos. **NOTE** For PA, the MC uses awqos values directly as port priorities, where the higher the value, the higher the port priority. You must set all map_level* registers to distinct values. Programming Mode: Quasi-dynamic Group 3

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIC Automotive Technology Co., Ltd.
c8c4b2e1_43b1_48a8_9213_90d3ab624c1

36.13.10 Port n Write QoS Configuration Register 1 (PCFGWQOS1_0)

Offset

Register	Offset
PCFGWQOS1_0	A8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved					wqos_map_timeout2										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					wqos_map_timeout1										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-27 —	Reserved
26-16 wqos_map_timeout2	Write region 2 timeout interval Specifies the timeout interval for write transactions in region 2. Programming Mode: Quasi-dynamic Group 3
15-11 —	Reserved
10-0 wqos_map_timeout1	Write region 0 and 1 timeout interval Specifies the timeout interval for write transactions in regions 0 and 1. Programming Mode: Quasi-dynamic Group 3

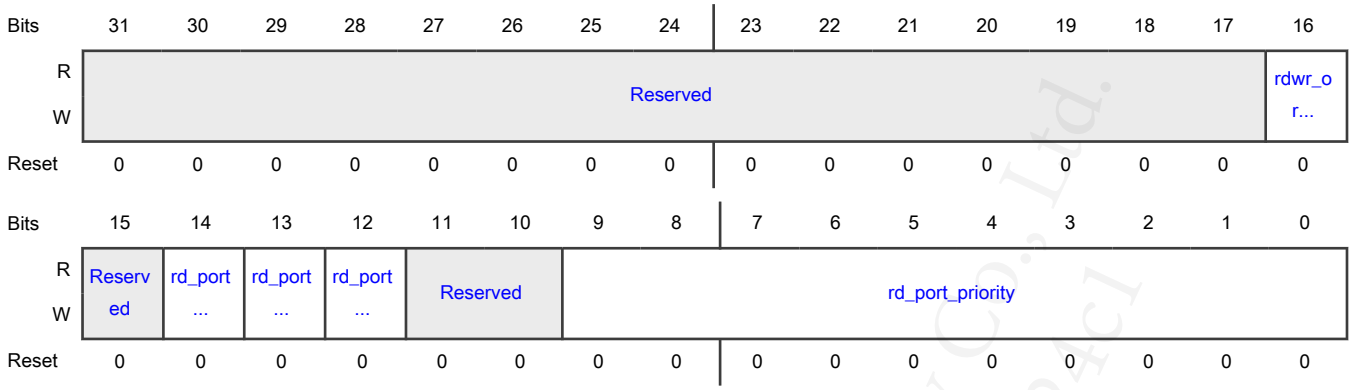
36.13.11 Port n Configuration Read (PCFGR_1)

Offset

Register	Offset
PCFGR_1	BCh

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESI Automotive Technology Co., Ltd.
c8c46225-4821-4858-b251-390a34b624c1

Diagram



Fields

Field	Function
31-17 —	Reserved
16 rdwr_ordered_en	Enable ordered read/writes. If set to 1, preserves the ordering between read transaction and write transaction issued to the same address, on a given port. In other words, the controller ensures that all same address read and write commands from the application port interface are transported to the DFI interface in the order of acceptance. This feature is useful in cases where software coherency is desired for masters issuing back-to-back read/write transactions without waiting for write/read responses. Note that this register has an effect only if necessary logic is instantiated via the UMCTL2_RDWR_ORDERED_n parameter. Programming Mode: Static
15 —	Reserved
14 rd_port_pagematch_en	If set to 1, enables the Page Match feature. If enabled, once a requesting port is granted, the port is continued to be granted if the following immediate commands are to the same memory page (same bank and same row). See also related PCCFG.pagematch_limit register. Programming Mode: Static
13 rd_port_urgent_en	If set to 1, enables the AXI urgent sideband signal (arurgent). When enabled and arurgent is asserted by the master, that port becomes the highest priority and co_gs_go2critical_lpr/co_gs_go2critical_hpr signal to MC is asserted if enabled in PCCFG.go2critical_en register. Note that arurgent signal can be asserted anytime and as long as required which is independent of address handshaking (it is not associated with any particular command). Programming Mode: Static
12 rd_port_aging_en	If set to 1, enables aging function for the read channel of the port. Programming Mode: Static
11-10 —	Reserved
9-0	Determines the initial load value of read aging counters. These counters will be parallel loaded after reset, or after each grant to the corresponding port. The aging counters down-count every clock cycle

Table continues on the next page...

Table continued from the previous page...

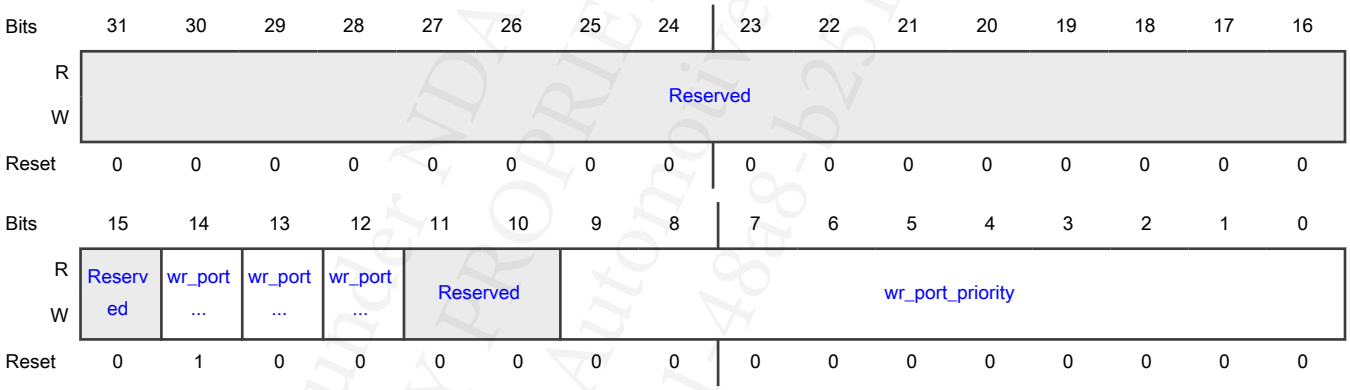
Field	Function
rd_port_priority	where the port is requesting but not granted. The higher significant 5-bits of the read aging counter sets the priority of the read channel of a given port. Port's priority will increase as the higher significant 5-bits of the counter starts to decrease. When the aging counter becomes 0, the corresponding port channel will have the highest priority level (timeout condition - Priority0). For multi-port configurations, the aging counters cannot be used to set port priorities when external dynamic priority inputs (arqos) are enabled (timeout is still applicable). For single port configurations, the aging counters are only used when they timeout (become 0) to force read-write direction switching. In this case, external dynamic priority input, arqos (for reads only) can still be used to set the MC read priority (2 priority levels: low priority read - LPR, high priority read - HPR) on a command by command basis. Note: The two LSBs of this register field are tied internally to 2'b00. Programming Mode: Static

36.13.12 Port n Configuration Write (PCFGW_1)

Offset

Register	Offset
PCFGW_1	C0h

Diagram



Fields

Field	Function
31-15 —	Reserved
14 wr_port_pagematch_en	If set to 1, enables the Page Match feature. If enabled, once a requesting port is granted, the port is continued to be granted if the following immediate commands are to the same memory page (same bank and same row). See also related PCCFG.pagematch_limit register. Programming Mode: Static

Table continues on the next page...

Table continued from the previous page...

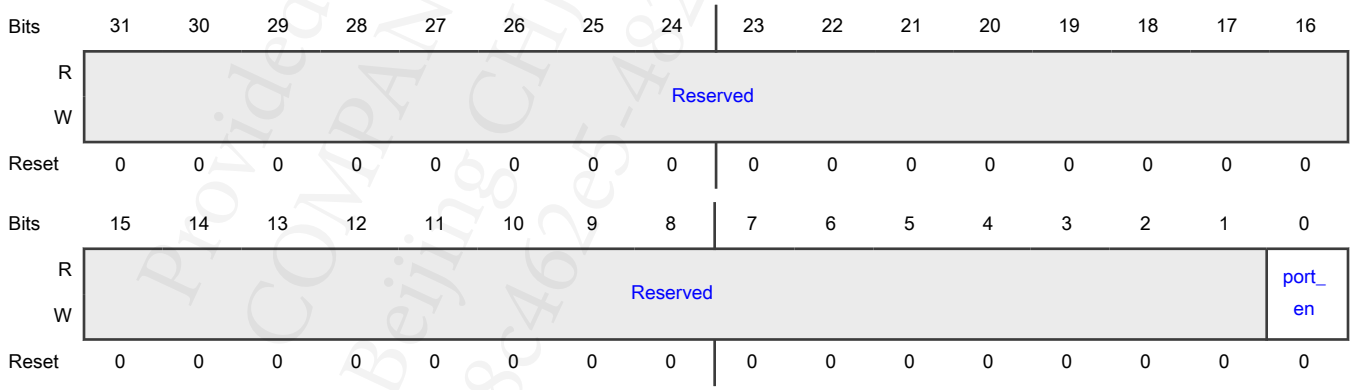
Field	Function
13 wr_port_urgent_en	If set to 1, enables the AXI urgent sideband signal (awurgent). When enabled and awurgent is asserted by the master, that port becomes the highest priority and co_gs_go2critical_wr signal to the MC is asserted if enabled in PCCFG.go2critical_en register. Note that awurgent signal can be asserted anytime and as long as required which is independent of address handshaking (it is not associated with any particular command). Programming Mode: Static
12 wr_port_aging_en	If set to 1, enables aging function for the write channel of the port. Programming Mode: Static
11-10 —	Reserved
9-0 wr_port_priority	Determines the initial load value of write aging counters. These counters will be parallel loaded after reset, or after each grant to the corresponding port. The aging counters down-count every clock cycle where the port is requesting but not granted. The higher significant 5-bits of the write aging counter sets the initial priority of the write channel of a given port. Port's priority will increase as the higher significant 5-bits of the counter starts to decrease. When the aging counter becomes 0, the corresponding port channel will have the highest priority level. For multi-port configurations, the aging counters cannot be used to set port priorities when external dynamic priority inputs (awqos) are enabled (timeout is still applicable). For single port configurations, the aging counters are only used when they timeout (become 0) to force read-write direction switching. Note: The two LSBs of this register field are tied internally to 2'b00. Programming Mode: Static

36.13.13 Port n Control (PCTRL_1)

Offset

Register	Offset
PCTRL_1	148h

Diagram



Fields

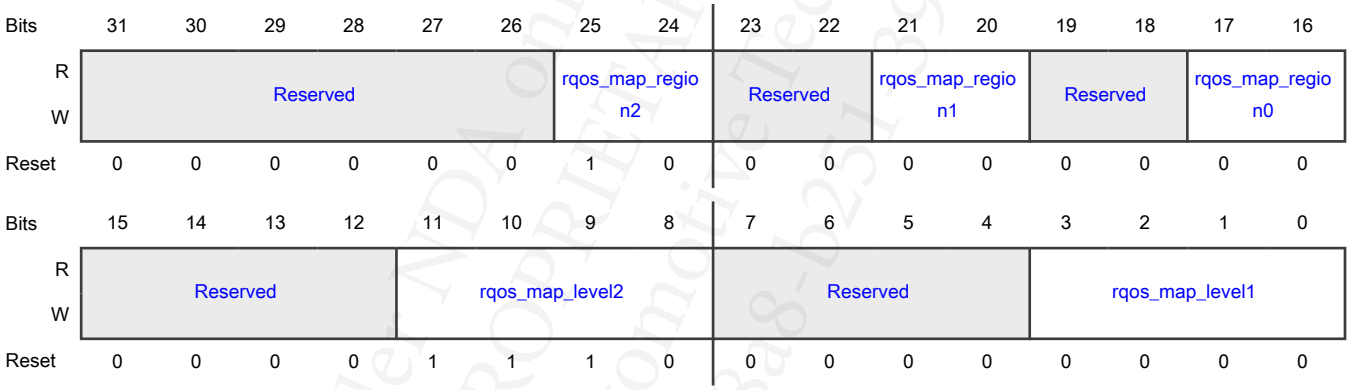
Field	Function
31-1 —	Reserved
0 port_en	Enables AXI port n. Programming Mode: Dynamic

36.13.14 Port n Read QoS Configuration Register 0 (PCFGQOS0_1)

Offset

Register	Offset
PCFGQOS0_1	14Ch

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24 rqos_map_region2	This bitfield indicates the traffic class of region2. For dual address queue configurations, region2 maps to the red address queue. Valid values are 1: VPR and 2: HPR only. When VPR support is disabled (UMCTL2_VPR_EN = 0) and traffic class of region2 is set to 1 (VPR), VPR traffic is aliased to LPR traffic. Programming Mode: Quasi-dynamic Group 3
23-22 —	Reserved
21-20	This bitfield indicates the traffic class of region 1. Valid values are: 0 : LPR, 1: VPR, 2: HPR. For dual address queue configurations, region1 maps to the blue address queue. In this case, valid values are 0:

Table continues on the next page...

Table continued from the previous page...

Field	Function
rqos_map_region1	LPR and 1: VPR only. When VPR support is disabled (UMCTL2_VPR_EN = 0) and traffic class of region 1 is set to 1 (VPR), VPR traffic is aliased to LPR traffic. Programming Mode: Quasi-dynamic Group 3
19-18 —	Reserved
17-16 rqos_map_region0	This bitfield indicates the traffic class of region 0. Valid values are: 0: LPR, 1: VPR, 2: HPR. For dual address queue configurations, region 0 maps to the blue address queue. In this case, valid values are: 0: LPR and 1: VPR only. When VPR support is disabled (UMCTL2_VPR_EN = 0) and traffic class of region0 is set to 1 (VPR), VPR traffic is aliased to LPR traffic. Programming Mode: Quasi-dynamic Group 3
15-12 —	Reserved
11-8 rqos_map_level2	Separation level2 indicating the end of region1 mapping; start of region1 is (level1 + 1). Possible values for level2 are (level1 + 1) to 14 which corresponds to arqos. Region2 starts from (level2 + 1) up to 15. Note that for PA, arqos values are used directly as port priorities, where the higher the value corresponds to higher port priority. All of the map_level* registers must be set to distinct values. Programming Mode: Quasi-dynamic Group 3
7-4 —	Reserved
3-0 rqos_map_level1	Separation level1 indicating the end of region0 mapping; start of region0 is 0. Possible values for level1 are 0 to 13 (for dual RAQ) or 0 to 14 (for single RAQ) which corresponds to arqos. Note that for PA, arqos values are used directly as port priorities, where the higher the value corresponds to higher port priority. All of the map_level* registers must be set to distinct values. Programming Mode: Quasi-dynamic Group 3

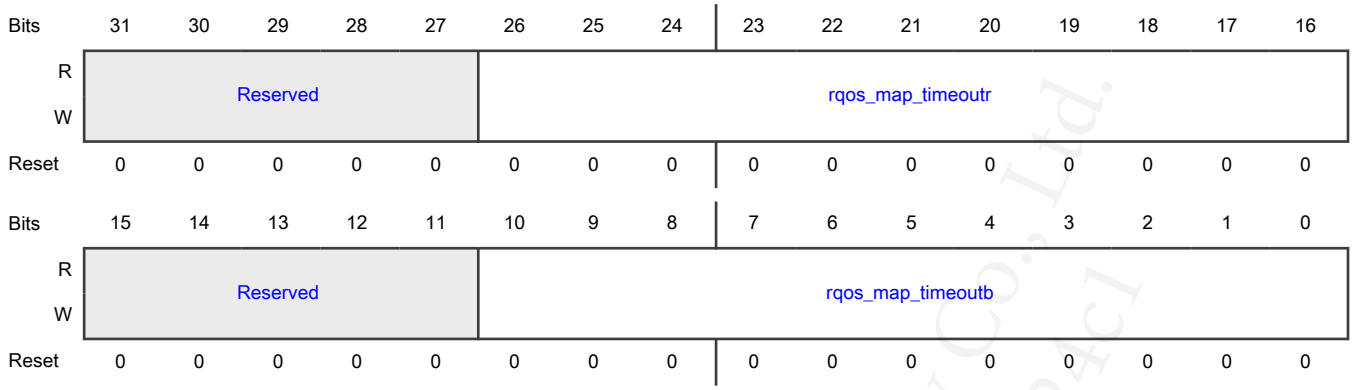
36.13.15 Port n Read QoS Configuration Register 1 (PCFGQOS1_1)

Offset

Register	Offset
PCFGQOS1_1	150h

Provided under NDA of COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-4821-48a8-9151-37e3ab6...

Diagram



Fields

Field	Function
31-27 —	Reserved
26-16 rqos_map_timeoutr	Specifies the timeout value for transactions mapped to the red address queue. Programming Mode: Quasi-dynamic Group 3
15-11 —	Reserved
10-0 rqos_map_timeoutb	Specifies the timeout value for transactions mapped to the blue address queue. Programming Mode: Quasi-dynamic Group 3

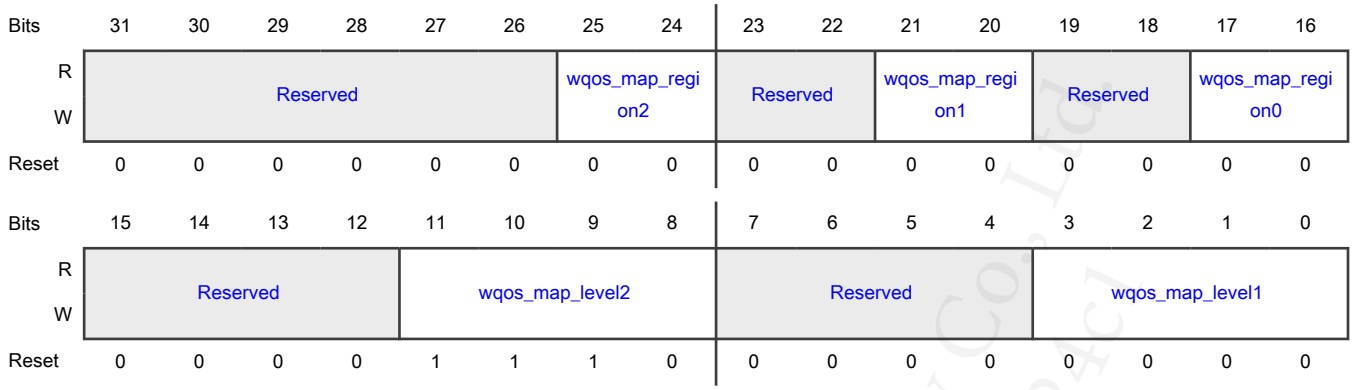
36.13.16 Port n Write QoS Configuration Register 0 (PCFGWQOS0_1)

Offset

Register	Offset
PCFGWQOS0_1	154h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-28a8-0251-330d3ab624c1

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24 wqos_map_region2	This bitfield indicates the traffic class of region 2. Valid values are: 0: NPW, 1: VPW. When VPW support is disabled (UMCTL2_VPW_EN = 0) and traffic class of region 2 is set to 1 (VPW), VPW traffic is aliased to NPW traffic. Programming Mode: Quasi-dynamic Group 3
23-22 —	Reserved
21-20 wqos_map_region1	This bitfield indicates the traffic class of region 1. Valid values are: 0: NPW, 1: VPW. When VPW support is disabled (UMCTL2_VPW_EN = 0) and traffic class of region 1 is set to 1 (VPW), VPW traffic is aliased to NPW traffic. Programming Mode: Quasi-dynamic Group 3
19-18 —	Reserved
17-16 wqos_map_region0	This bitfield indicates the traffic class of region 0. Valid values are: 0: NPW, 1: VPW. When VPW support is disabled (UMCTL2_VPW_EN = 0) and traffic class of region 0 is set to 1 (VPW), VPW traffic is aliased to NPW traffic. Programming Mode: Quasi-dynamic Group 3
15-12 —	Reserved
11-8 wqos_map_level2	Separation level2 indicating the end of region1 mapping; start of region1 is (level1 + 1). Possible values for level2 are (level1 + 1) to 14 which corresponds to awqos. Region2 starts from (level2 + 1) up to 15. Note that for PA, awqos values are used directly as port priorities, where the higher the value corresponds to higher port priority. All of the map_level* registers must be set to distinct values. Programming Mode: Quasi-dynamic Group 3
7-4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3-0 wqos_map_level1	Separation level indicating the end of region0 mapping; start of region0 is 0. Possible values for level1 are 0 to 13 which corresponds to awqos. Note that for PA, awqos values are used directly as port priorities, where the higher the value corresponds to higher port priority. All of the map_level* registers must be set to distinct values. Programming Mode: Quasi-dynamic Group 3

36.13.17 Port n Write QoS Configuration Register 1 (PCFGWQOS1_1)

Offset

Register	Offset
PCFGWQOS1_1	158h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved					wqos_map_timeout2										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved					wqos_map_timeout1										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

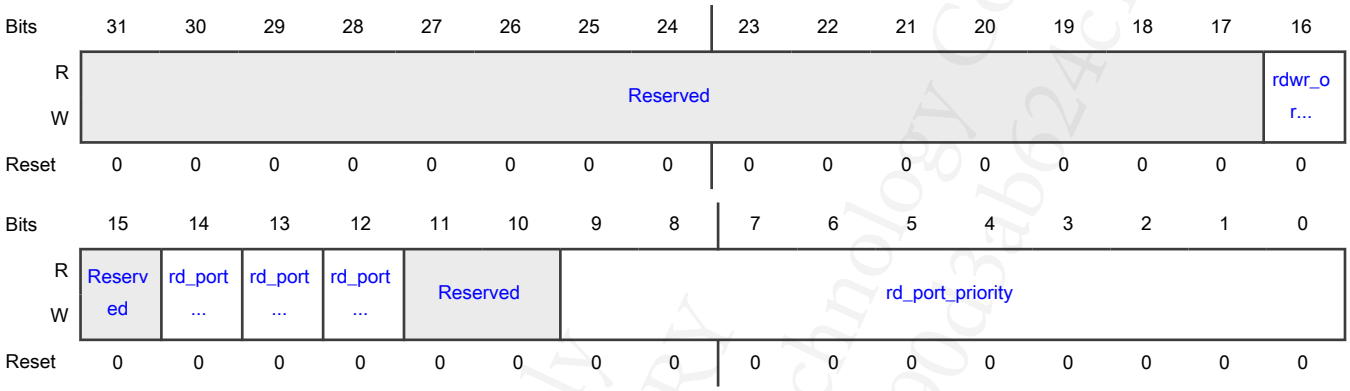
Field	Function
31-27 —	Reserved
26-16 wqos_map_timeout2	Specifies the timeout value for write transactions in region 2. Programming Mode: Quasi-dynamic Group 3
15-11 —	Reserved
10-0 wqos_map_timeout1	Specifies the timeout value for write transactions in region 0 and 1. Programming Mode: Quasi-dynamic Group 3

36.13.18 Port n Configuration Read (PCFGR_2)

Offset

Register	Offset
PCFGR_2	16Ch

Diagram



Fields

Field	Function
31-17 —	Reserved
16 rdwr_ordered_en	Enable ordered read/writes. If set to 1, preserves the ordering between read transaction and write transaction issued to the same address, on a given port. In other words, the controller ensures that all same address read and write commands from the application port interface are transported to the DFI interface in the order of acceptance. This feature is useful in cases where software coherency is desired for masters issuing back-to-back read/write transactions without waiting for write/read responses. Note that this register has an effect only if necessary logic is instantiated via the UMCTL2_RDWR_ORDERED_n parameter. Programming Mode: Static
15 —	Reserved
14 rd_port_pagematch_en	If set to 1, enables the Page Match feature. If enabled, once a requesting port is granted, the port is continued to be granted if the following immediate commands are to the same memory page (same bank and same row). See also related PCCFG.pagematch_limit register. Programming Mode: Static
13 rd_port_urgent_en	If set to 1, enables the AXI urgent sideband signal (arurgent). When enabled and arurgent is asserted by the master, that port becomes the highest priority and co_gs_go2critical_lpr/co_gs_go2critical_hpr signal to the MC is asserted if enabled in PCCFG.go2critical_en register. Note that arurgent signal can be asserted anytime and as long as required which is independent of address handshaking (it is not associated with any particular command). Programming Mode: Static
12	If set to 1, enables aging function for the read channel of the port. Programming Mode: Static

Table continues on the next page...

Table continued from the previous page...

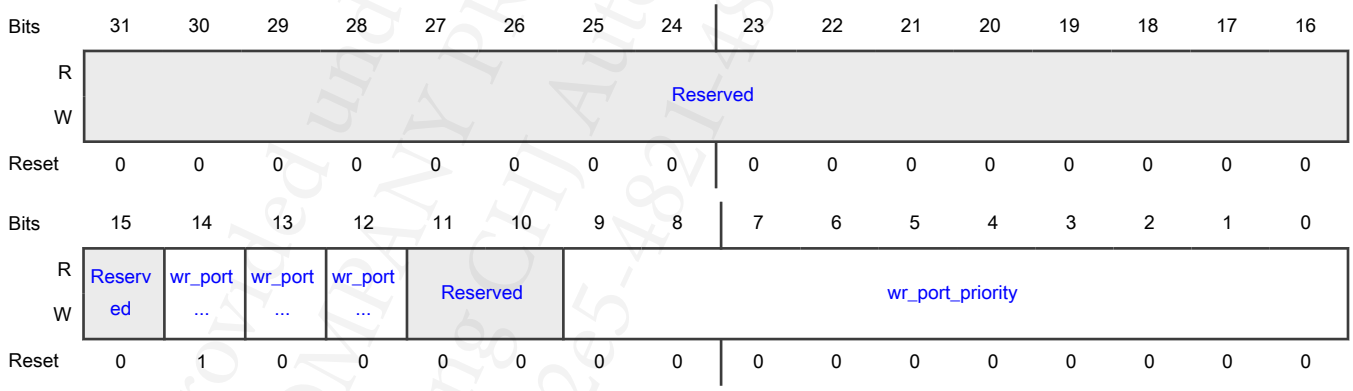
Field	Function
rd_port_aging_en	
11-10 —	Reserved
9-0 rd_port_priority	Determines the initial load value of read aging counters. These counters will be parallel loaded after reset, or after each grant to the corresponding port. The aging counters down-count every clock cycle where the port is requesting but not granted. The higher significant 5-bits of the read aging counter sets the priority of the read channel of a given port. Port's priority will increase as the higher significant 5-bits of the counter starts to decrease. When the aging counter becomes 0, the corresponding port channel will have the highest priority level (timeout condition - Priority0). For multi-port configurations, the aging counters cannot be used to set port priorities when external dynamic priority inputs (arqos) are enabled (timeout is still applicable). For single port configurations, the aging counters are only used when they timeout (become 0) to force read-write direction switching. In this case, external dynamic priority input, arqos (for reads only) can still be used to set the MC read priority (2 priority levels: low priority read - LPR, high priority read - HPR) on a command by command basis. Note: The two LSBs of this register field are tied internally to 2'b00. Programming Mode: Static

36.13.19 Port n Configuration Write (PCFGW_2)

Offset

Register	Offset
PCFGW_2	170h

Diagram



Fields

Field	Function
31-15	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14 wr_port_pagematch_en	If set to 1, enables the Page Match feature. If enabled, once a requesting port is granted, the port is continued to be granted if the following immediate commands are to the same memory page (same bank and same row). See also related PCCFG.pagematch_limit register. Programming Mode: Static
13 wr_port_urgent_en	If set to 1, enables the AXI urgent sideband signal (awurgent). When enabled and awurgent is asserted by the master, that port becomes the highest priority and co_gs_go2critical_wr signal to the MC is asserted if enabled in PCCFG.go2critical_en register. Note that awurgent signal can be asserted anytime and as long as required which is independent of address handshaking (it is not associated with any particular command). Programming Mode: Static
12 wr_port_aging_en	If set to 1, enables aging function for the write channel of the port. Programming Mode: Static
11-10 —	Reserved
9-0 wr_port_priority	Determines the initial load value of write aging counters. These counters will be parallel loaded after reset, or after each grant to the corresponding port. The aging counters down-count every clock cycle where the port is requesting but not granted. The higher significant 5-bits of the write aging counter sets the initial priority of the write channel of a given port. Port's priority will increase as the higher significant 5-bits of the counter starts to decrease. When the aging counter becomes 0, the corresponding port channel will have the highest priority level. For multi-port configurations, the aging counters cannot be used to set port priorities when external dynamic priority inputs (awqos) are enabled (timeout is still applicable). For single port configurations, the aging counters are only used when they timeout (become 0) to force read-write direction switching. Note: The two LSBs of this register field are tied internally to 2'b00. Programming Mode: Static

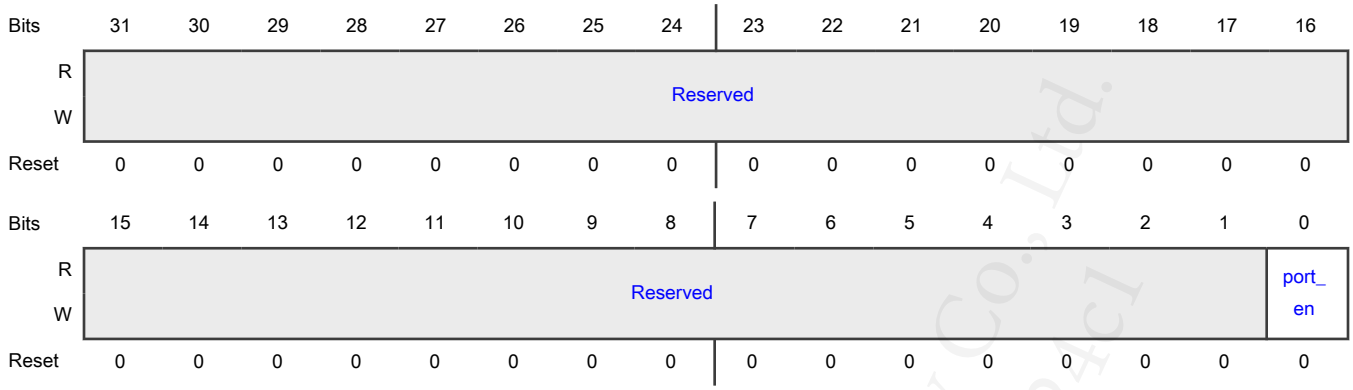
36.13.20 Port n Control (PCTRL_2)

Offset

Register	Offset
PCTRL_2	1F8h

Provided under NXP Only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
c8c462e5-4821-48a6-a90d-3a6

Diagram



Fields

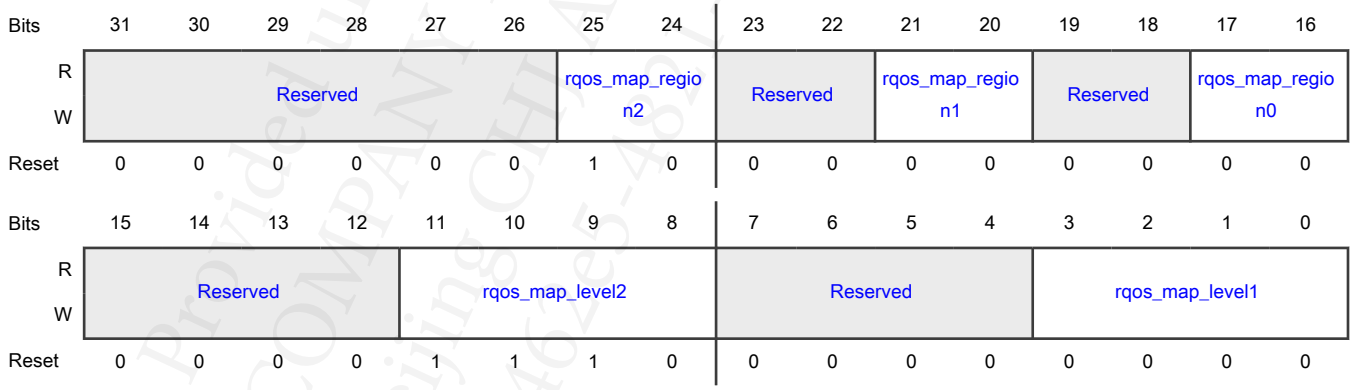
Field	Function
31-1 —	Reserved
0 port_en	Enables AXI port n. Programming Mode: Dynamic

36.13.21 Port n Read QoS Configuration Register 0 (PCFGQOS0_2)

Offset

Register	Offset
PCFGQOS0_2	1FCh

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHX Automotive Technology Co.,Ltd.
c8c46e5-4821-48a8-b231-390d3ab624c1

Fields

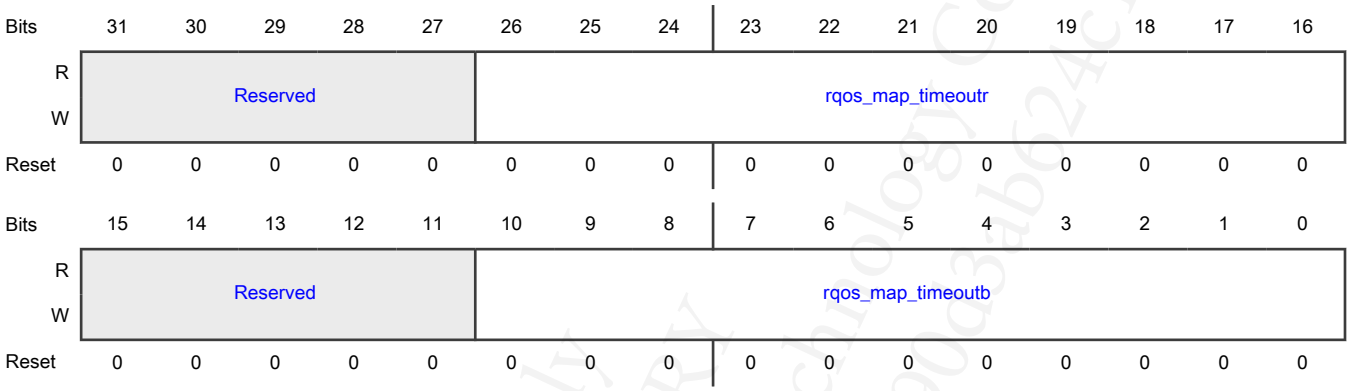
Field	Function
31-26 —	Reserved
25-24 rqos_map_region2	This bitfield indicates the traffic class of region2. For dual address queue configurations, region2 maps to the red address queue. Valid values are 1: VPR and 2: HPR only. When VPR support is disabled (UMCTL2_VPR_EN = 0) and traffic class of region2 is set to 1 (VPR), VPR traffic is aliased to LPR traffic. Programming Mode: Quasi-dynamic Group 3
23-22 —	Reserved
21-20 rqos_map_region1	This bitfield indicates the traffic class of region 1. Valid values are: 0 : LPR, 1: VPR, 2: HPR. For dual address queue configurations, region1 maps to the blue address queue. In this case, valid values are 0: LPR and 1: VPR only. When VPR support is disabled (UMCTL2_VPR_EN = 0) and traffic class of region 1 is set to 1 (VPR), VPR traffic is aliased to LPR traffic. Programming Mode: Quasi-dynamic Group 3
19-18 —	Reserved
17-16 rqos_map_region0	This bitfield indicates the traffic class of region 0. Valid values are: 0: LPR, 1: VPR, 2: HPR. For dual address queue configurations, region 0 maps to the blue address queue. In this case, valid values are: 0: LPR and 1: VPR only. When VPR support is disabled (UMCTL2_VPR_EN = 0) and traffic class of region0 is set to 1 (VPR), VPR traffic is aliased to LPR traffic. Programming Mode: Quasi-dynamic Group 3
15-12 —	Reserved
11-8 rqos_map_level2	Separation level2 indicating the end of region1 mapping; start of region1 is (level1 + 1). Possible values for level2 are (level1 + 1) to 14 which corresponds to arqos. Region2 starts from (level2 + 1) up to 15. Note that for PA, arqos values are used directly as port priorities, where the higher the value corresponds to higher port priority. All of the map_level* registers must be set to distinct values. Programming Mode: Quasi-dynamic Group 3
7-4 —	Reserved
3-0 rqos_map_level1	Separation level1 indicating the end of region0 mapping; start of region0 is 0. Possible values for level1 are 0 to 13 (for dual RAQ) or 0 to 14 (for single RAQ) which corresponds to arqos. Note that for PA, arqos values are used directly as port priorities, where the higher the value corresponds to higher port priority. All of the map_level* registers must be set to distinct values. Programming Mode: Quasi-dynamic Group 3

36.13.22 Port n Read QoS Configuration Register 1 (PCFGQOS1_2)

Offset

Register	Offset
PCFGQOS1_2	200h

Diagram



Fields

Field	Function
31-27 —	Reserved
26-16 rqos_map_timeoutr	Specifies the timeout value for transactions mapped to the red address queue. Programming Mode: Quasi-dynamic Group 3
15-11 —	Reserved
10-0 rqos_map_timeoutb	Specifies the timeout value for transactions mapped to the blue address queue. Programming Mode: Quasi-dynamic Group 3

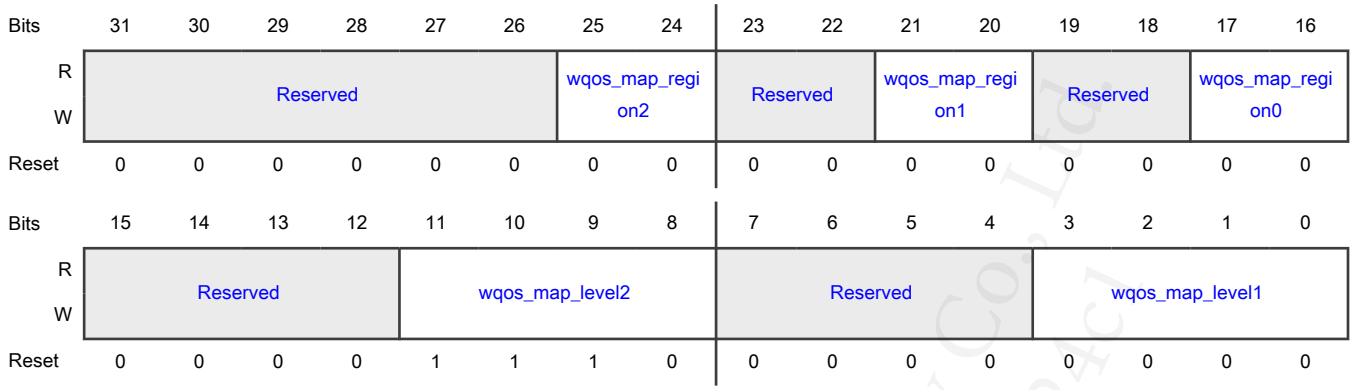
36.13.23 Port n Write QoS Configuration Register 0 (PCFGWQOS0_2)

Offset

Register	Offset
PCFGWQOS0_2	204h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHX Automotive Technology Co., Ltd.
c8c46225-4891-48ac-b251-3903ab624c1

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24 wqos_map_region2	This bitfield indicates the traffic class of region 2. Valid values are: 0: NPW, 1: VPW. When VPW support is disabled (UMCTL2_VPW_EN = 0) and traffic class of region 2 is set to 1 (VPW), VPW traffic is aliased to NPW traffic. Programming Mode: Quasi-dynamic Group 3
23-22 —	Reserved
21-20 wqos_map_region1	This bitfield indicates the traffic class of region 1. Valid values are: 0: NPW, 1: VPW. When VPW support is disabled (UMCTL2_VPW_EN = 0) and traffic class of region 1 is set to 1 (VPW), VPW traffic is aliased to NPW traffic. Programming Mode: Quasi-dynamic Group 3
19-18 —	Reserved
17-16 wqos_map_region0	This bitfield indicates the traffic class of region 0. Valid values are: 0: NPW, 1: VPW. When VPW support is disabled (UMCTL2_VPW_EN = 0) and traffic class of region 0 is set to 1 (VPW), VPW traffic is aliased to NPW traffic. Programming Mode: Quasi-dynamic Group 3
15-12 —	Reserved
11-8 wqos_map_level2	Separation level2 indicating the end of region1 mapping; start of region1 is (level1 + 1). Possible values for level2 are (level1 + 1) to 14 which corresponds to awqos. Region2 starts from (level2 + 1) up to 15. Note that for PA, awqos values are used directly as port priorities, where the higher the value corresponds to higher port priority. All of the map_level* registers must be set to distinct values. Programming Mode: Quasi-dynamic Group 3
7-4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3-0 wqos_map_level1	Separation level indicating the end of region0 mapping; start of region0 is 0. Possible values for level1 are 0 to 13 which corresponds to awqos. Note that for PA, awqos values are used directly as port priorities, where the higher the value corresponds to higher port priority. All of the map_level* registers must be set to distinct values. Programming Mode: Quasi-dynamic Group 3

36.13.24 Port n Write QoS Configuration Register 1 (PCFGWQOS1_2)

Offset

Register	Offset
PCFGWQOS1_2	208h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved					wqos_map_timeout2										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved					wqos_map_timeout1										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

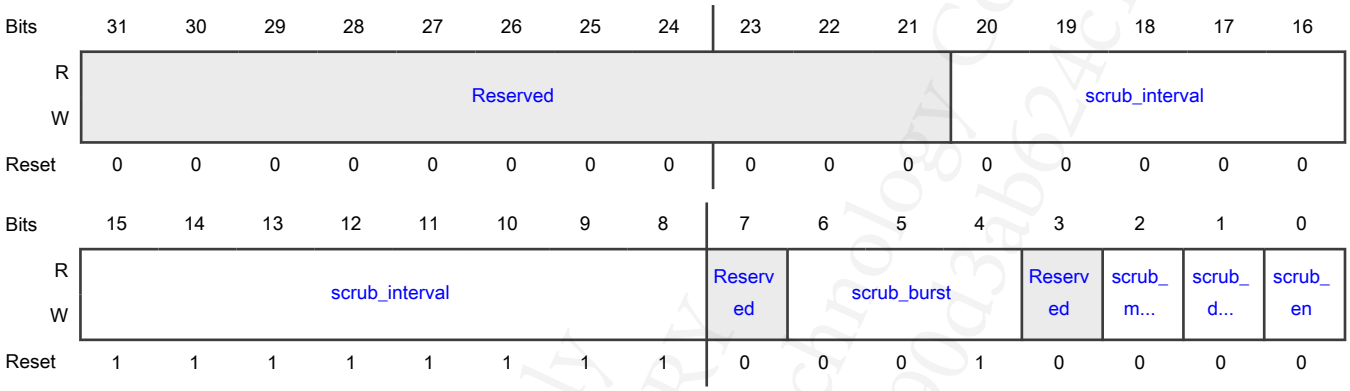
Field	Function
31-27 —	Reserved
26-16 wqos_map_timeout2	Specifies the timeout value for write transactions in region 2. Programming Mode: Quasi-dynamic Group 3
15-11 —	Reserved
10-0 wqos_map_timeout1	Specifies the timeout value for write transactions in region 0 and 1. Programming Mode: Quasi-dynamic Group 3

36.13.25 Scrubber Control (SBRCTL)

Offset

Register	Offset
SBRCTL	B2Ch

Diagram



Fields

Field	Function
31-21 —	Reserved
20-8 scrub_interval	Scrub interval. (512 x scrub_interval) number of clock cycles between two scrub read commands. If set to 0, scrub commands are issued back-to-back. This mode of operation (scrub_interval=0) can typically be used for scrubbing the full range of memory at once before or after SW controlled low power operations. After completing the full range of scrub while scrub_interval=0, scrub_done register is set and sbr_done_intr interrupt signal is asserted. This mode can't be used with Inline ECC: If MEMC_INLINE_ECC is 1 and scrub_interval is programme to 0, then RMW logic inside scrubber is disabled. New programmed value will take effect only after scrubber is disabled by programming scrub_en to 0. Unit: Multiples of 512 sbr_clk cycles. Programming Mode: Dynamic
7 —	Reserved
6-4 scrub_burst	Scrub burst count. Determines the number of back-to-back scrub read commands that can be issued together when the controller is in one of the HW controlled low power modes, both normal operation mode and low-power mode with Inline ECC. During these modes, the period of the scrub burst becomes "scrub_burst*scrub_interval" cycles. (Inline ECC): 1: 8 reads, 2: 16 reads, 3: 32 reads. New programmed value will take effect only after scrubber is disabled by programming scrub_en to 0. Programming Mode: Dynamic
3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

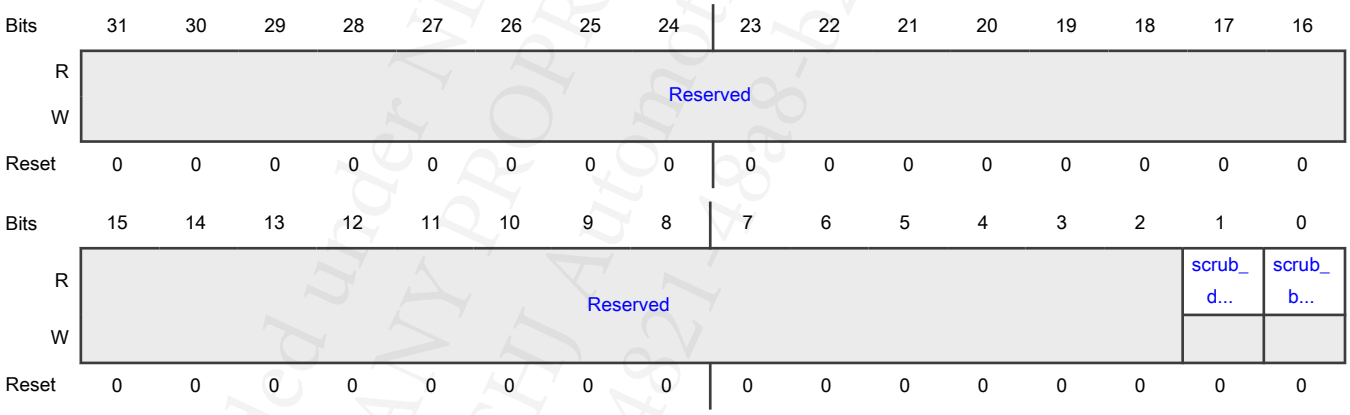
Field	Function
2 scrub_mode	scrub_mode:0 ECC scrubber will perform reads scrub_mode:1 ECC scrubber will perform writes Programming Mode: Dynamic
1 scrub_during_lowpower	Continue scrubbing during low power. If set to 1, burst of scrubs will be issued in HW controlled low power modes. There are two such modes: automatically initiated by idleness or initiated by Hardware low power interface. If set to 0, the scrubber will not attempt to send commands while the MC is in HW controlled low power modes. In this case, the scrubber will remember the last address issued and will automatically continue from there when the MC exits the LP mode. Programming Mode: Dynamic
0 scrub_en	Enable ECC scrubber. If set to 1, enables the scrubber to generate background read commands after the memories are initialized. If set to 0, disables the scrubber, resets the address generator to 0 and clears the scrubber status. This bitfield must be accessed separately from the other bitfields in this register. Programming Mode: Dynamic

36.13.26 Scrubber Status (SBRSTAT)

Offset

Register	Offset
SBRSTAT	B30h

Diagram



Fields

Field	Function
31-2 —	Reserved
1	Scrubber done. Controller sets this bit to 1, after full range of addresses are scrubbed once while scrub_interval is set to 0. Cleared if scrub_en is set to 0 (scrubber disabled) or scrub_interval is set to a

Table continues on the next page...

Table continued from the previous page...

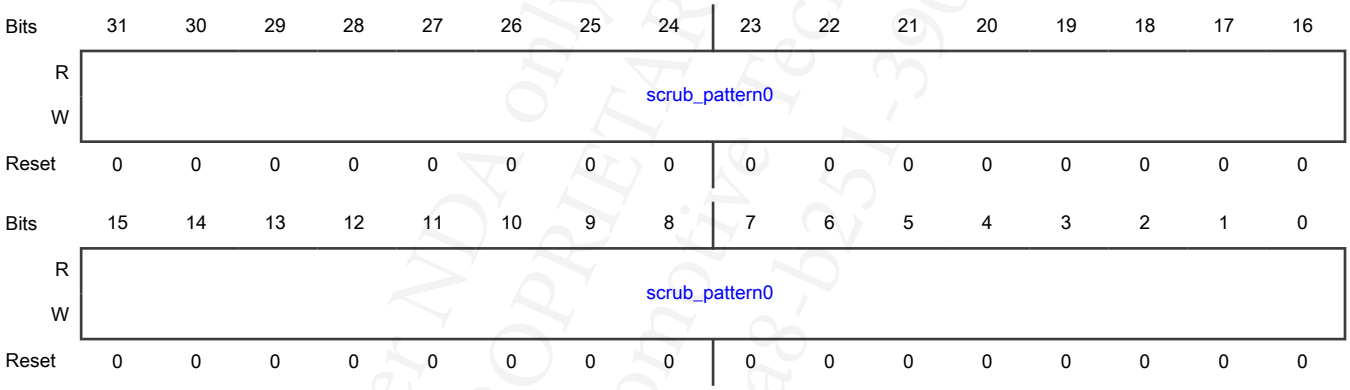
Field	Function
scrub_done	non-zero value for normal scrub operation. The interrupt signal, sbr_done_intr, is equivalent to this status bitfield. Programming Mode: Dynamic
0 scrub_busy	Scrubber busy. Controller sets this bit to 1 when the scrubber logic has outstanding read commands being executed. Cleared when there are no active outstanding scrub reads in the system. Programming Mode: Dynamic

36.13.27 Scrubber Write Data Pattern0 (SBRWDATA0)

Offset

Register	Offset
SBRWDATA0	B34h

Diagram



Fields

Field	Function
31-0 scrub_pattern0	ECC Scrubber write data pattern for data bus[31:0] Programming Mode: Dynamic

36.14 UMCTL2_REGS register descriptions

36.14.1 UMCTL2_REGS memory map

This section and all its sub-sections are Synopsys Proprietary. Used with permission. DDR MC registers

UMCTL2_REGS base address: 403C_0000h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e2-4811-484a-b251-390d3ab624c1

Offset	Register	Width (In bits)	Access	Reset value
0h	Master (MSTR)	32	RW	0304_0001h
4h	Operating mode status (STAT)	32	RO	0000_0000h
10h	Mode Register read/write control 0 (MRCTRL0)	32	RW	0000_0030h
14h	Mode Register Read/Write Control 1 (MRCTRL1)	32	RW	0000_0000h
18h	Mode Register Read/Write Status (MRSTAT)	32	RO	0000_0000h
1Ch	Mode Register Read/Write Control 2 (MRCTRL2)	32	RW	0000_0000h
20h	Temperature derate enable (DERATEEN)	32	RW	0000_0000h
24h	Temperature derate interval (DERATEINT)	32	RW	0080_0000h
2Ch	Temperature derate control (DERATECTL)	32	RW	0000_0001h
30h	Low power control (PWRCTL)	32	RW	0000_0000h
34h	Low Power Timing (PWRTMG)	32	RW	0040_2010h
38h	Hardware low power control (HWLPCTL)	32	RW	0000_0003h
50h	Refresh Control 0 (RFSHCTL0)	32	RW	0021_0000h
54h	Refresh control 1 (RFSHCTL1)	32	RW	0000_0000h
60h	Refresh Control 3 (RFSHCTL3)	32	RW	0000_0000h
64h	Refresh Timing (RFSHTMG)	32	RW	0062_008Ch
68h	Refresh timing 1 (RFSHTMG1)	32	RW	008C_0000h
70h	ECC Configuration 0 (ECCCFG0)	32	RW	033F_7F40h
74h	ECC Configuration 1 (ECCCFG1)	32	RW	0000_07B0h
78h	SECDED ECC Status (ECCSTAT)	32	RO	0000_0000h
7Ch	ECC Control (ECCCTL)	32	RW	0000_0700h
80h	ECC Error Counter (ECCERRCNT)	32	RO	0000_0000h
84h	ECC Corrected Error Address 0 (ECCCADDR0)	32	RO	0000_0000h
88h	ECC Corrected Error Address 1 (ECCCADDR1)	32	RO	0000_0000h
8Ch	ECC Corrected Syndrome 0 (ECCCSYN0)	32	RO	0000_0000h
90h	ECC Corrected Syndrome 1 (ECCCSYN1)	32	RO	0000_0000h
94h	ECC Corrected Syndrome 2 (ECCCSYN2)	32	RO	0000_0000h
98h	ECC Corrected Data Bit Mask 0 (ECCBITMASK0)	32	RO	0000_0000h
9Ch	ECC Corrected Data Bit Mask 1 (ECCBITMASK1)	32	RO	0000_0000h
A0h	ECC Corrected Data Bit Mask 2 (ECCBITMASK2)	32	RO	0000_0000h
A4h	ECC Uncorrected Error Address 0 (ECCUADDR0)	32	RO	0000_0000h
A8h	ECC Uncorrected Error Address 1 (ECCUADDR1)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
ACh	ECC Uncorrected Syndrome 0 (ECCUSYN0)	32	RO	0000_0000h
B0h	ECC Uncorrected Syndrome 1 (ECCUSYN1)	32	RO	0000_0000h
B4h	ECC Uncorrected Syndrome 2 (ECCUSYN2)	32	RO	0000_0000h
B8h	ECC Data Poisoning Address 0 (ECCPOISONADDR0)	32	RW	0000_0000h
BCh	ECC Data Poisoning Address 1 (ECCPOISONADDR1)	32	RW	0000_0000h
C0h	CRC Parity Control 0 (CRCPARCTL0)	32	RW	0000_0000h
C4h	CRC Parity Control 1 (CRCPARCTL1)	32	RW	0000_1000h
CCh	CRC Parity Status (CRCPARSTAT)	32	RO	0000_0000h
D0h	DRAM Initialization 0 (INIT0)	32	RW	0002_004Eh
D4h	DRAM Initialization 1 (INIT1)	32	RW	0000_0000h
D8h	DRAM Initialization 2 (INIT2)	32	RW	0000_0D05h
DCh	DRAM Initialization 3 (INIT3)	32	RW	0000_0510h
E0h	DRAM Initialization 4 (INIT4)	32	RW	0000_0000h
E4h	DRAM Initialization 5 (INIT5)	32	RW	0010_0004h
E8h	DRAM Initialization 6 (INIT6)	32	RW	0000_0000h
ECh	DRAM Initialization 7 (INIT7)	32	RW	0000_0000h
F0h	DIMM Control (DIMMCTL)	32	RW	0000_0000h
F4h	Rank Control (RANKCTL)	32	RW	0000_066Fh
100h	DRAM Timing 0 (DRAMTMG0)	32	RW	0F10_1B0Fh
104h	DRAM Timing 1 (DRAMTMG1)	32	RW	0008_0414h
108h	DRAM Timing 2 (DRAMTMG2)	32	RW	0305_060Dh
10Ch	DRAM Timing 3 (DRAMTMG3)	32	RW	0050_400Ch
110h	DRAM Timing 4 (DRAMTMG4)	32	RW	0504_0405h
114h	DRAM Timing 5 (DRAMTMG5)	32	RW	0505_0403h
118h	DRAM Timing 6 (DRAMTMG6)	32	RW	0202_0005h
11Ch	DRAM Timing 7 (DRAMTMG7)	32	RW	0000_0202h
120h	DRAM Timing 8 (DRAMTMG8)	32	RW	0303_4405h
124h	DRAM Timing 9 (DRAMTMG9)	32	RW	0004_040Dh
128h	DRAM Timing 10 (DRAMTMG10)	32	RW	001C_180Ah
12Ch	DRAM Timing 11 (DRAMTMG11)	32	RW	440C_021Ch
130h	DRAM Timing 12 (DRAMTMG12)	32	RW	0002_0010h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
134h	DRAM Timing 13 (DRAMTMG13)	32	RW	1C20_0004h
138h	DRAM Timing 14 (DRAMTMG14)	32	RW	0000_00A0h
13Ch	DRAM Timing 15 (DRAMTMG15)	32	RW	0000_0000h
180h	ZQ Control 0 (ZQCTL0)	32	RW	0200_0040h
184h	ZQ Control 1 (ZQCTL1)	32	RW	0200_0100h
188h	ZQ Control 2 (ZQCTL2)	32	RW	0000_0000h
18Ch	ZQ Status (ZQSTAT)	32	RO	0000_0000h
190h	DFI Timing 0 (DFITMG0)	32	RW	0702_0002h
194h	DFI Timing 1 (DFITMG1)	32	RW	0000_0404h
198h	DFI Low Power Configuration 0 (DFILPCFG0)	32	RW	0700_0000h
19Ch	DFI Low Power Configuration 1 (DFILPCFG1)	32	RW	0000_0000h
1A0h	DFI Update 0 (DFIUPD0)	32	RW	0040_0003h
1A4h	DFI Update 1 (DFIUPD1)	32	RW	0001_0001h
1A8h	DFI Update 2 (DFIUPD2)	32	RW	8000_0000h
1B0h	DFI Miscellaneous Control (DFIMISC)	32	RW	0000_0041h
1B4h	DFI Timing 2 (DFITMG2)	32	RW	0000_0202h
1B8h	DFI Timing 3 (DFITMG3)	32	RW	0000_0000h
1BCh	DFI Status (DFISTAT)	32	RO	0000_0000h
1C0h	DM/DBI Control (DBICTL)	32	RW	0000_0001h
1C4h	DFI PHY Master (DFIPHYMSTR)	32	RW	0000_0001h
200h	Address Map 0 (ADDRMAP0)	32	RW	0000_0000h
204h	Address Map 1 (ADDRMAP1)	32	RW	0000_0000h
208h	Address Map 2 (ADDRMAP2)	32	RW	0000_0000h
20Ch	Address Map 3 (ADDRMAP3)	32	RW	0000_0000h
210h	Address Map 4 (ADDRMAP4)	32	RW	0000_0000h
214h	Address Map 5 (ADDRMAP5)	32	RW	0000_0000h
218h	Address Map 6 (ADDRMAP6)	32	RW	0000_0000h
21Ch	Address Map 7 (ADDRMAP7)	32	RW	0000_0000h
220h	Address Map 8 (ADDRMAP8)	32	RW	0000_0000h
224h	Address Map 9 (ADDRMAP9)	32	RW	0000_0000h
228h	Address Map 10 (ADDRMAP10)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
22Ch	Address Map 11 (ADDRMAP11)	32	RW	0000_0000h
240h	ODT Configuration (ODTCFG)	32	RW	0400_0400h
244h	ODT/Rank Map (ODTMAP)	32	RW	0000_2211h
250h	Scheduler Control (SCHED)	32	RW	0000_1005h
254h	Scheduler Control 1 (SCHED1)	32	RW	0000_0000h
25Ch	High Priority Read CAM 1 (PERFHPR1)	32	RW	0F00_0001h
264h	Low Priority Read CAM 1 (PERFLPR1)	32	RW	0F00_007Fh
26Ch	Write CAM 1 (PERFWR1)	32	RW	0F00_007Fh
300h	Debug 0 (DBG0)	32	RW	0000_0000h
304h	Debug 1 (DBG1)	32	RW	0000_0000h
308h	CAM Debug (DBGCAM)	32	RO	0000_0000h
30Ch	Command Debug (DBGCMD)	32	RW	0000_0000h
310h	Status Debug (DBGSTAT)	32	RO	0000_0000h
318h	CAM Debug 1 (DBGCAM1)	32	RO	0000_0000h
320h	Software Register Programming Control Enable (SWCTL)	32	RW	0000_0001h
324h	Software Register Programming Control Status (SWSTAT)	32	RO	0000_0001h
330h	On-Chip Parity Configuration 0 (OCPARCFG0)	32	RW	00B0_3032h
334h	On-Chip Parity Configuration 1 (OCPARCFG1)	32	RW	0000_0000h
338h	On-Chip Parity Status 0 (OCPARSTAT0)	32	RO	0000_0000h
33Ch	On-Chip Parity Status 1 (OCPARSTAT1)	32	RO	0000_0000h
340h	On-Chip Parity Status 2 (OCPARSTAT2)	32	RO	0000_0000h
36Ch	AXI Poison Configuration (POISONCFG)	32	RW	0011_0011h
370h	AXI Poison Status (POISONSTAT)	32	RO	0000_0000h
374h	Advanced ECC Index (ADVECCINDEX)	32	RW	0000_0000h
37Ch	ECC Poison Pattern 0 (ECCPOISONPAT0)	32	RW	0000_0000h
384h	ECC Poison Pattern 2 (ECCPOISONPAT2)	32	RW	0000_0000h
388h	Address protection within ECC Status (ECCAPSTAT)	32	RO	0000_0000h
3C0h	Register parity configuration (REGPARCFG)	32	RW	0000_0002h
3C4h	Register parity status (REGPARSTAT)	32	RO	See description
3E0h	On-Chip command/Address Protection Configuration (OCCAPCFG)	32	RW	0001_0000h

Table continues on the next page...

Table continued from the previous page...

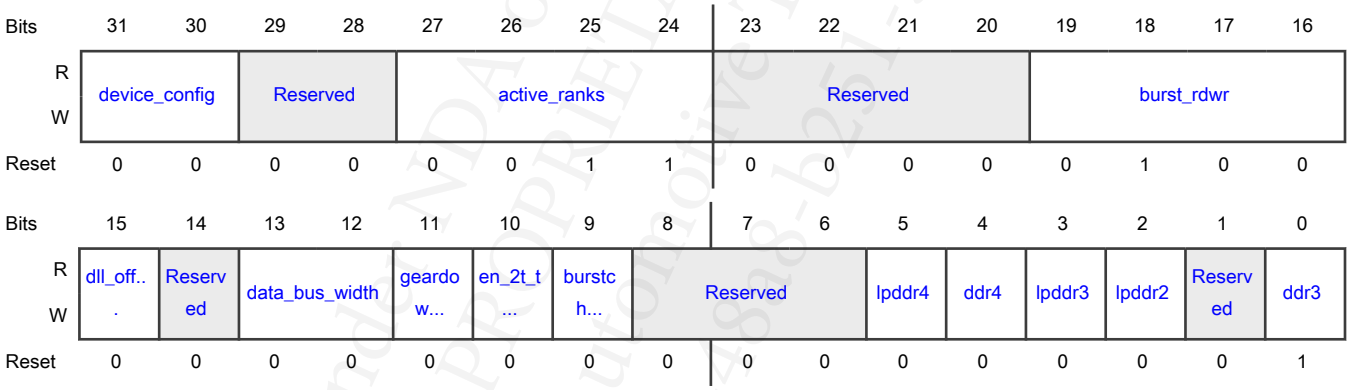
Offset	Register	Width (In bits)	Access	Reset value
3E4h	On-Chip command/Address Protection Status (OCCAPSTAT)	32	RO	0000_0000h
3E8h	On-Chip command/Address Protection Configuration 1 (OCCAPCFG1)	32	RW	0001_0001h
3ECh	On-Chip command/Address Protection Status 1 (OCCAPSTAT1)	32	RO	0000_0000h
3F0h	Temperature Derate Status (DERATESTAT)	32	RO	0000_0000h

36.14.2 Master (MSTR)

Offset

Register	Offset
MSTR	0h

Diagram



Fields

Field	Function
31-30 device_config	Device configuration Selects the configuration of the DRAM device. Programming Mode: Static 00b - x4 device 01b - x8 device 10b - x16 device 11b - x32 device

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-28 —	Reserved
27-24 active_ranks	Active ranks Selects the number of active ranks. This field is present only for multi-rank configurations. Each bit represents one rank: 1 = rank is populated; 0 = rank is not populated. For two-rank configurations, bits 26–27 are reserved. The lsb is the lowest rank number. Programming Mode: Static **NOTE** All values not listed below are reserved. 0001b - One rank populated 0011b - Two ranks populated 1111b - Four ranks populated. Valid only in a four-rank configuration.
23-20 —	Reserved
19-16 burst_rdwr	DRAM burst length Selects the burst length (BL) used to access the DRAM. This selection must match the burst length mode register setting in the DRAM. Programming Mode: Static **NOTE** All values not listed below are reserved. 0001b - BL2. Only supported for mDDR and if the MC is operating at 1:1 frequency ratio. Not supported with AXI ports if MEMC_BURST_LENGTH = 8. 0010b - BL4 0100b - BL8. You must use BL8 for DDR3, DDR4 and LPDDR3. You must use BL8 for BC4/BC8 on-the-fly mode for DDR3 and DDR4. 1000b - BL16. Only supported for mDDR, LPDDR2, and LPDDR4. You must use BL16 for LPDDR4.
15 dll_off_mode	Dll-off mode Selects DLL-off mode for low frequency operation. DLL-on mode is for normal frequency operation. Programming Mode: Quasi-dynamic Group 2 0b - DLL-on mode 1b - DLL-off mode
14	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
13-12 data_bus_width	Data bus width Selects the portion of the DQ bus width (excluding any ECC width) used by the DRAM. Programming Mode: Static **NOTE** The following bus-width restrictions apply. The DDR subsystem supports: • Half bus width only when the DRAM bus width is a multiple of 16 • Quarter bus width only when the DRAM bus width is a multiple of 32 and the configuration parameter MEMC_QBUS_SUPPORT = 1. 00b - Full DQ bus width 01b - Half DQ bus width 10b - Half DQ bus width 11b - Reserved
11 geardown_mode	Geardown Selects geardown mode (2N) or normal mode (1N). In 2N mode, the address, command, and control signals are asserted for 2 clock cycles. You can write this field only when the MC is in the Self Refresh state. **NOTE** Geardown mode is not supported if: • Configuration parameter MEMC_CMD_RTN2IDLE = 1 • Configuration parameter UMCTL2_SHARED_AC = 1 (in Shared-AC mode) and the register value is don't care Programming Mode: Quasi-dynamic Group 2 0b - Normal (1N) mode 1b - Geardown (2N) mode
10 en_2t_timing_mode	2T timing Selects 1T or 2T timing. In 2T timing, all command signals (except chip select) are held for 2 clocks on the DRAM bus, and chip select is asserted on the second cycle of the command.

Table continues on the next page...

Provided under NDA only
COMPANY CONFIDENTIAL
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462a-4821-4823-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
	Programming Mode: Static **NOTE** 2T timing is not supported: p> • For LPDDR2, LPDDR3, or LPDDR4 • If the configuration parameter MEMC_CMD_RTN2IDLE = 1 • In DDR4 geardown mode • In Shared-AC dual channel mode and the register value is don't care 0b - 1T timing 1b - 2T timing
9 burstchop	Burst Enable burst chop (BC4 or 8 on-the-fly) in DDR3 or DDR4. The subsystem exercises burst chop for reads only in HIF configurations (UMCTL2_INCL_ARB not set) and if in full bus width mode (data_bus_width = 00) and if MEMC_BURST_LENGTH is equal to 8 or 16. The subsystem exercises burst chop for writes only if partial writes are enabled (UMCTL2_PARTIAL_WR = 1) and if CRC is disabled (CRCPARCTL1[crc_enable] is equal to 0). BC4 (fixed) mode is not supported. Programming Mode: Static 0b - Disabled 1b - Enabled
8-6 —	Reserved
5 lpddr4	LPDDR4 Selects LPDDR4 DRAM. Only present in systems that support LPDDR4. Programming Mode: Static 0b - Not selected 1b - Selected
4 ddr4	DDR4 Selects DDR4 DRAM. Only present in systems that support DDR4. Programming Mode: Static 0b - Not selected 1b - Selected
3	LPDDR3

Table continues on the next page...

Table continued from the previous page...

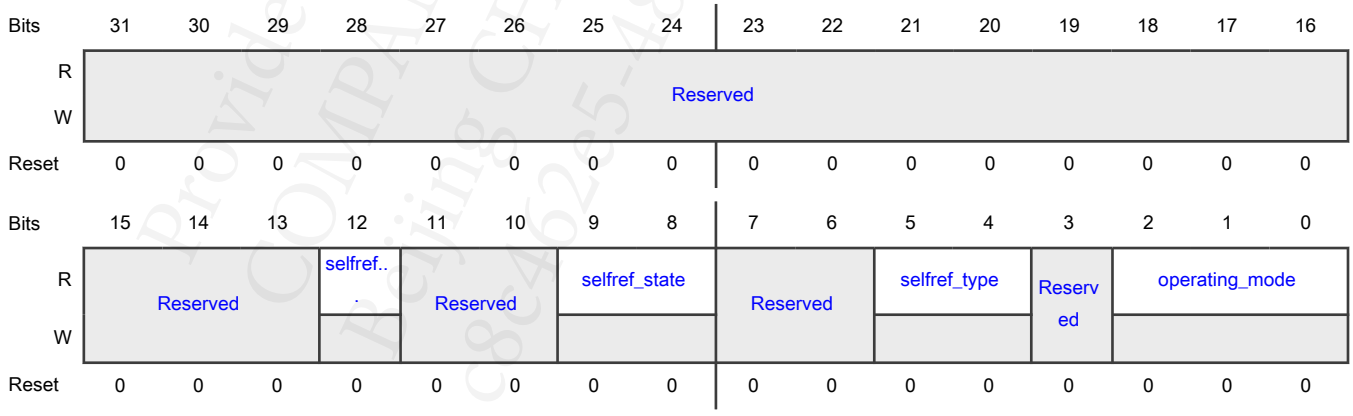
Field	Function
lpddr3	Selects LPDDR3 DRAM. Only present in systems that support LPDDR3. Programming Mode: Static 0b - Not selected 1b - Selected
2 lpddr2	LPDDR2 Selects LPDDR2 DRAM. Only present in systems that support LPDDR2. Programming Mode: Static 0b - Not selected 1b - Selected
1 —	Reserved
0 ddr3	DDR3 Selects DDR3 DRAM. Only present in systems that support DDR3. Programming Mode: Static 0b - Not selected 1b - Selected

36.14.3 Operating mode status (STAT)

Offset

Register	Offset
STAT	4h

Diagram



Provided under IDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8846255-4821-48a8-b251-590d3ab624c1

Fields

Field	Function
31-13 —	Reserved
12 selfref_cam_not_empty	Self refresh with CAMs not empty Indicates if DRAM is in Self Refresh state but the CAMs are not drained. Programming Mode: Static 0b - DRAM has exited Self Refresh state 1b - DRAM is in Self Refresh state but CAMs are not drained
11-10 —	Reserved
9-8 selfref_state	Self refresh state Indicates whether DRAM is in the Self Refresh (other than LPDDR4) or Self Refresh Power Down (LPDDR4) state. Use this field to determine Self Refresh state for frequency change and MRR or MRW access. Programming Mode: Static 00b - Not in Self Refresh state 01b - In Self Refresh 1 state 10b - In Self Refresh Power Down state 11b - In Self Refresh 2 state
7-6 —	Reserved
5-4 selfref_type	Self refresh mode Indicates if DRAM enters the Self Refresh state and whether it did so only as the result of automatic self-refresh control. Programming Mode: Static 00b - Not in Self Refresh (except LPDDR4) or Self Refresh Power Down (LPDDR4). 01b - In Self Refresh (except LPDDR4) or Self Refresh Power Down (LPDDR4) caused by a PHY Master Request. 10b - In Self Refresh (except LPDDR4) or Self Refresh Power Down (LPDDR4) not caused soley under automatic self-refresh control. It could be caused by the hardware low-power interface or software (PWRCTL[selfref_sw]). 11b - In Self Refresh (except LPDDR4) or Self Refresh Power Down (LPDDR4) caused only by automatic self-refresh control.
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2-0 operating_mode	Operating state Indicates the current operating state. Programming Mode: Static 000b - Init 001b - Normal 010b - Power Down 011b - Self Refresh. LPPD4: Self Refresh Power Down 1xxb - mDDR, LPDDR2, LPDDR3, or DDR4: Deep Power Down or Maximum Power Saving

36.14.4 Mode Register read/write control 0 (MRCTRL0)

Offset

Register	Offset
MRCTRL0	10h

Function

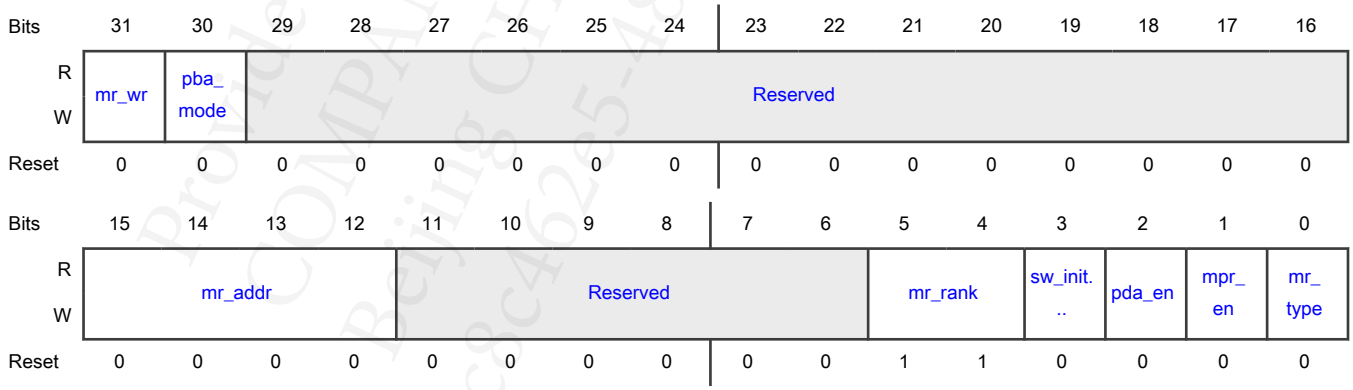
Controls and initates an DRAM Mode Register read or write transaction.

NOTE

Enable only one of the following functions at any given time:

- Software Intervention (sw_init_int)
- Per DRAM Addressability (pda_en)
- Multi Purpose Register (mpr_en)

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-43c8-b251-390d3ab624c1

Fields

Field	Function
31 mr_wr	Mode Register read/write Triggers a Mode Register read or write operation. When the MR operation is complete, the MC automatically clears this field. You must write the other fields in this register in a separate transaction before writing this field. Do not write to this field while DRAM is intializing or is in the Deep Power Down or Maximum Power Saving Mode states. Programming Mode: Dynamic
30 pba_mode	Per buffer addressability Selects per buffer addressability (PBA) mode. If both pda_en and pba_en are selected, the MC initiates PBA access instead of PDA. MRSTAT[pda_done] indicates the completion of PBA access just as it does for PDA. Programming Mode: Dynamic 0b - PDA 1b - PBA
29-16 —	Reserved
15-12 mr_addr	Mode Register address Selects the Mode Register to be written to. For Mode Register addressing in LPDDR2, LPDDR3, and LPDDR4, see MRCTRL1[mr_data]. This signal is also used for writing to control words of the register chip on RDIMMs and LRDIMMs. In that case, it corresponds to the bank address bits sent to the RDIMM or LRDIMM. For DDR4, bits 15-14 correspond to the bank group bits. Therefore, bit 15 as well as bits 14-12 must be set to an appropriate value that is considered both the address mirroring of UDIMMs, RDIMMs, or LRDIMMs, and the output inversion of RDIMMs or LRDIMMs. Programming Mode: Dynamic 0000b - MR0 0001b - MR1 0010b - MR2 0011b - MR3 0100b - MR4 0101b - MR5 0110b - MR6 0111b - MR7
11-6	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5-4 mr_rank	Rank access Selects which ranks are accessed by the Mode Register operation (mr_wr). In typical operation, you access all ranks. However, for multi-rank UDIMMs, RDIMMs, and LRDIMMs that implement address mirroring, you may need to access individual ranks. Each bit selects one rank and lsb is rank 0. Examples: Assume MC is configured for 2 ranks. • 0b01 - Select rank 0 only. • 0b10 - Select rank 1 only. • 0b11 - Select rank 0 and 1. Programming Mode: Dynamic
3 sw_init_int	Software intervention Allows software intervention via MRCTRL0/MRCTRL1 before the automatic DRAM initialization routine. DDR4: Initialize the DDR4 RCD (MR7) before automatic DRAM initialization. LPDDR4: Program additional mode registers before automatic DRAM initialization. For dual-channel DRAM, you must program both channels first. Programming Mode: Dynamic **NOTE** You must write 0 to this field after completing software intervention; otherwise, the DRAM initialization routine will not re-start. 0b - Not allowed 1b - Allowed
2 pda_en	Per DRAM addressability Selects Per DRAM Addressability (PDA) mode. Programming Mode: Dynamic 0b - MRS 1b - MRS in PDA mode
1 mpr_en	Multi-purpose register Selects MRS or WR/RD for MPR operation. **NOTE** The DDR subsystem supports WR/RD for MPR operation only for DDR4. Programming Mode: Dynamic

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MRS 1b - WR/RD for MPR
0 mr_type	Mode Register operation Selects whether the next Mode Register operation is read or write. Use only for for LPDDR2, LPDDR3, LPDDR4, or DDR4 DRAM. Programming Mode: Dynamic 0b - Write 1b - Read

36.14.5 Mode Register Read/Write Control 1 (MRCTRL1)

Offset

Register	Offset
MRCTRL1	14h

Function

Specifies data for a Mode Register write operation.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved														mr_data	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	mr_data															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-18 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
17-0 mr_data	Mode Register write data Write data for all non-LPDDR2, non-LPDDR3, and non-LPDDR4 DRAM. For LPDDR2, LPDDR3, and LPDDR4, bits 15–8 are the Mode Register address and bits 7–0 are the write data. Programming Mode: Dynamic

36.14.6 Mode Register Read/Write Status (MRSTAT)

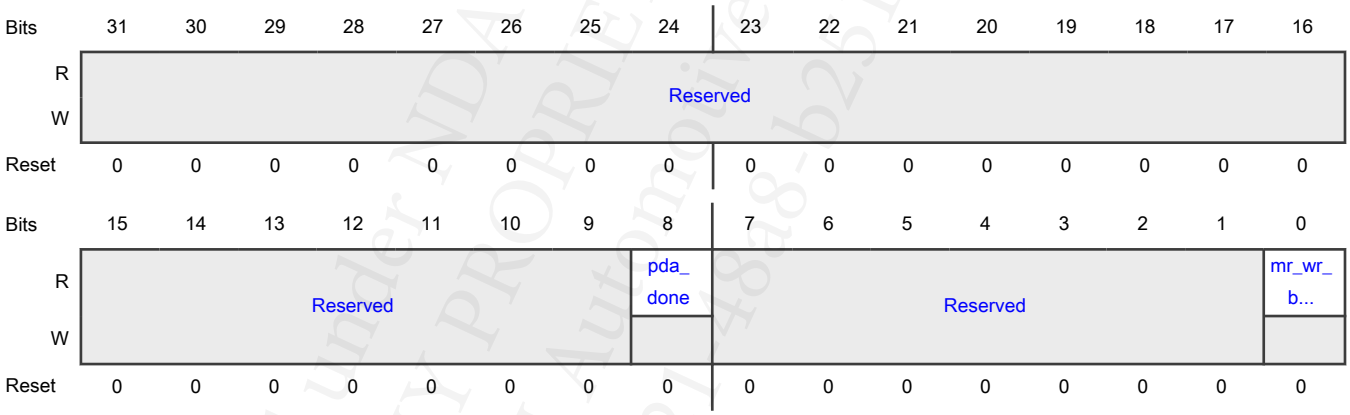
Offset

Register	Offset
MRSTAT	18h

Function

Controls when Mode Register read or write operations can be initiated.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 pda_done	PDA operation complete Indicates whether a MR operation in PDA or PBA mode has completed. This signal goes high when three consecutive MRS commands related to the PDA or PBA mode are issued to the DRAM. It goes low when MRCTRL0[pda_en] becomes 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The SoC core may initiate a MR write operation in PDA or PBA mode only if pda_done = 0. Therefore, we recommended you write 0 to MRCTRL0[pda_en] after pda_done goes high to prepare for the next PDA operation. Programming Mode: Dynamic 0b - In progress or not yet started 1b - Completed
7-1 —	Reserved
0 mr_wr_busy	MR busy Indicates that a Mode Register (MR) operation is in progress. This signal goes high in the clock after the MC accepts the MR write or read request. It goes low when the MR write or read command is issued to the DRAM. The SoC core may initiate a MR write operation only if mr_wr_busy = 0. We recommended that you do not perform MR write or read commands when mr_wr_busy = 1. Programming Mode: Dynamic 0b - Not busy 1b - Busy

36.14.7 Mode Register Read/Write Control 2 (MRCTRL2)

Offset

Register	Offset
MRCTRL2	1Ch

Function

Selects the devices for Mode Register Set operation in PDA mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	mr_device_sel															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	mr_device_sel															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 mr_device_sel	MRS device select Selects to which devices to apply a Mode Register Set (MRS) command in PDA mode. Each bit represents one device. For example, bit 0 corresponds to device 0, bit 1 corresponds to device 1, and so on. Programming Mode: Dynamic Here are the permitted values for each bit: 00000000000000000000000000000000b - Do not apply 00000000000000000000000000000001b - Apply

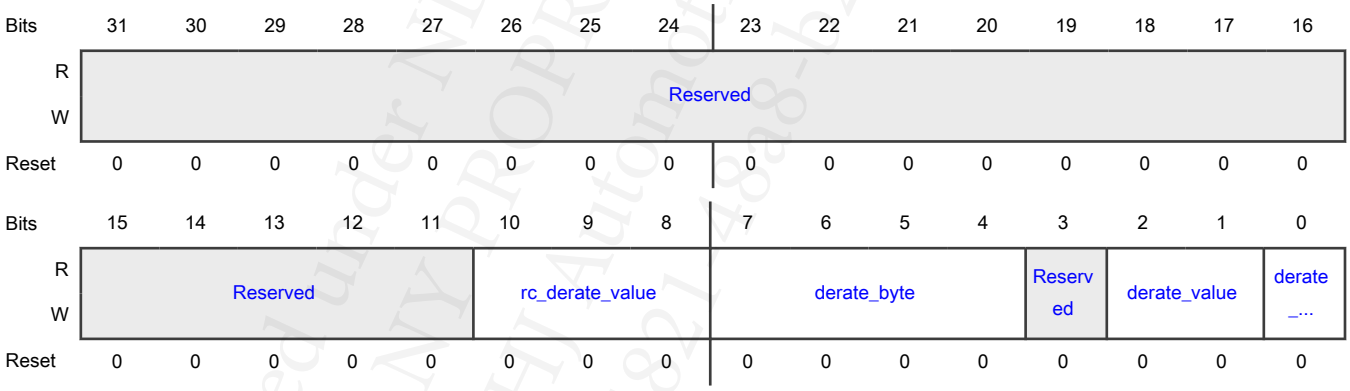
36.14.8 Temperature derate enable (DERATEEN)

Offset

Register	Offset
DERATEEN	20h

Function
Controls temperateure derating operation.

Diagram



Fields

Field	Function
31-11 —	Reserved
10-8 rc_derate_value	tRC derate value

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Specifies the derate value of tRC for LPDDR4 DRAM. Determine the required cycles for derating by dividing 3.75 ns by the core_ddrc_core_clk period and rounding up to the next integer. Set this field to that value minus 1. **NOTE** This field is present only in systems that support LPDDR4 DRAM. Programming Mode: Quasi-dynamic Group 2, Group 4 000b - Derating uses +1 001b - Derating uses +2 010b - Derating uses +3 011b - Derating uses +4
7-4 derate_byte	Derate byte Selects which byte of the MR read data to use for derating. **NOTE** This field is present only in systems that support LPDDR2, LPDDR3, or LPDDR4 DRAM. Programming Mode: Static
3 —	Reserved
2-1 derate_value	Derate value Selects the derating value. **NOTE** This field is present only in systems that support LPDDR2, LPDDR3, or LPDDR4 DRAM. • For all LPDDR2 speed grades: Set to 0 because the derating value of +1.875 ns is less than the core_ddrc_core_clk period. • For LPDDR3 and LPDDR4: If the core_ddrc_core_clk period is less than 1.875 ns, set to 1; otherwise, set to 0. Programming Mode: Quasi-dynamic Group 2, Group 4 00b - Derating uses +1 01b - Derating uses +2
0 derate_enable	Derate enable Enables timing parameter derating using the value in MR4. **NOTE** This field is present only in systems that support LPDDR2, LPDDR3, or LPDDR4 DRAM. It must be 0 for any other DRAM types.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Programming Mode: Dynamic 0b - Disable 1b - Enable

36.14.9 Temperature derate interval (DERATEINT)

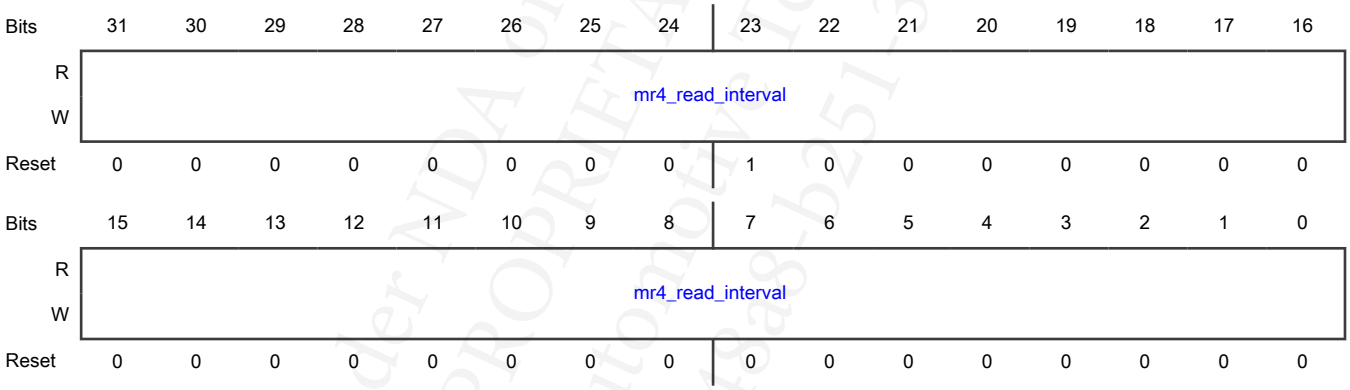
Offset

Register	Offset
DERATEINT	24h

Function

Specifies the internal between two MR4 read operations in DFI clock cycles. This interval derates the timing parameters.

Diagram



Fields

Field	Function
31-0 mr4_read_interval	MR4 read interval The interval between two MR4 read operations in DFI clock cycles. **NOTE** • Do not set this field to zero. • This field is present only in systems that support LPDDR2, LPDDR3, or LPDDR4 DRAM. Programming Mode: Static

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca4c2e5-4821-49a8-b251-3901a3ab624c1

36.14.10 Temperature derate control (DERATECTL)

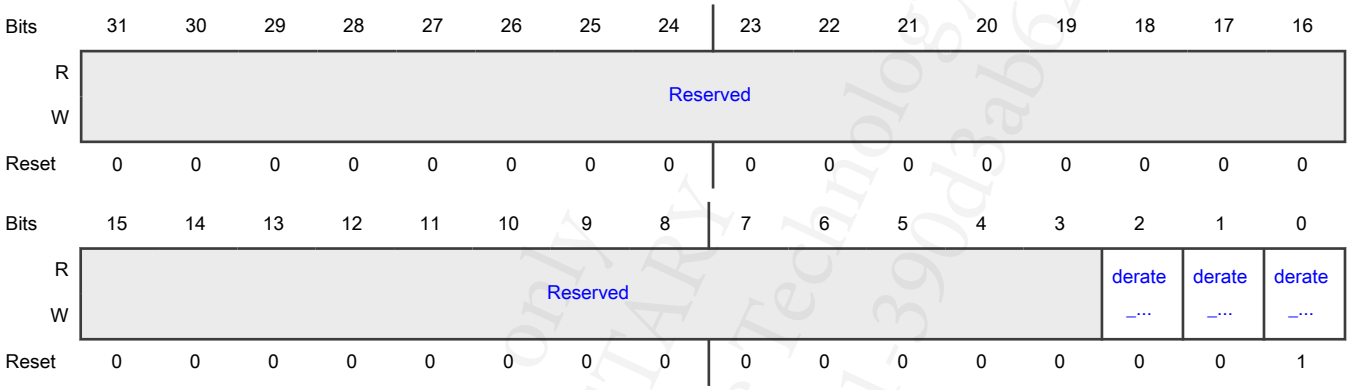
Offset

Register	Offset
DERATECTL	2Ch

Function

Specifies temperature derate behavior.

Diagram



Fields

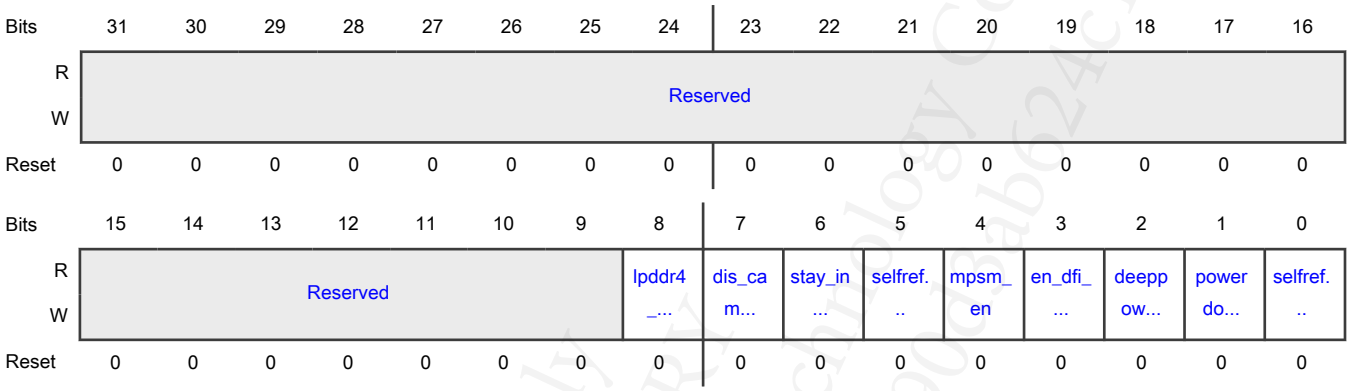
Field	Function
31-3 —	Reserved
2 derate_temp_limit_intr_force	Temperature derate limit interrupt clear Forces assertion of the temperature derate limit interrupt. To force interrupt assertion, write 1 to this field. At the end of the force operation, the MC automatically clears the field. Programming Mode: Dynamic
1 derate_temp_limit_intr_clr	Temperature derate limit interrupt clear Clears the temperature derate limit interrupt. To clear the interrupt, write 1 to this field. At the end of the clear operation, the MC automatically clears the field. Programming Mode: Dynamic
0 derate_temp_limit_intr_en	Temperature derate limit interrupt enable Enables the temperature derate limit interrupt. Programming Mode: Dynamic 0b - Disable 1b - Enable

36.14.11 Low power control (PWRCTL)

Offset

Register	Offset
PWRCTL	30h

Diagram



Fields

Field	Function
31-9 —	Reserved
8 lpddr4_sr_allowed	LPDDR4 SRPD-SR transition Specifies if the transition from Self Refresh Power Down state to Self Refresh state and back is allowed. **NOTE** You cannot modify this field while selfref_sw == 1. Programming Mode: Dynamic 0b - Prohibit transition 1b - Allow transition
7 dis_cam_drain_selfref	Skip CAM draining Specifies whether the MC can put DRAM into the Self Refresh state without draining the content-addressable memory (CAM). **NOTE** • You cannot modify this field while selfref_sw == 1. • This release does not support dis_cam_drain_selfref = 1. dis_cam_drain_selfref must equal 0. Programming Mode: Dynamic

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - CAMs must be drained 1b - CAMs do not have to be drained (unsupported)
6 stay_in_selfref	Stay in self refresh Controls whether the MC can transition DRAM out of the Self Refresh state. Self Refresh is an intermediate state when entering or exiting Self Refresh Power Down for LPDDR4 DRAM. Programming Mode: Dynamic 0b - Allow transition 1b - Prohibit transition
5 selfref_sw	Software self refresh Triggers immediate entry into or exit from the Self Refresh state as long as the MC is not in any of these conditions: • initializing • in the Deep Power Down state • in maximum power saving mode Programming Mode: Dynamic 0b - Exit 1b - Entry
4 mpsm_en	Maximum power saving mode enable Enables the MC to put DRAM into maximum power saving mode (MPSM) when the transaction story is empty. To bring the MC out of MPSM, write 0 to this field. Applies only to systems that support DDR4 DRAM. For all other DRAM types, do not set this field to 1. Affects performance only. If the PHY DWC_AC_CS_USE parameter is disabled, the MC does not support MPSM because the MPSM exit sequence requires the chip-select signal to toggle. **NOTE** If the value of INIT0[skip_dram_init] is 2'b11, you must write 2'b01 to INIT0[skip_dram_init] by entering the self-refresh mode through PWRCTL[selfref_sw], the value of which is 1'b1. This needs to be done before writing 1 to PWRCTL[mpsm_en]. Programming Mode: Dynamic 0b - Disable 1b - Enable
3 en_dfi_dram_clk_disable	Enable DFI DRAM clock disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables the MC to assert DFI DRAM Clock Disable (dfi_dram_clk_disable) if a clock is not required by the DRAM. The MC can assert DFI DRAM Clock Disable under the following restrictions: • DDR2 or DDR3: can only be asserted in the Self Refresh state • DDR4: can be asserted in the following states: — Self Refresh — Maximum Power Saving • mDDR, LPDDR2, LPDDR3: can be asserted in the following states: — Self Refresh — Power Down — Deep Power Down — Normal operation (clock stop) • LPDDR4: can be asserted in the following states: — Self Refresh Power Down — Power Down — Deep Power Down — Normal operation (clock stop) Programming Mode: Dynamic 0b - Disable 1b - Enable
2 deeppowerdown_en	Deep power down enable Enables the MC to put DRAM into the Deep Power Down state when the transaction store is empty. To bring the MC out of Deep Power Down, write 0 to this field. The MC performs automatic DRAM initialization when exiting Deep Power Down. **NOTE** • Applies only to systems that support mDDR, LPDDR2, or LPDDR3 DRAM. For all other DRAM types, do not set this field to 1. • Affects performance only. Programming Mode: Dynamic
1 powerdown_en	Power down enable Enables the MC to put DRAM into the Power Down state after a programmable number of cycles (PWRTMG[powerdown_to_x32]). Programming Mode: Dynamic 0b - Disable

Table continues on the next page...

Table continued from the previous page...

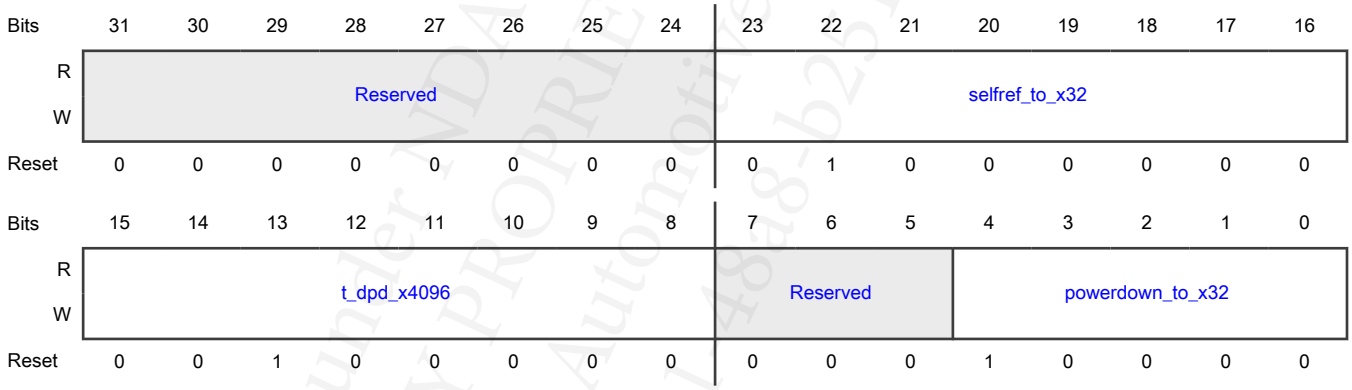
Field	Function
	1b - Enable
0 selfref_en	Self-refresh enable Enables the MC to put DRAM into the Self Refresh state after a programmable number of cycles (PWRTMG[selfref_to_x32]). Programming Mode: Dynamic 0b - Disable 1b - Enable

36.14.12 Low Power Timing (PWRTMG)

Offset

Register	Offset
PWRTMG	34h

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 selfref_to_x32	Self-refresh idle period Specifies the interval, in multiples of 32 DFI clock cycles, of the MC command channel being idle before the MC automatically puts the DRAM into the Self Refresh state. The MC command channel is considered idle when there are no outstanding HIF commands. This function must be enabled (PWRCTL[selfref_en]).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Affects performance only. Programming Mode: Quasi-dynamic Group 4
15-8 t_dpd_x4096	Minimum deep power down period Specifies the minimum amount of time, in multiples of 4096 DFI clock cycles, that DRAM stays in the Deep Power Down state after deassertion of PWRCTL[deeppowerdown_en]. **NOTE** • According to the JEDEC mDDR specification, mDDR exits from Deep Power Down state immediately after deassertion of PWRCTL[deeppowerdown_en]. • According to the JEDEC LPDDR2 and LPDDR3 specifications, the minimum time should be 500 µs. • Applies only to systems that support mDDR, LPDDR2, or LPDDR3 DRAM. • Affects performance only. Programming Mode: Quasi-dynamic Group 4
7-5 —	Reserved
4-0 powerdown_to_x32	Power down idle period Specifies the interval, in multiples of 32 DFI clock cycles, of the MC command channel being idle before the MC automatically puts the DRAM into the Power Down state. The MC command channel is considered idle when there are no outstanding HIF commands. This function must be enabled (PWRCTL[powerdown_en]). **NOTE** Affects performance only. Programming Mode: Quasi-dynamic Group 4

36.14.13 Hardware low power control (HWLPCTL)

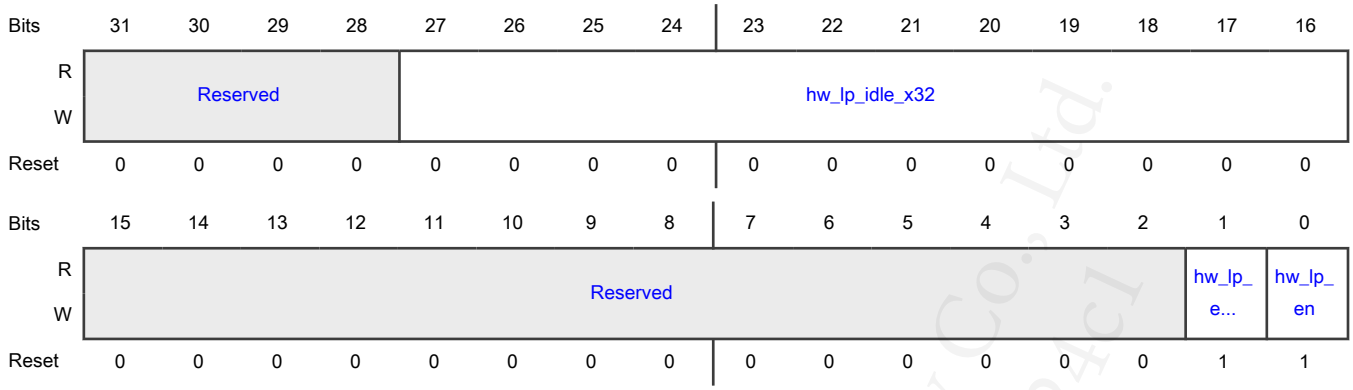
Offset

Register	Offset
HWLPCTL	38h

Function

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHX Automotive Technology Co., Ltd.
c8c462e5-481-A6e-251-39033462461

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 hw_lp_idle_x32	Hardware low power idle period Specifies the period, in multiples of 32 DFI clock cycles, of the MC command channel being idle before the MC automatically puts the DRAM into Hardware Low Power (cactive_ddrc = 0). The MC command channel is considered idle when there are no outstanding HIF commands. Do not write to this field while DRAM is intializing or is in the Deep Power Down or Maximum Power Saving Mode states. **NOTE** Affects performance only. Programming Mode: Static 000000000000b - Disable Hardware Low Power 000000000001b - Illegal value All other values - Idle period, in multiples of 32 DFI clock cycles
15-2 —	Reserved
1 hw_lp_exit_idle_en	Hardware low power exit enable Enables the MC to force exit, via cactive_in_ddrc, from the automatic clock stop, automatic Power Down or automatic Self Refresh states. **NOTE** Does not cause exit of Self Refresh caused by the Hardware Low Power Interface or software (PWRCTL[selfref_sw]). Programming Mode: Static

Table continues on the next page...

Table continued from the previous page...

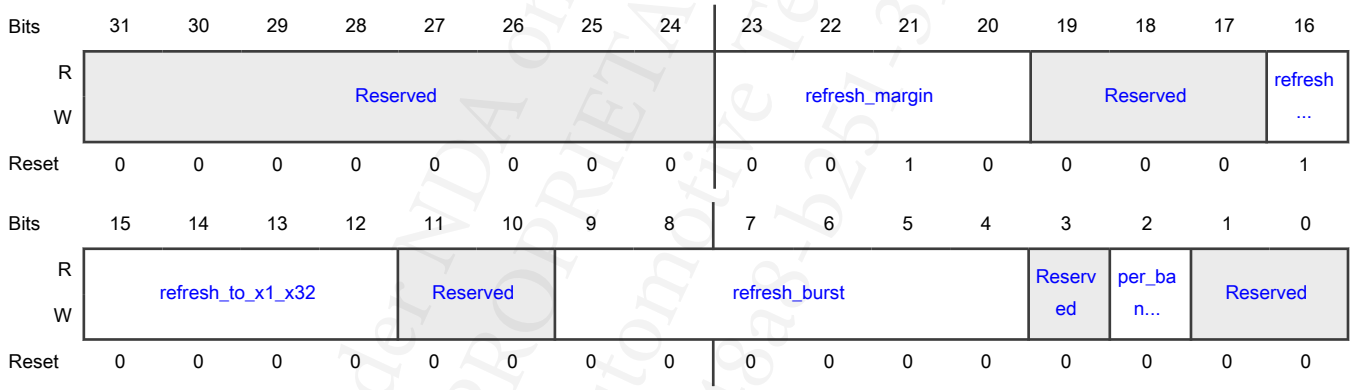
Field	Function
0 hw_lp_en	Hardware low power interface enable Enables the Hardware Low Power Interface. Programming Mode: Quasi-dynamic Group 3 0b - Disable 1b - Enable

36.14.14 Refresh Control 0 (RFSHCTL0)

Offset

Register	Offset
RFSHCTL0	50h

Diagram



Fields

Field	Function
31-24 —	Reserved
23-20 refresh_margin	Refresh margin

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
	Specifies the time period, in multiples of 32 DFI clock cycles, before the critical refresh or page timer expires. You must issue a critical refresh before the end of this time. The recommended value is the default value of 2h. **NOTE** • This value must always be less than the internal t_rfc_nom/32. If RFSHTMG[t_rfc_nom_x1_sel]is equal to 0, then t_rfc_nom is equal to (RFSHTMG[t_rfc_nom_x1_x32] * 32). If RFSHTMG[t_rfc_nom_x1_sel]is equal to 1 (for LPDDR2, LPDDR3, and LPDDR4 per-bank refresh only), then t_rfc_nom is equal to RFSHTMG[t_rfc_nom_x1_x32]. • For LPDDR2, LPDDR3, or LPDDR4, if derating is enabled (DERATEEN[derate_enable]), the internal t_rfc_nom may be divided by four. Programming Mode: Dynamic - Refresh Related
19-17 —	Reserved
16-12 refresh_to_x1_x32	Speculative refresh period Specifies the time of the DRAM bus being idle before the MC initiates a speculative refresh, if the refresh timer (tRFCnom, also known as tREFI) has expired at least once but not enough times to trigger a burst refresh (refresh_burst + 1). The specified value is in DFI clock cycles or multiples of 32 DFI clock cycles depending on RFSHTMG[t_rfc_nom_x1_sel]. Speculative refreshes continue successively until there are no pending refreshes or until the MC receives new read or write commands. **NOTE** Affects performance only. Programming Mode: Dynamic - Refresh Related
11-10 —	Reserved
9-4 refresh_burst	Burst refresh Specifies the number refresh timeouts allowed to accumulate before the MC blocks traffic and forces the refreshs to execute. Closing pages to perform a refresh encurs a one-time penalty for each group of refreshes. Therefore, performing refreshes in a burst reduces the per-refresh penalty for page closings.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	A higher number for refresh_burst slightly increases utilization; a lower number decreases the worst-case latency associated with refreshes. **NOTE** • For information about burst refresh, see section 3.9 of the DDR2 JEDEC specification, JESD79-2F.pdf. • For DDR2 and DDR3, refresh is always per rank, not per bank. Rank refresh can be accumulated over (8 * tREFI) cycles using burst refresh. • For DDR4, in Fine Granularity Refresh state, you can postpone up to 8 refreshes in 1X mode, 16 refreshes in 2X mode, and 32 refreshes in 4X mode. If using PHY-initiated updates, you must ensure that tRFCmax is not violated due to a PHY-initiated update occurring shortly before a refresh burst is due. In this situation, the MC delays the refresh burst until the PHY-initiated update is complete. • For LPDDR2, LPDDR3, and LPDDR4, in per-bank refresh (per_bank_refresh == 1) you postpone up to 64 refreshes. Programming Mode: Dynamic - Refresh Related 000000b - One-refresh burst 000001b - Two-refresh burst All other values - (Value + 1)-refresh burst
3 —	Reserved
2 per_bank_refresh	Per-bank refresh enable Enables per bank refresh, which allows traffic to flow to other banks while each bank refreshes. **NOTE** • This field is present only in systems that support LPDDR2, LPDDR3, or LPDDR4 DRAM. It must be 0 for any other DRAM types. • Per-bank refresh is not supported by all LPDDR2 devices but should be supported by all LPDDR3 and LPDDR4 devices. Programming Mode: Static 0b - All-bank refresh 1b - Per bank refresh
1-0 —	Reserved

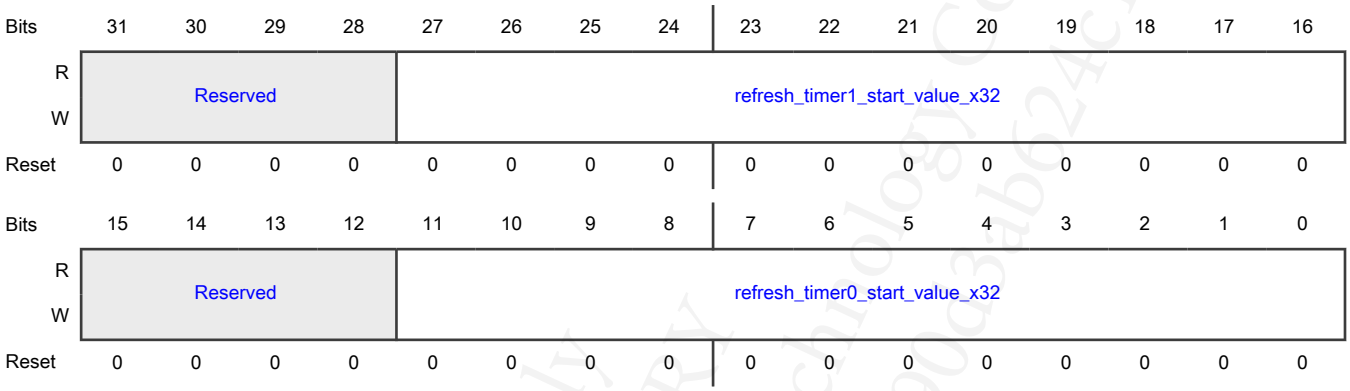
Provided under NDA of NXP
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
c8c462e5-482a-46a8-b251-390d3a0e4d1

36.14.15 Refresh control 1 (RFSHCTL1)

Offset

Register	Offset
RFSHCTL1	54h

Diagram



Fields

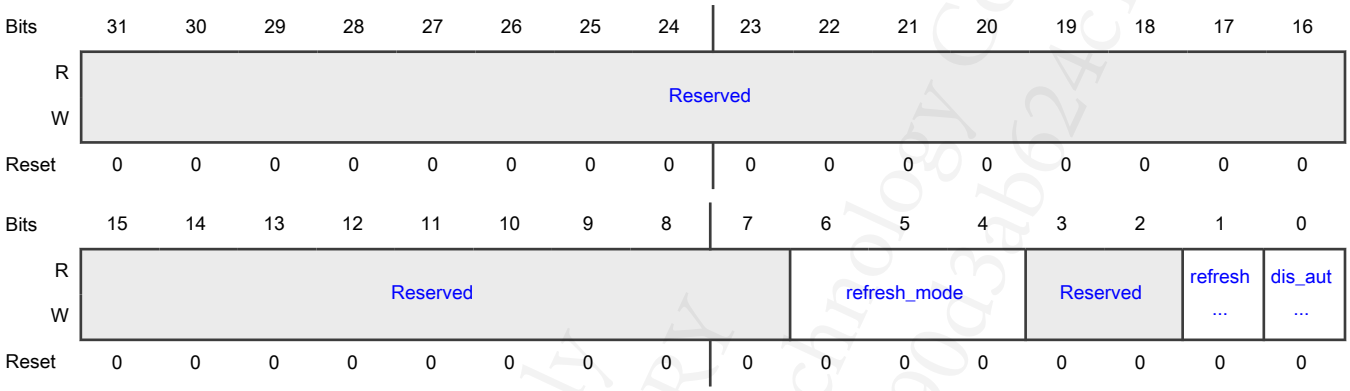
Field	Function
31-28 —	Reserved
27-16 refresh_timer1_start_value_x32	Refresh timer start rank 1 Specifies the relative timing, in multiples of 32 DFI clock cycles, of the refresh of rank 0 (in a multi-rank system). Staggering refreshes to multiple ranks can increase traffic flow. For more information, see Refresh controls. **NOTE** Affects performance only. Programming Mode: Dynamic - Refresh Related
15-12 —	Reserved
11-0 refresh_timer0_start_value_x32	Refresh timer start rank 0 Specifies the relative timing, in multiples of 32 DFI clock cycles, of the refresh of rank 0 (in a multi-rank system). Staggering refreshes to multiple ranks can increase traffic flow. For more information, see Refresh controls. **NOTE** Affects performance only. Programming Mode: Dynamic - Refresh Related

36.14.16 Refresh Control 3 (RFSHCTL3)

Offset

Register	Offset
RFSHCTL3	60h

Diagram



Fields

Field	Function
31-7 —	Reserved
6-4 refresh_mode	Fine granularity refresh mode Selects the Fine Granularity Refresh (FGR) mode of operation. **NOTE** • This field applies only to DDR4 DRAM (MSTR[ddr4]equal to 1). • You must set the FGR mode while the MC is in the Reset or Self Refresh states. Changing FGR mode during normal operation is not allowed. Making this a dynamic register will be supported in future version of the MC. • If auto-refresh is disabled (dis_auto_refresh == 1), only Fixed 1x mode is supported. • The on-the-fly modes are not supported in this version of the MC. Programming Mode: Quasi-dynamic Group 2 000b - Fixed 1x (normal mode) 001b - Fixed 2x 010b - Fixed 4x All other values - Reserved
3-2	Reserved

Table continues on the next page...

Table continued from the previous page...

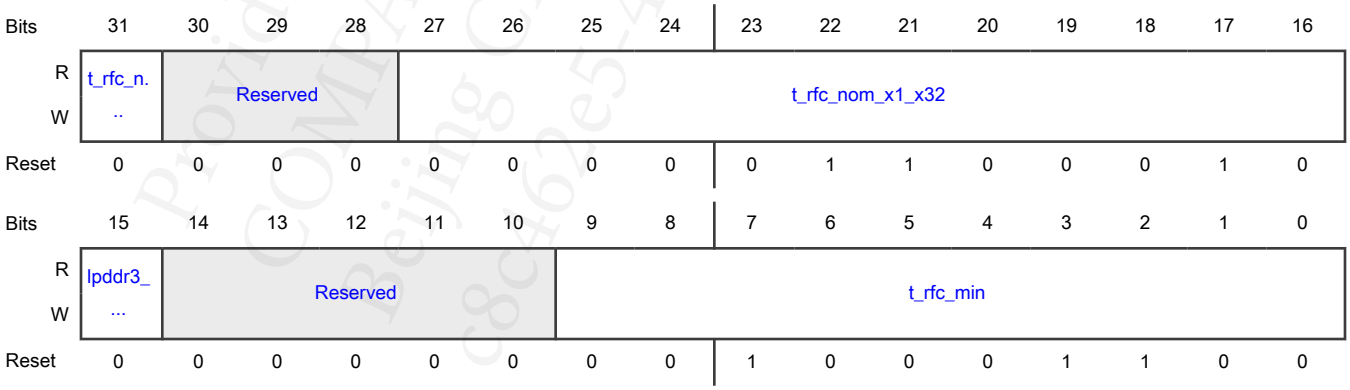
Field	Function
—	
1 refresh_update_level	Refresh update level Specifies by toggling (from 0 to 1 or 1 to 0) that the refresh registers have been updated. The MC automatically updates the refresh registers when exiting reset. **NOTE** Do not toggle this field when the MC is in reset (core_ddrc_rstn == 0). Programming Mode: Dynamic
0 dis_auto_refresh	Auto refresh disable Disables auto-refresh generated by the MC. When auto-refresh is disabled, the SoC core must generate refreshes using DBGCMD[rankn_refresh]. When dis_auto_refresh transitions from 0 to 1, the MC immediately schedules any pending refreshes. **NOTE** • You can change this field on the fly. • For DDR4: If Fine Granularity Refresh is enabled (RFSHCTL3[refresh_mode]), this field must be 0. Programming Mode: Dynamic - Refresh Related 0b - Enable 1b - Disable

36.14.17 Refresh Timing (RFSHTMG)

Offset

Register	Offset
RFSHTMG	64h

Diagram



Fields

Field	Function
31 t_rfc_nom_x1_sel	Refresh units Specifies whether the t_rfc_nom_x1_x32 and RFSHCTL0[refresh_to_x1_x32] values are in DFI clock cycles or multiples of 32 DFI clock cycles (x1 or x32). Programming Mode: Dynamic - Refresh Related
30-28 —	Reserved
27-16 t_rfc_nom_x1_x32	Average refresh interval per rank Specifies the average time interval between refreshes per rank. The specified value is in DFI clock cycles or multiples of 32 DFI clock cycles depending on RFSHTMG[t_rfc_nom_x1_sel]. When the MC is operating at: • 1:1 frequency ratio, set t_rfc_nom_x1_x32 to RoundUp(tRFCI/tCK) • 1:2 frequency ratio, set t_rfc_nom_x1_x32 to RoundUp(RoundUp(tRFCI/tCK)/2) In both cases, if t_rfc_nom_x1_sel == 0, divide the result by 32 and round down. **NOTE** • For DDR2, DDR3, and DDR4: JEDEC specifies 7.8 µs. • For DDR4: The tREFI value depends on the refresh mode. You should program the appropriate value from the specification based on RFSHCTL3[refresh_mode]. • For mDDR, LPDDR2, LPDDR3, and LPDDR4: See the relevant JEDEC specification. • For LPDDR2, LPDDR3, and LPDDR4: — If using all-bank refreshes (RFSHCTL0[per_bank_refresh] equal to 0), use tREFIab in the above calculations. — If using per-bank refreshes (RFSHCTL0[per_bank_refresh] equal to 1), use tREFIpb in the above calculations. **NOTE** • t_rfc_nom_x1_x32 must be greater than 1. • If t_rfc_nom_x1_sel == 1, t_rfc_nom_x1_x32 must be greater than t_rfc_min. • If t_rfc_nom_x1_sel == 0, t_rfc_nom_x1_x32 * 32 must be greater than t_rfc_min. • For non-DDR4 or DDR4 in Fixed 1x mode: t_rfc_nom_x1_x32 must be less than or equal to 0xFFE. • For DDR4 in Fixed 2x mode: t_rfc_nom_x1_x32 must be less than or equal to 0x7FF. • For DDR4 in Fixed 4x mode: t_rfc_nom_x1_x32 must be less than or equal to 0x3FF. Programming Mode: Dynamic - Refresh Related
15 lpddr3_trefbw_en	Burst refresh window parameter enable

Table continues on the next page...

Table continued from the previous page...

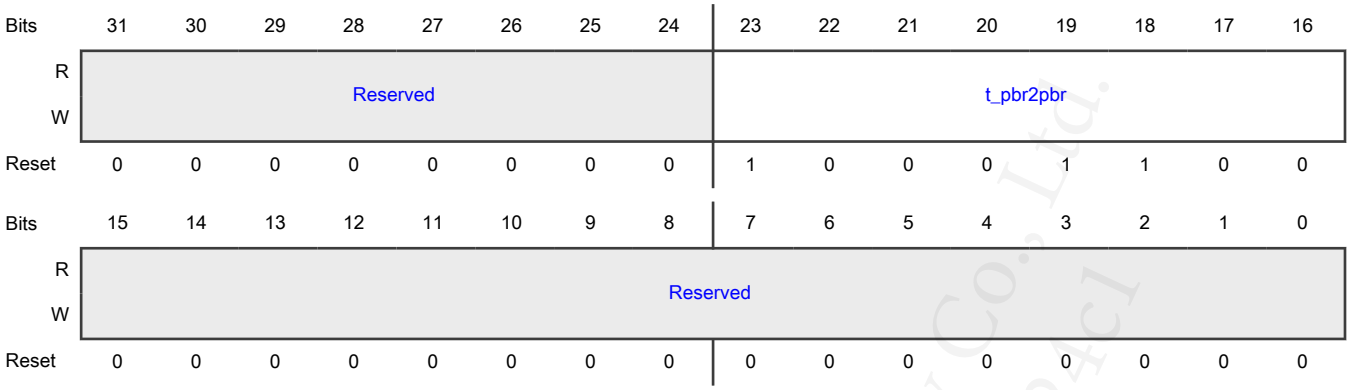
Field	Function
	Specifies whether to use the Burst Refresh Window (tREFBW) parameter. **NOTE** This field applies only to certain LPDDR3 DRAM devices that comply with earlier versions of the JEDEC LPDDR3 specification. Programming Mode: Static 0b - tREFBW not used 1b - tREFBW used
14-10 —	Reserved
9-0 t_rfc_min	Refresh cycle time minimum Specifies the minimum time in DFI clock cycles from refresh to refresh or refresh to activate. When the MC is operating at: • 1:1 frequency ratio, set t_rfc_min to RoundUp(tRFCmin/tCK) • 1:2 frequency ratio, set t_rfc_min to RoundUp(RoundUp(tRFCmin/tCK)/2) **NOTE** • For LPDDR2, LPDDR3, and LPDDR4: — If using all-bank refreshes, the tRFCmin value in the above equations is equal to tRFCab. — If using per-bank refreshes, the tRFCmin value in the above equations is equal to tRFCpb. • For DDR4: The tRFCmin value in the above equations depends on the refresh mode (fixed 1X, 2X, or 4X) and the device density. Programming Mode: Dynamic - Refresh Related

36.14.18 Refresh timing 1 (RFSHTMG1)

Offset

Register	Offset
RFSHTMG1	68h

Diagram



Fields

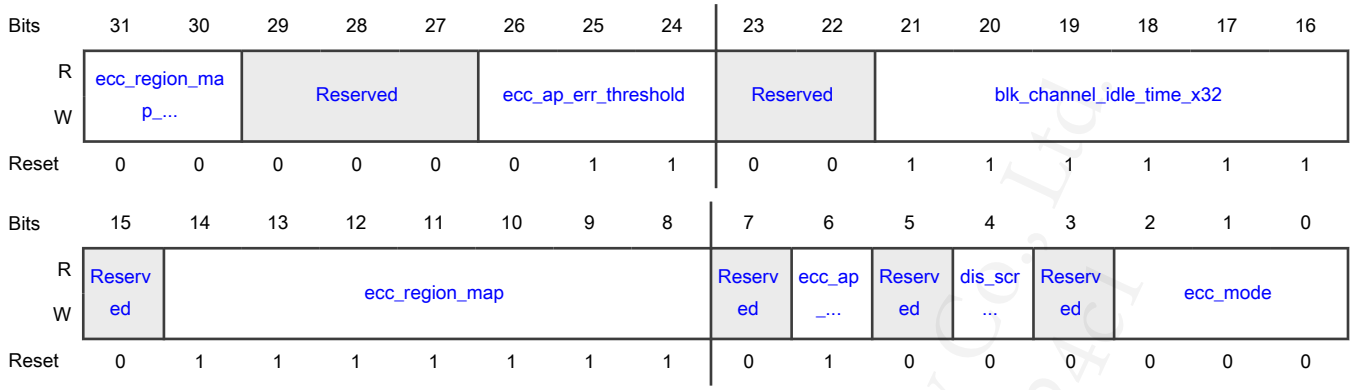
Field	Function
31-24 —	Reserved
23-16 t_pbr2pbr	Per-bank refresh to per-bank refresh different bank time Specifies the time interval between a per-bank refresh and a per-bank refresh of a different bank. When the MC is operating at: • 1:1 frequency ratio, set t_pbr2pbr to RoundUp(tpbR2pbR/tCK) • 1:2 frequency ratio, set t_pbr2pbr to RoundUp(RoundUp(tpbR2pbR/tCK)/2) **NOTE** • This field applies only to LPDDR DRAM in per-bank refresh (RFSHCTL0[per_bank_refresh]). • The tpbR2pbR value in the above equations depends on the device density. You should program the appropriate value from the JEDEC LPDDR4 spec. Programming Mode: Dynamic - Refresh Related
15-0 —	Reserved

36.14.19 ECC Configuration 0 (ECCCFG0)

Offset

Register	Offset
ECCCFG0	70h

Diagram



Fields

Field	Function
31-30 ecc_region_map_granu	Selectable protected region granularity Specifies the portion of memory space occupied by each selectable protected region in the ECC region map (ecc_region_map). Programming Mode: Static 00b - 1/8 of memory space 01b - 1/16 of memory space 10b - 1/32 of memory space 11b - 1/64 of memory space
29-27 —	Reserved
26-24 ecc_ap_err_threshold	Address parity error threshold Specifies the number of ECC errors (correctable or uncorrectable) that can occur within one burst before the MC indicates an ECC address parity error (ECCAPSTAT[ecc_ap_err]). The value must be less than the total number of ECC checks within one burst: (DRAM Data width) x (DRAM BL)/64 Programming Mode: Static
23-22 —	Reserved
21-16 blk_channel_idle_time_x32	Block channel idle time Specifies the time interval, in multiples of 32 DFI clock cycles, of no access through the host interface (HIF) to protected regions that causes the MC to flush all block channels. To flush a block channel, the MC injects a write ECC command if there is any write in the block and then stops tracking the block address. Programming Mode: Quasi-dynamic Group 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000000b - Invalid value.
15 —	Reserved
14-8 ecc_region_map	Selectable protected region map Specifies which of the lowest seven memory regions (the selectable protected regions) are protected with ECC. Each bit corresponds to each of the lowest seven regions, with the lsb corresponding to the lowest address region. Unselected regions (corresponding bit == 0) are unprotected. This field interacts with ecc_region_map_granu, which specifies how many regions the total memory space is divided into: 8, 16, 32, or 64 regions. For example: To protect all of the memory space: • ecc_region_map = 0b1111111 • ecc_region_map_granu = 0 (eight total regions) To protect only the first 1/64 of the memory space: • ecc_region_map = 0b0000001 • ecc_region_map_granu = 3 (64 total regions) In this last example, the region map is: • Regions 63–56 are dedicated ECC. • Regions 55–7 are not protected (not in the selectable protected regions) • Regions 6–1 are not protected (corresponding bits = 0) • Region 0 is protected (corresponding bit = 1) **NOTE** • The highest 1/8 of the memory space is always an ECC region. • If inline ECC is enabled (ecc_mode == 1), 0b0000000 is an invalid value. You must specify at least one protected region. Programming Mode: Quasi-dynamic Group 3
7 —	Reserved
6 ecc_ap_en	Address protection enable Enables address protection. This field applies only in inline ECC mode. Programming Mode: Static 0b - Disable 1b - Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 —	Reserved
4 dis_scrub	ECC scrubs disable Disables ECC scrubs. **NOTE** • This field applies only when ECCCFG0[ecc_mode] == 3'b100 or 3'b101 and MEMC_USE_RMW is defined. • In inline ECC mode, scrub is not supported and this field is ignored. Programming Mode: Static 0b - Enable 1b - Disable
3 —	Reserved
2-0 ecc_mode	ECC mode Speifies the ECC mode. Programming Mode: Static 000b - ECC disabled 100b - ECC enabled - SEC/DED over 1 beat 101b - ECC enabled - Advanced ECC All other values - Reserved

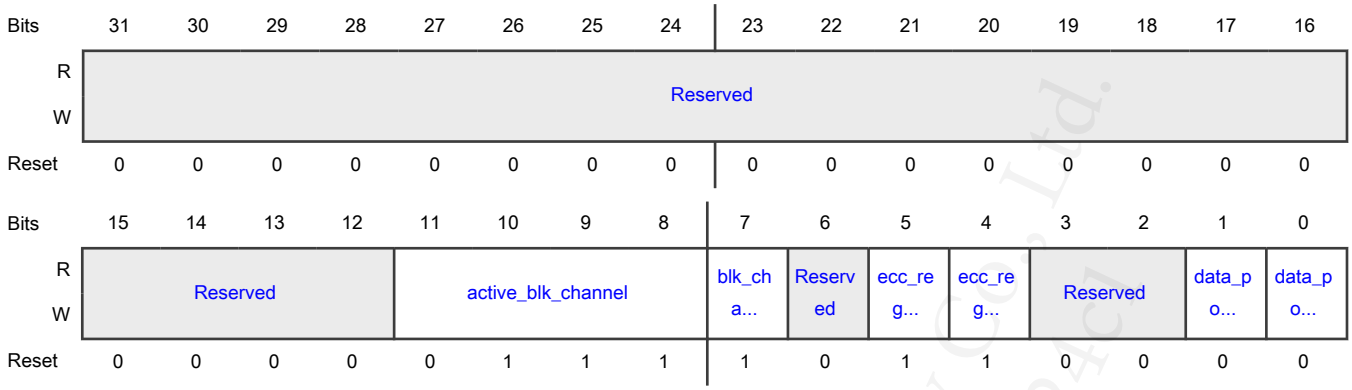
36.14.20 ECC Configuration 1 (ECCCFG1)

Offset

Register	Offset
ECCCFG1	74h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-12 —	Reserved
11-8 active_blk_channel	Active block channels Specifies the number of active block channels. Use this value to limit the number of available channels. For example, set this value to 0 to have only one active channel, and therefore disable block interleaving. Programming Mode: Quasi-dynamic Group 3
7 blk_channel_active_term	Block channel active terminate enable Enables termination of block channel on a full-block read or write (all addresses within the block are read or written). Programming Mode: Static 0b - Disable (use only for debug purposes) 1b - Enable (default for normal operation)
6 —	Reserved
5 ecc_region_waste_lock	ECC waste region lock Locks the portion of the ECC region that is not either the ECC parity section or a designated selectable protected region. When locked, an access of this region generates an error. When unlocked, you can access the region like any other non-ECC protected region. 0b - Unlocked 1b - Locked
4 ecc_region_parity_lock	ECC parity section lock

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Locks the parity section of the ECC region, the highest 1/8 of memory address space. This region stores ECC parity data for the protected region. When locked, an access of this region generates an error. When unlocked, you can access the region like any other non-ECC protected region. Programming Mode: Quasi-dynamic Group 3 0b - Unlocked 1b - Locked
3-2 —	Reserved
1 data_poison_bit	Data poison bits Selects one or two bits of data poisoning. Applies only if ECC data poisoning is enabled (data_poison_en). Programming Mode: Quasi-dynamic Group 3 0b - Two bits (uncorrectable) 1b - One bit (correctable)
0 data_poison_en	ECC data poisoning enable Enables ECC data poisoning. When ECC data poisoning is enabled, the MC introduces errors on writes to the address specified by ECCPOISONADDR0 and ECCPOISONADDR1. Programming Mode: Quasi-dynamic Group 3 "ECC Data Poisoning is not supported with INLINE-ECC feature (you must write 0 to ECCCFG1[data_poison_en]). There is a software method to inject errors to ECC parity bits with inline ECC. **Software method:** Errors can be easily injected through the software by unlocking the ECC region through the ecc_region_parity_lock register and overriding certain ECC parity bits. When the corresponding addresses are read from a protected memory region, ECC errors are generated as correctable or uncorrectable, depending on the type of error introduced."

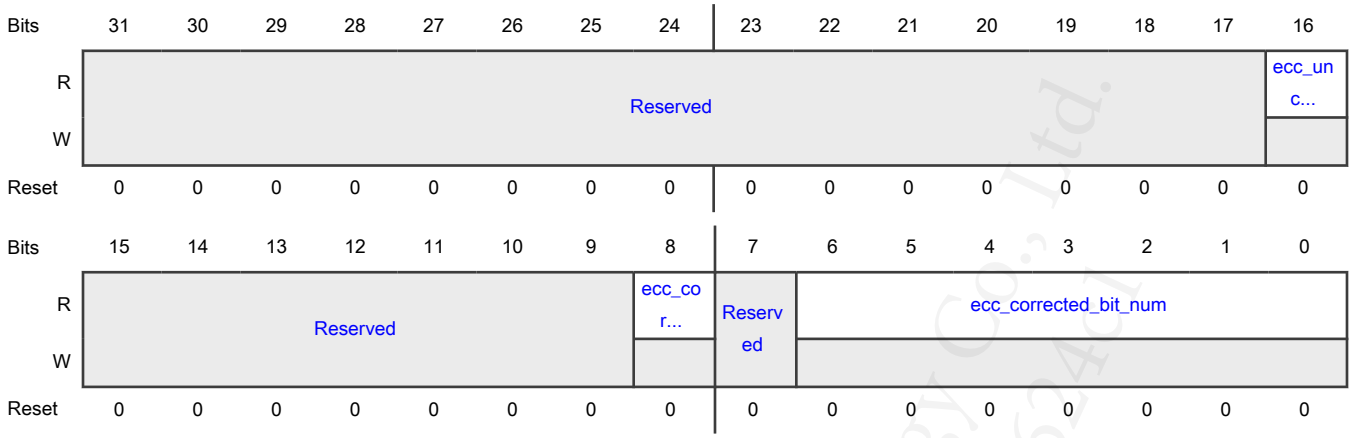
36.14.21 SECDED ECC Status (ECCSTAT)

Offset

Register	Offset
ECCSTAT	78h

Function
(Valid only in MEMC_ECC_SUPPORT==1 (SECDED ECC mode))

Diagram



Fields

Field	Function
31-17 —	Reserved
16 ecc_uncorrected_err	Double-bit error indicator Indicates an uncorrectable double-bit ECC error on any lane. Programming Mode: Dynamic 0b - No error 1b - Double-bit error
15-9 —	Reserved
8 ecc_corrected_err	Single-bit error indicator Indicates a correctable single-bit ECC error on any lane. Programming Mode: Dynamic 0b - No error 1b - Single-bit error
7 —	Reserved
6-0 ecc_corrected_bit_num	ECC corrected bit number Indicates the bit number corrected by a single-bit ECC error. Programming Mode: Dynamic

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e2e5-4821-48a8-b251-390d3ab6140f

36.14.22 ECC Control (ECCCTL)

Offset

Register	Offset
ECCCTL	7Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved													ecc_ap_...	ecc_unc...	ecc_cor...
W														W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					ecc_ap_...	ecc_unc...	ecc_cor...	Reserved			ecc_ap_...	ecc_unc...	ecc_cor...	ecc_unc...	ecc_cor...
W												W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18 ecc_ap_err_intr_force	ECC address protection error interrupt force Forces assertion of the ECC address protection error interrupt. There is no interaction between functionally triggering an interrupt and forcing an interrupt; the two actions are mutually exclusive. The MC returns this bit to zero automatically. Programming Mode: Dynamic 0b - No effect 1b - Force the interrupt
17 ecc_uncorrected_err_intr_force	Uncorrected ECC error interrupt force Forces assertion of the uncorrected ECC error interrupt. There is no interaction between functionally triggering an interrupt and forcing an interrupt; the two actions are mutually exclusive. The MC returns this bit to zero automatically. Programming Mode: Dynamic 0b - No effect 1b - Force the interrupt

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 ecc_corrected_err_intr_force	Corrected ECC error interrupt force Forces assertion of the corrected ECC error interrupt. There is no interaction between functionally triggering an interrupt and forcing an interrupt; the two actions are mutually exclusive. The MC returns this bit to zero automatically. Programming Mode: Dynamic 0b - No effect 1b - Force the interrupt
15-11 —	Reserved
10 ecc_ap_err_intr_en	ECC address protection error interrupt enable Enables the ECC address protection error interrupt. Programming Mode: Dynamic 0b - Disable 1b - Enable
9 ecc_uncorrected_err_intr_en	Uncorrected ECC error interrupt enable Enables the uncorrected ECC error interrupt. Programming Mode: Dynamic 0b - Disable 1b - Enable
8 ecc_corrected_err_intr_en	Corrected ECC error interrupt enable Enables the corrected ECC error interrupt. Programming Mode: Dynamic 0b - Disable 1b - Enable
7-5 —	Reserved
4 ecc_ap_err_intr_clr	Clear ECC address protection error interrupt Clears ECCAPSTAT[ecc_ap_err]. The MC returns this bit to zero automatically. Programming Mode: Dynamic 0b - No effect 1b - Clear the ECC address protection error interrupt

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ecc_uncorr_err_cnt_clr	Clear uncorrected ECC error count Clears ECCERRCNT[ecc_uncorr_err_cnt]. The MC returns this bit to zero automatically. Programming Mode: Dynamic 0b - No effect 1b - Clear the uncorrected ECC error count
2 ecc_corr_err_cnt_clr	Clear corrected ECC error count Clears ECCERRCNT[ecc_corr_err_cnt]. The MC returns this bit to zero automatically. Programming Mode: Dynamic 0b - No effect 1b - Clear the corrected ECC error count
1 ecc_uncorrected_err_clr	Clear uncorrected ECC error Clears the following registers and fields: • ECCSTAT[ecc_uncorrected_err] • ECCUSYN0 • ECCUSYN1 • ECCUSYN2 The MC returns this bit to zero automatically. Programming Mode: Dynamic 0b - No effect 1b - Clear the uncorrected ECC error
0 ecc_corrected_err_clr	Clear corrected ECC error Clears the following registers and fields: • ECCSTAT[ecc_corrected_err] • ECCSTAT[ecc_corrected_bit_num] • ECCCADDR0 • ECCCADDR1 • ECCCSYN0 • ECCCSYN1 • ECCCSYN2 • ECCBITMASK0

Table continues on the next page...

Table continued from the previous page...

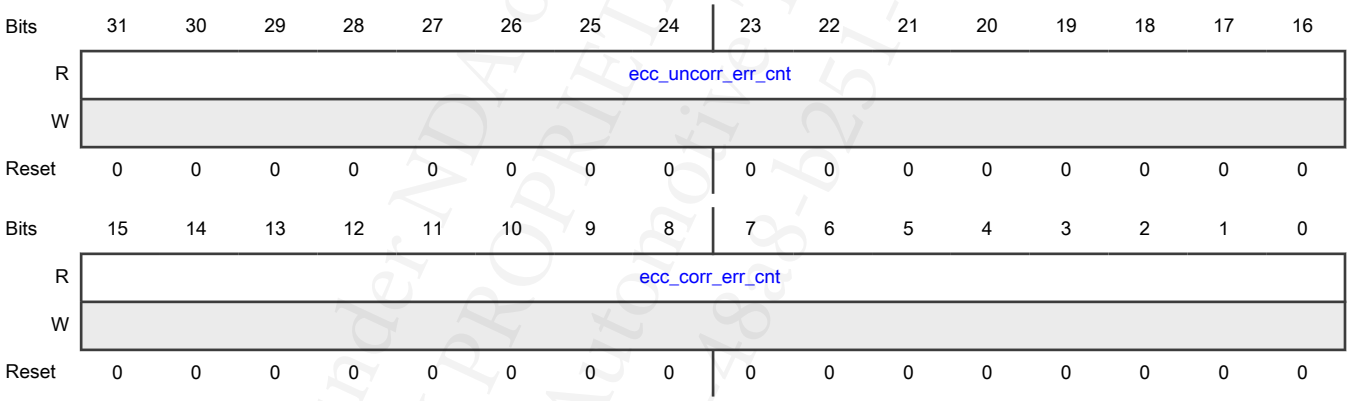
Field	Function
	• ECCBITMASK1 • ECCBITMASK2 The MC returns this bit to zero automatically. Programming Mode: Dynamic 0b - No effect 1b - Clear the corrected ECC error

36.14.23 ECC Error Counter (ECCERRCNT)

Offset

Register	Offset
ECCERRCNT	80h

Diagram



Fields

Field	Function
31-16 ecc_uncorr_err_cnt	Uncorrectable ECC errors count Indicates the number of detected uncorrectable ECC errors. For inline ECC, this field saturates at FFFFh. For side-band ECC with SECDED ECC, it saturates at FFFCh or above. Programming Mode: Dynamic
15-0	Correctable ECC errors count Indicates the number of detected correctable ECC errors.

Table continues on the next page...

Table continued from the previous page...

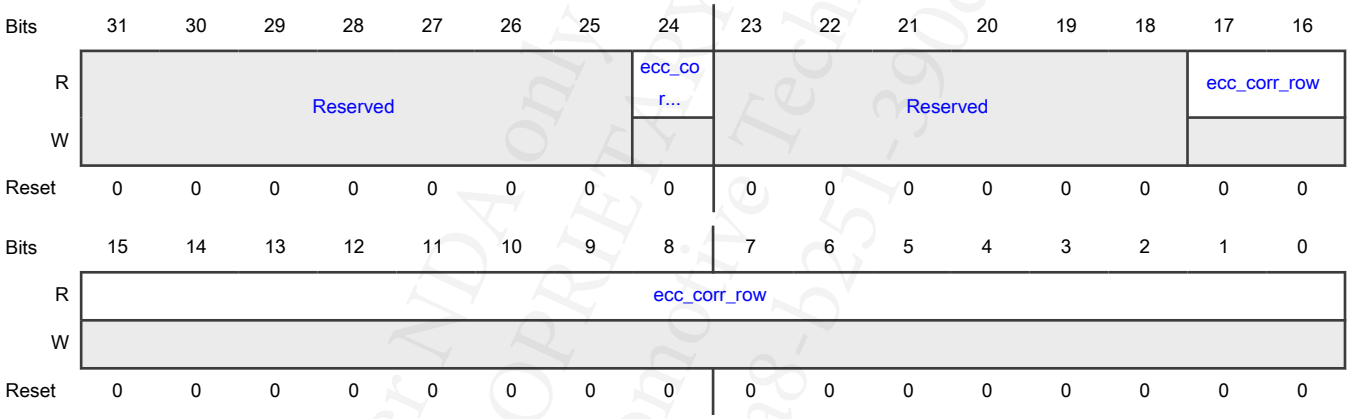
Field	Function
ecc_corr_err_cnt	For inline ECC, this field saturates at FFFFh. For side-band ECC with SECDED ECC, it saturates at FFFCh or above. Programming Mode: Dynamic

36.14.24 ECC Corrected Error Address 0 (ECCCADDR0)

Offset

Register	Offset
ECCCADDR0	84h

Diagram



Fields

Field	Function
31-25 —	Reserved
24 ecc_corr_rank	Rank number Indicates the rank number of a read resulting in a corrected ECC error. Programming Mode: Dynamic
23-18 —	Reserved
17-0	Page/row number Indicates the page/row number of a read resulting in a corrected ECC error.

Table continues on the next page...

Table continued from the previous page...

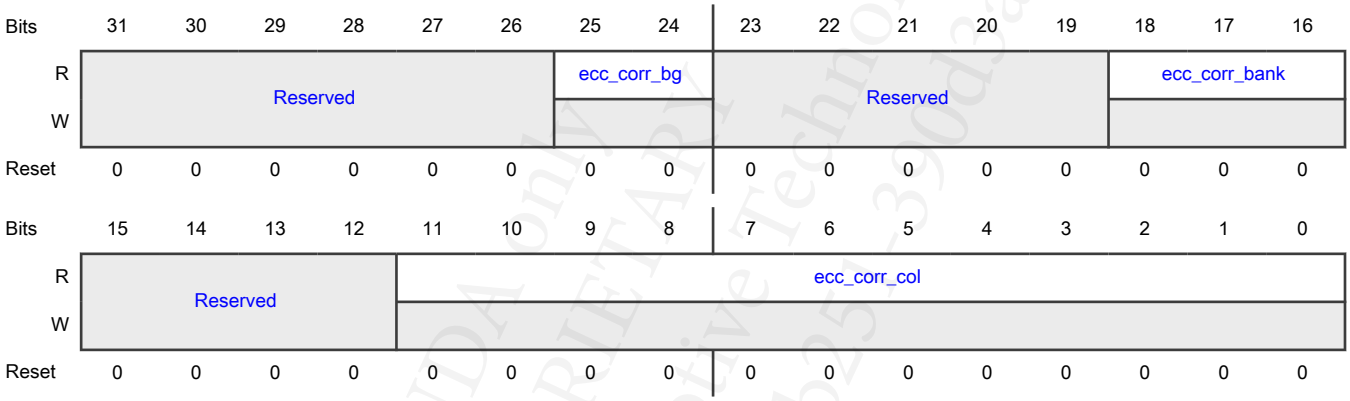
Field	Function
ecc_corr_row	Programming Mode: Dynamic

36.14.25 ECC Corrected Error Address 1 (ECCCADDR1)

Offset

Register	Offset
ECCCADDR1	88h

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24 ecc_corr_bg	Bank group Indicates the bank group number of a read resulting in a corrected ECC error. Programming Mode: Dynamic
23-19 —	Reserved
18-16 ecc_corr_bank	Bank number Indicates the bank number of a read resulting in a corrected ECC error. Programming Mode: Dynamic
15-12	Reserved

Table continues on the next page...

Table continued from the previous page...

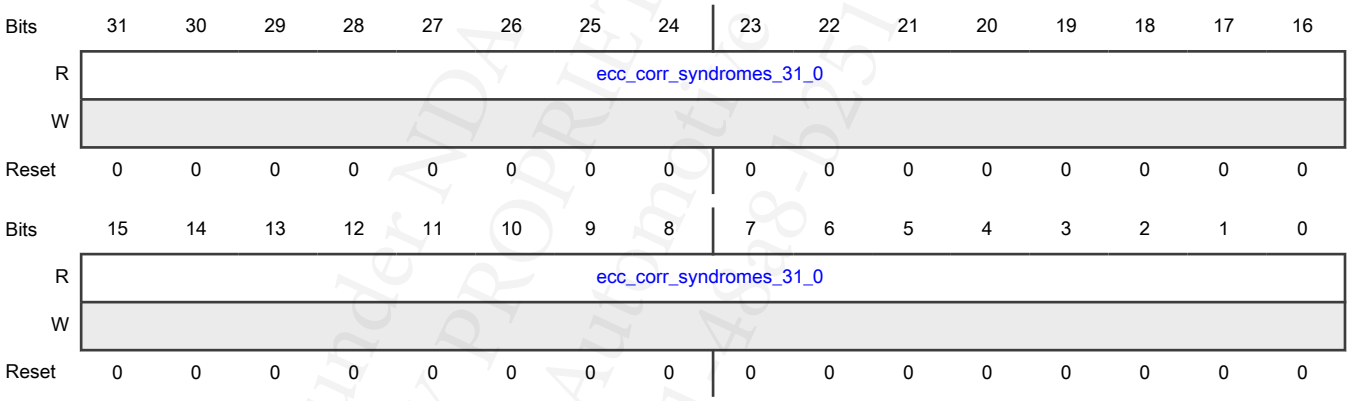
Field	Function
—	
11-0 ecc_corr_col	Block number Indicates the block number of a read resulting in a corrected ECC error. **NOTE** The lsb is not shown. Programming Mode: Dynamic

36.14.26 ECC Corrected Syndrome 0 (ECCCSYN0)

Offset

Register	Offset
ECCCSYN0	8Ch

Diagram



Fields

Field	Function
31-0 ecc_corr_syndromes_31_0	Corrected ECC error data pattern: bits 31-0 Indicates the least significant 32 bits of a data pattern that resulted in a corrected ECC error. 16-bit ECC uses only the 16 least significant bits. Programming Mode: Dynamic

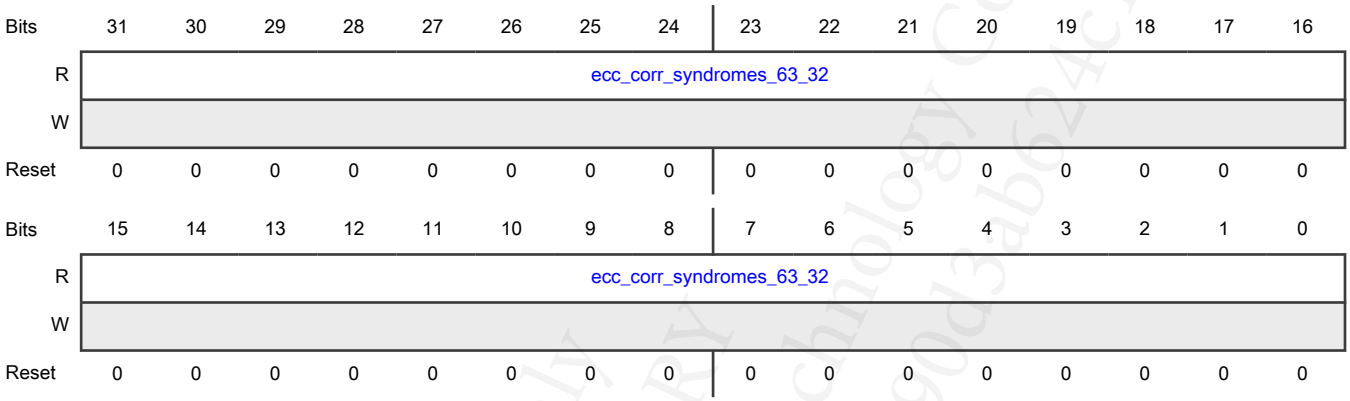
Printed under NDA only
COMPANY PROPRIETARY
Bellus CHJ Automotive Technology Co., Ltd.
c8c4623c-4821-4e38-b251-390d3ab624c1

36.14.27 ECC Corrected Syndrome 1 (ECCCSYN1)

Offset

Register	Offset
ECCCSYN1	90h

Diagram



Fields

Field	Function
31-0 ecc_corr_syndromes_63_32	Corrected ECC error data pattern: bits 63-31 Indicates the most significant 32 bits of a 64-bit data pattern that resulted in a corrected ECC error. 16-bit and 32-bit ECC do not use this field. Programming Mode: Dynamic

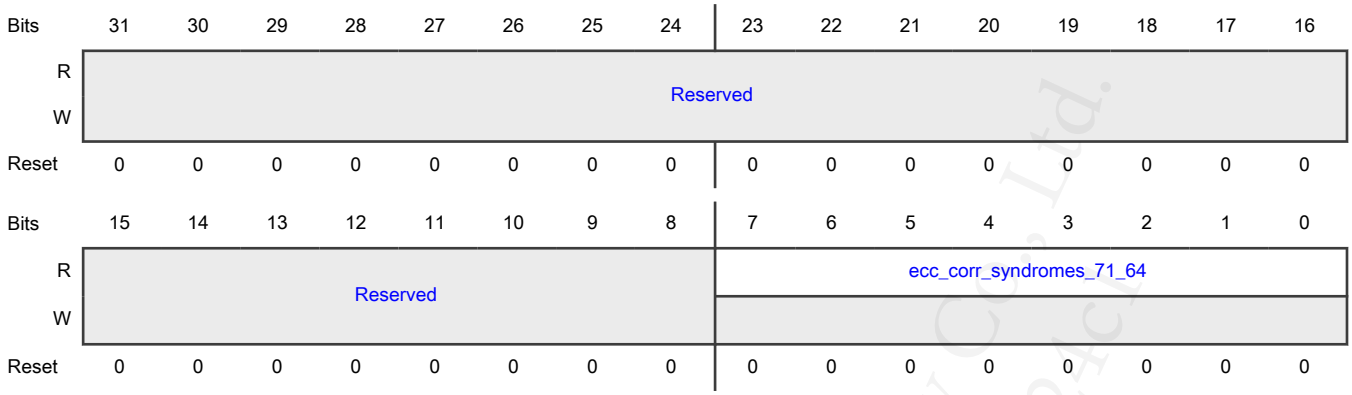
36.14.28 ECC Corrected Syndrome 2 (ECCCSYN2)

Offset

Register	Offset
ECCCSYN2	94h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-9251-390d3ab624c1

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 ecc_corr_syndromes_71_64	Corrected ECC error data pattern: ECC byte Indicates the ECC byte for an ECC lane that resulted in a corrected ECC error. This byte's position in the ECC lane depends on the ECC width: • For 64-bit ECC: bits 71-64 • For 32-bit ECC: bits 39-32 • For 16-bit ECC: bits 23-16 Programming Mode: Dynamic

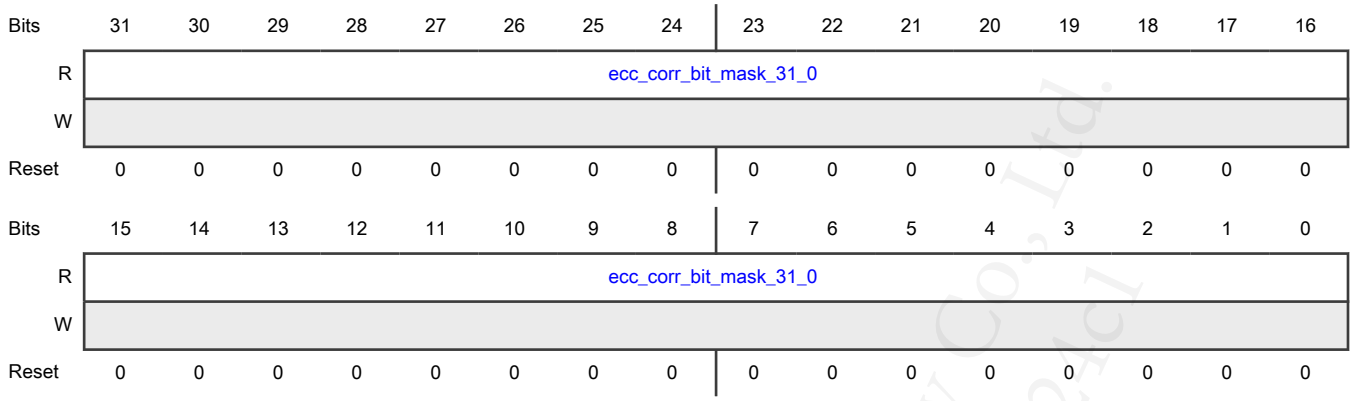
36.14.29 ECC Corrected Data Bit Mask 0 (ECCBITMASK0)

Offset

Register	Offset
ECCBITMASK0	98h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4848-b251-390d3ab624c6

Diagram



Fields

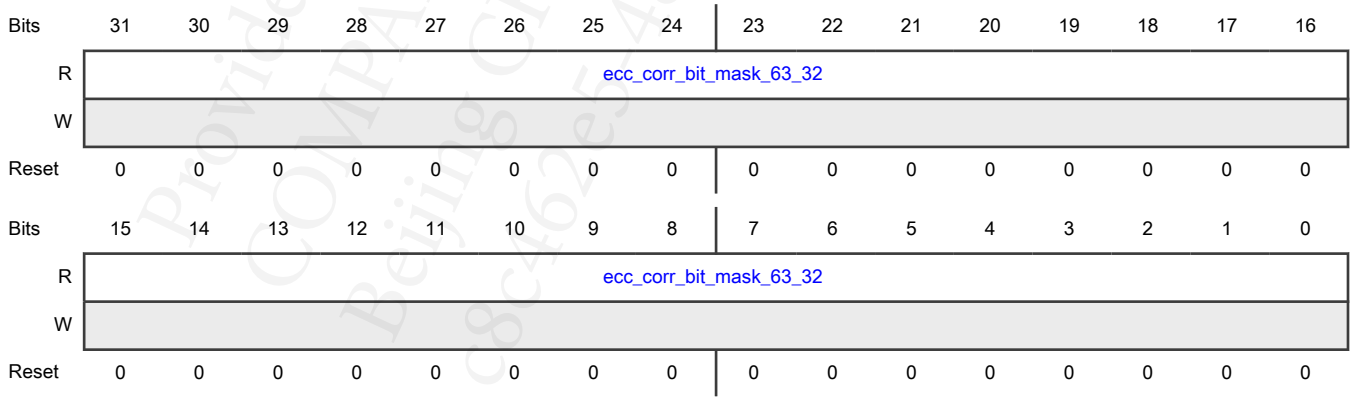
Field	Function
31-0 ecc_corr_bit_mask_31_0	Corrected data bit mask: bits 31-0 Indicates the corrected bits in the least signifcant 32 bits of the data portion of an ECC lane: 1 indicates that the bit has been corrected; 0 indicates that the bit has not been corrected. 16-bit ECC uses only the least significant 16 bits. This field accumulates data over multiple ECC errors to give an overall indication of which bits are being corrected. To clear this field, write a 1 to ECCCTL[ecc_corrected_err_clr] Programming Mode: Dynamic

36.14.30 ECC Corrected Data Bit Mask 1 (ECCBITMASK1)

Offset

Register	Offset
ECCBITMASK1	9Ch

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
08c462e54821148a8b2513d3ab624c1

Fields

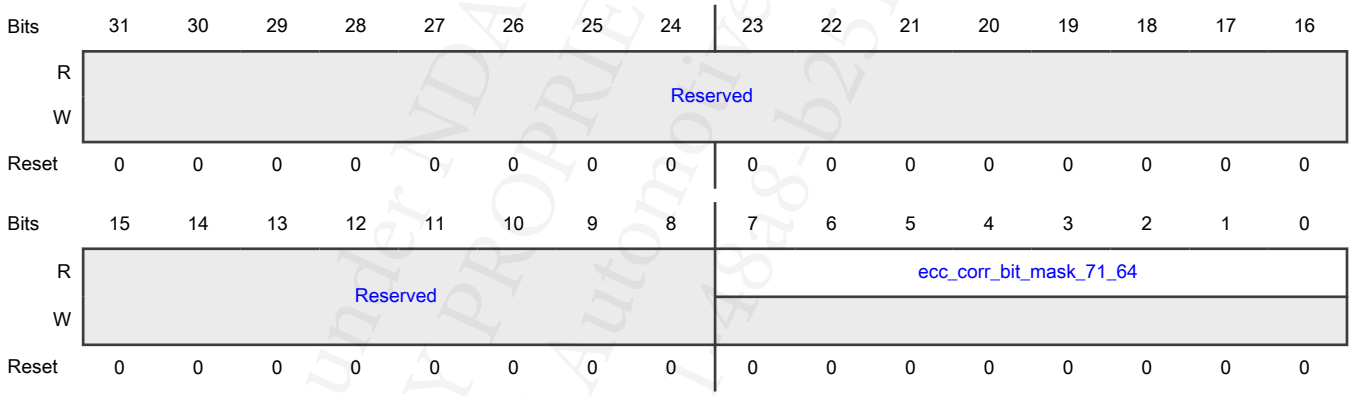
Field	Function
31-0 ecc_corr_bit_mask_63_32	Corrected data bit mask: bits 63-32 Indicates the corrected bits in the most signifcant 32 bits of the data portion of an ECC lane: 1 indicates that the bit has been corrected; 0 indicates that the bit has not been corrected. 16-bit and 32-bit ECC does not use this field. This field accumulates data over multiple ECC errors to give an overall indication of which bits are being corrected. To clear this field, write a 1 to ECCCTL[ecc_corrected_err_clr] Programming Mode: Dynamic

36.14.31 ECC Corrected Data Bit Mask 2 (ECCBITMASK2)

Offset

Register	Offset
ECCBITMASK2	A0h

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 ecc_corr_bit_mask_71_64	Corrected data bit mask: ECC byte Indicates the corrected bits in the ECC byte of an ECC lane: 1 indicates that the bit has been corrected; 0 indicates that the bit has not been corrected. This byte's position in the ECC lane depends on the ECC width:

Table continues on the next page...

Table continued from the previous page...

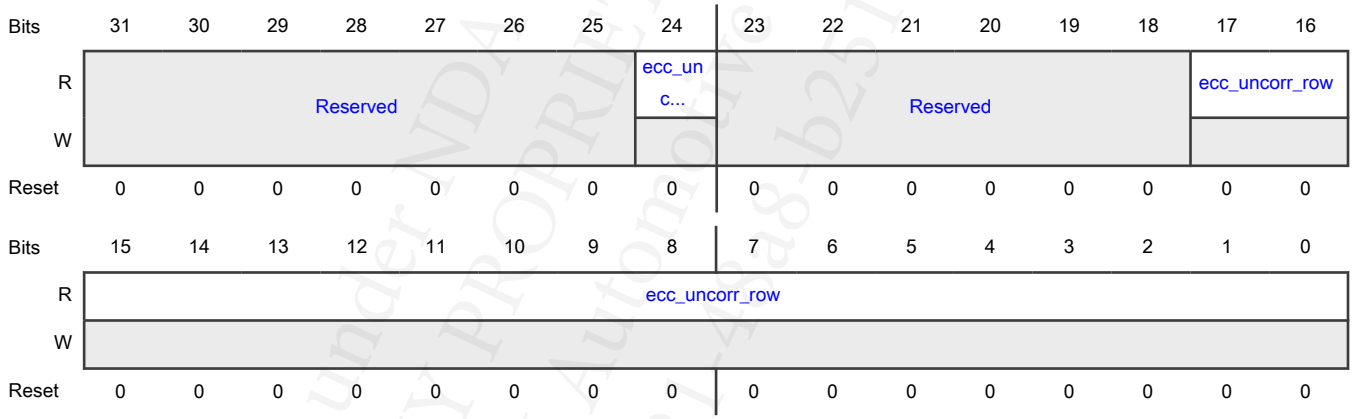
Field	Function
	• For 64-bit ECC: bits 71-64 • For 32-bit ECC: bits 39-32 • For 16-bit ECC: bits 23-16 This field accumulates data over multiple ECC errors to give an overall indication of which bits are being corrected. To clear this field, write a 1 to ECCCTL[ecc_corrected_err_clr] Programming Mode: Dynamic

36.14.32 ECC Uncorrected Error Address 0 (ECCUADDR0)

Offset

Register	Offset
ECCUADDR0	A4h

Diagram



Fields

Field	Function
31-25 —	Reserved
24 ecc_uncorr_rank	Rank number Indicates the rank number of a read resulting in an uncorrected ECC error. Programming Mode: Dynamic

Table continues on the next page...

Table continued from the previous page...

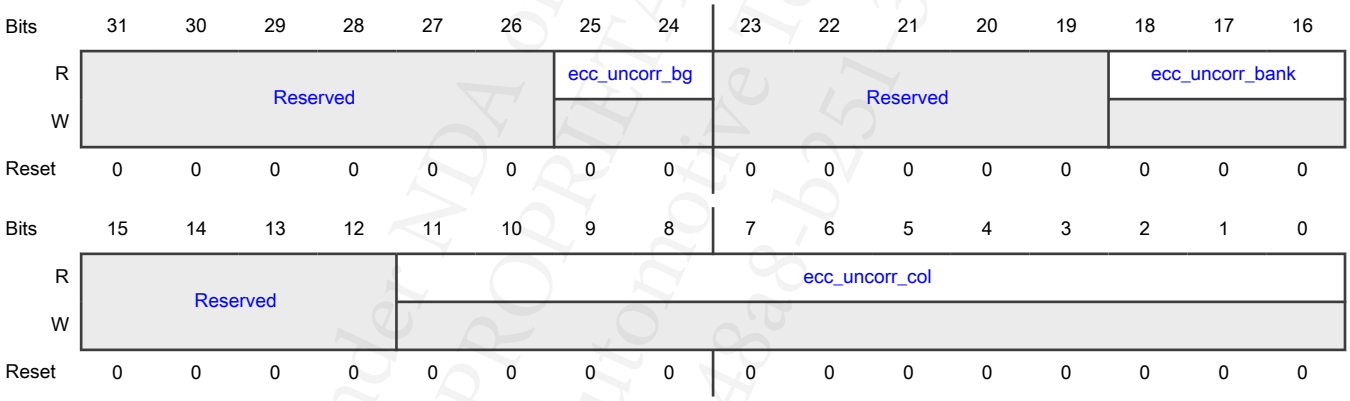
Field	Function
23-18 —	Reserved
17-0 ecc_uncorr_row	Page/row number Indicates the page/row number of a read resulting in an uncorrected ECC error. Programming Mode: Dynamic

36.14.33 ECC Uncorrected Error Address 1 (ECCUADDR1)

Offset

Register	Offset
ECCUADDR1	A8h

Diagram



Fields

Field	Function
31-26 —	Reserved
25-24 ecc_uncorr_bg	Bank group Indicates the bank group number of a read resulting in an uncorrected ECC error. Programming Mode: Dynamic
23-19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18-16 ecc_uncorr_bank	Bank number Indicates the bank number of a read resulting in an uncorrected ECC error. Programming Mode: Dynamic
15-12 —	Reserved
11-0 ecc_uncorr_col	Block number Indicates the block number of a read resulting in an uncorrected ECC error. **NOTE** The lsb is not shown. Programming Mode: Dynamic

36.14.34 ECC Uncorrected Syndrome 0 (ECCUSYN0)

Offset

Register	Offset
ECCUSYN0	ACh

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ecc_uncorr_syndromes_31_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ecc_uncorr_syndromes_31_0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0	Uncorrected ECC error data pattern: bits 31-0

Table continues on the next page...

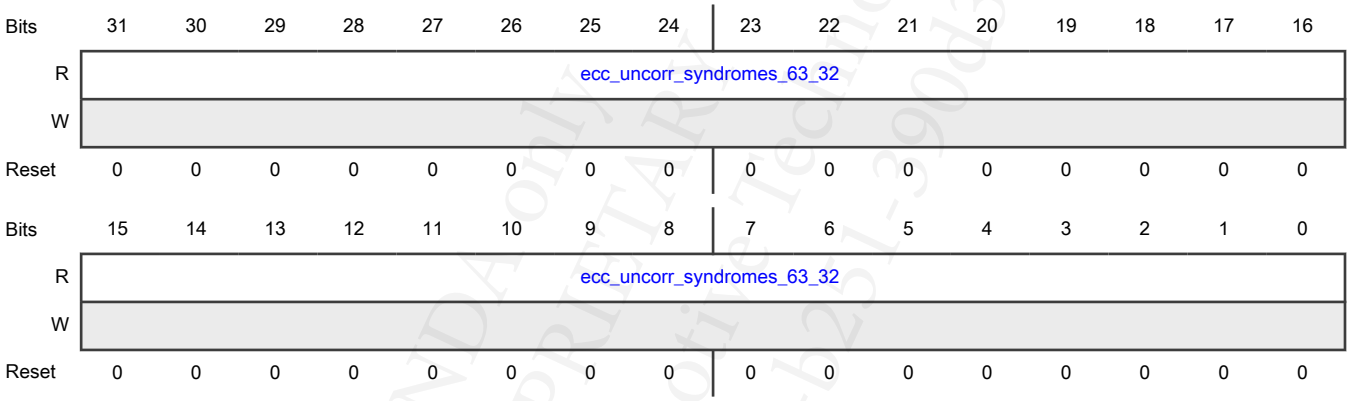
Field	Function
ecc_uncorr_syndromes_31_0	Indicates the least significant 32 bits of a data pattern that resulted in an uncorrected ECC error. 16-bit ECC uses only the 16 least significant bits. Programming Mode: Dynamic

36.14.35 ECC Uncorrected Syndrome 1 (ECCUSYN1)

Offset

Register	Offset
ECCUSYN1	B0h

Diagram



Fields

Field	Function
31-0 ecc_uncorr_syndromes_63_32	Uncorrected ECC error data pattern: bits 63-31 Indicates the most significant 32 bits of a 64-bit data pattern that resulted in an uncorrected ECC error. 16-bit and 32-bit ECC do not use this field. Programming Mode: Dynamic

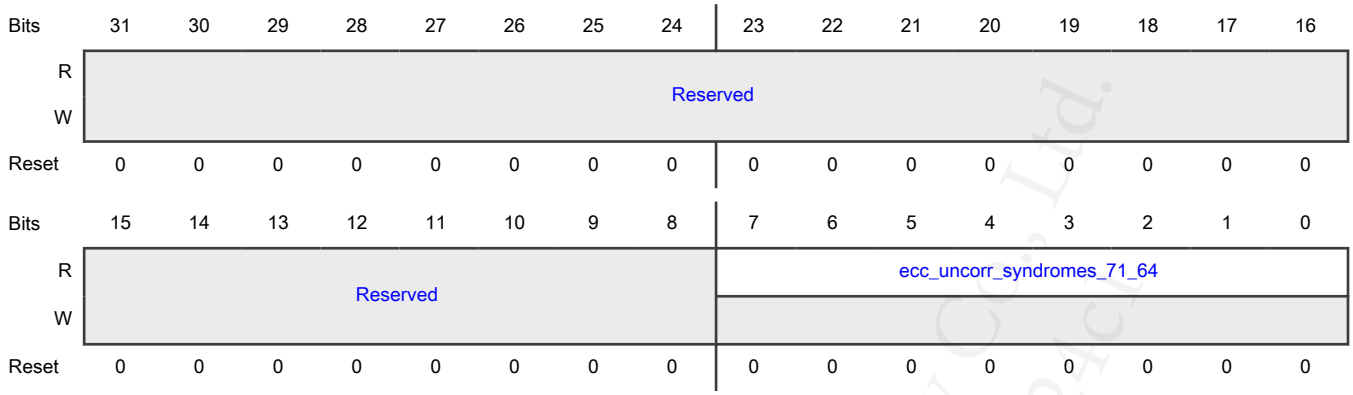
36.14.36 ECC Uncorrected Syndrome 2 (ECCUSYN2)

Offset

Register	Offset
ECCUSYN2	B4h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHTC Automotive Technology Co., Ltd.
ca462e54-8214-48a8-b251-290d3ab624c1

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 ecc_uncorr_syndromes_71_64	Uncorrected ECC error data pattern: ECC byte Indicates the ECC byte for an ECC lane that resulted in an uncorrected ECC error. This byte's position in the ECC lane depends on the ECC width: • For 64-bit ECC: bits 71-64 • For 32-bit ECC: bits 39-32 • For 16-bit ECC: bits 23-16 Programming Mode: Dynamic

36.14.37 ECC Data Poisoning Address 0 (ECCPOISONADDR0)

Offset

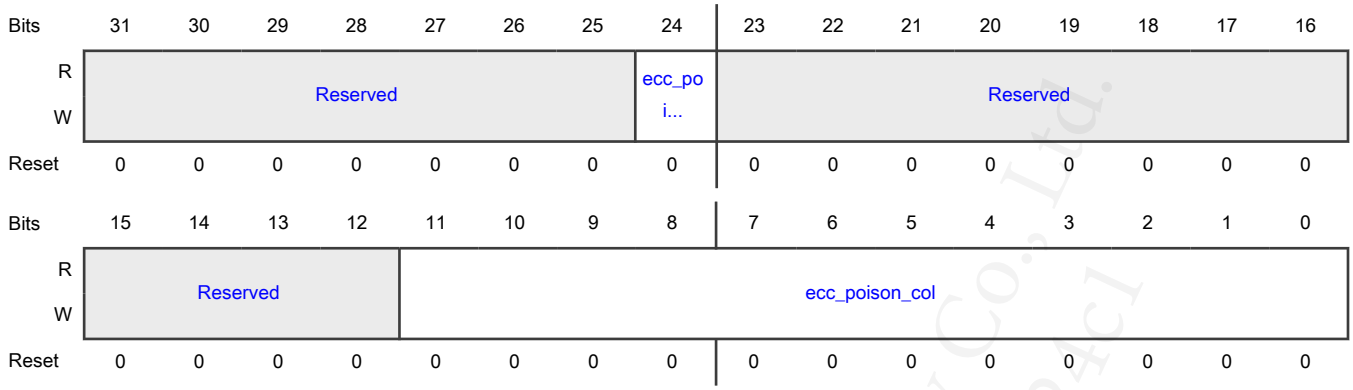
Register	Offset
ECCPOISONADDR0	B8h

Function

When ECC data poisoning is enabled (ECCCFG1[data_poison_en]), if a HIF write data beat matches the address in ECCPOISONADDR0 and ECCPOISONADDR1, the MC introduces an ECC error on that transaction (write/RMW).

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4626e-4821-4848-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-25 —	Reserved
24 ecc_poison_rank	ECC poisoning rank address Specifies the rank addres for ECC poisoning. Programming Mode: Static
23-12 —	Reserved
11-0 ecc_poison_col	ECC poisoning column address Specifies the column addres for ECC poisoning. This address must be burst-aligned: • For full bus width: bits 2-0 must be 000b • For half bus width: bits 3-0 must be 0000b • For quarter bus width: bits 4-0 must be 00000b Programming Mode: Static

36.14.38 ECC Data Poisoning Address 1 (ECCPOISONADDR1)

Offset

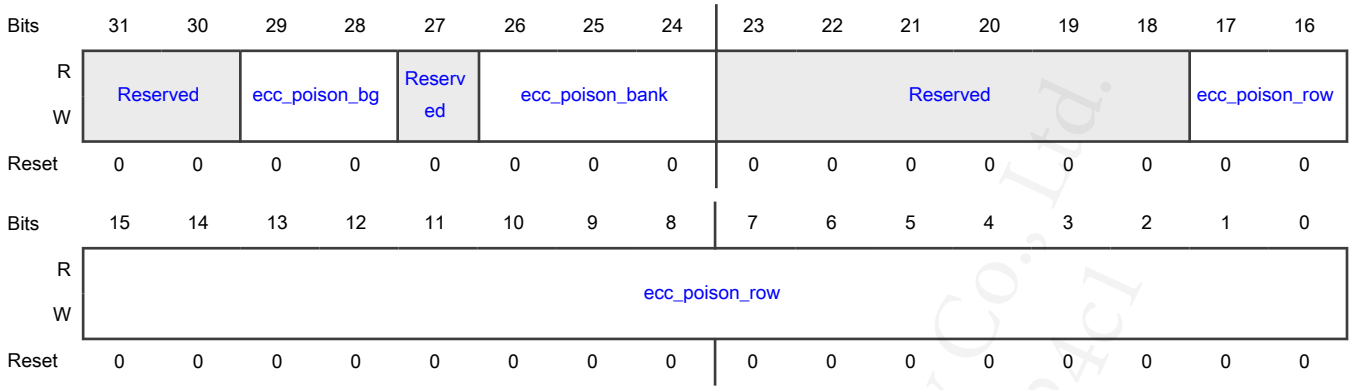
Register	Offset
ECCPOISONADDR1	BCh

Function

See the description for ECCPOISONADDR0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cc4625-4821-48a2-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-30 —	Reserved
29-28 ecc_poison_bg	ECC poisoning bank group address Specifies the bank group address for ECC poisoning. Programming Mode: Static
27 —	Reserved
26-24 ecc_poison_bank	ECC poisoning bank address Specifies the bank address for ECC poisoning. Programming Mode: Static
23-18 —	Reserved
17-0 ecc_poison_row	ECC poisoning row address Specifies the row addres for ECC poisoning. Programming Mode: Static

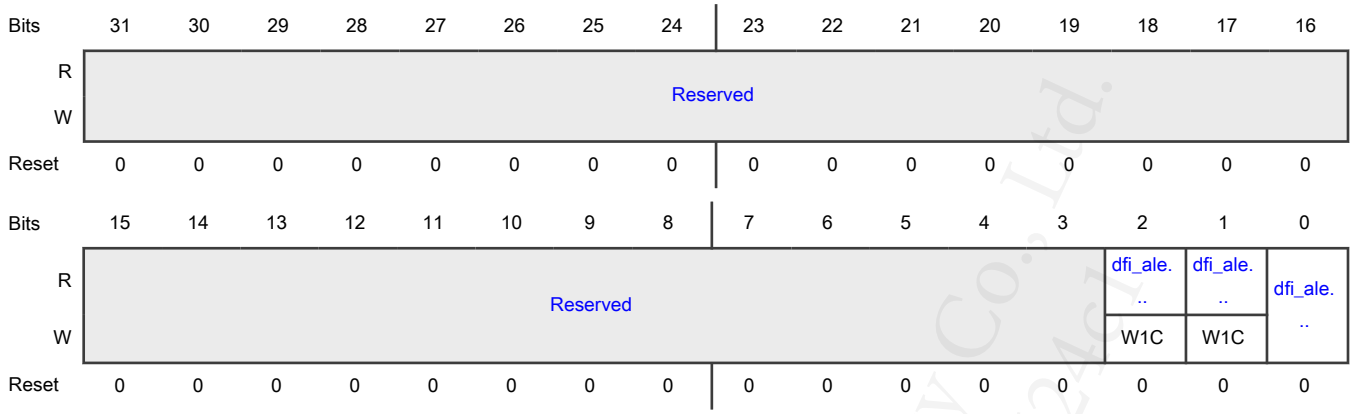
36.14.39 CRC Parity Control 0 (CRCPARCTL0)

Offset

Register	Offset
CRCPARCTL0	C0h

Preliminary Information
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46255-4321-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-3 —	Reserved
2 dfi_alert_err_cnt_clr	DFI alert error count clear. Clear bit for DFI alert error counter. Asserting this bit will clear the DFI alert error counter, CRCPARSTAT.dfi_alert_err_cnt. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
1 dfi_alert_err_int_clr	Interrupt clear bit for DFI alert error. If this bit is set, the alert error interrupt on CRCPARSTAT.dfi_alert_err_int will be cleared. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
0 dfi_alert_err_int_en	Interrupt enable bit for DFI alert error. If this bit is set, any parity/CRC error detected on the dfi_alert_n input will result in an interrupt being set on CRCPARSTAT.dfi_alert_err_int. Programming Mode: Dynamic

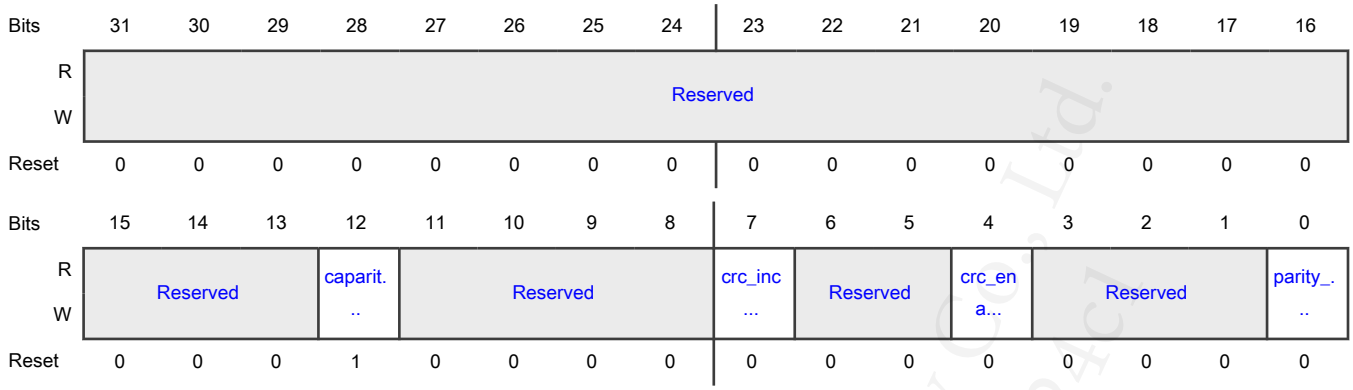
36.14.40 CRC Parity Control 1 (CRCPARCTL1)

Offset

Register	Offset
CRCPARCTL1	C4h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-482a-b2f1-90d3ab624c1

Diagram



Fields

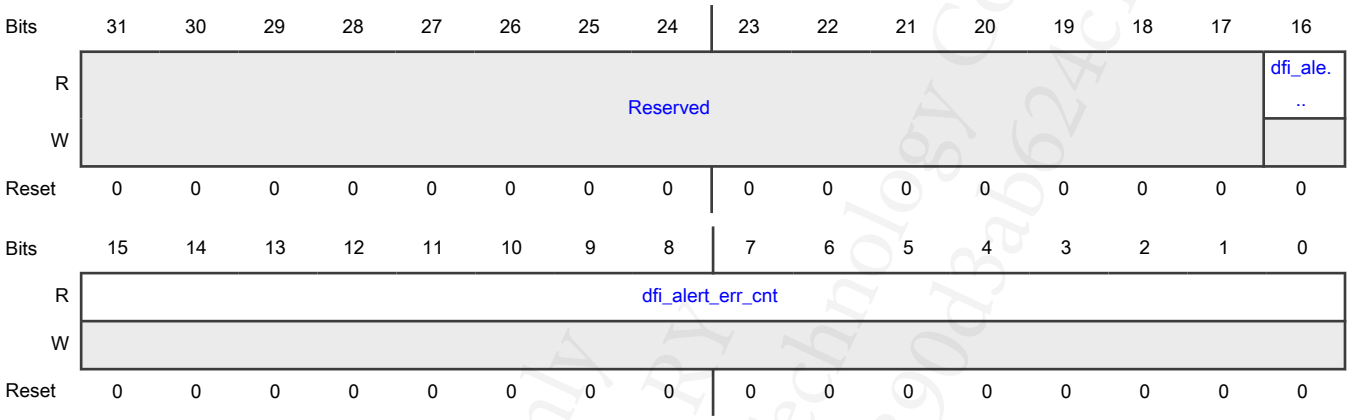
Field	Function
31-13 —	Reserved
12 caparity_disable_before_sr	If DDR4-DRAM's CA parity is enabled by INIT6.mr5[2:0]!=0 and this register is set to 1, CA parity is automatically disabled before Self-Refresh entry and enabled after Self-Refresh exit by issuing MR5. - 1: CA parity is disabled before Self-Refresh entry - 0: CA parity is not disabled before Self-Refresh entry If Geardown is used by MSTR.geardown_mode=1, this register must be set to 1. If this register set to 0, DRAMTMG5.t_ckesr and DRAMTMG5.t_cksre must be increased by PL(Parity latency) Programming Mode: Static
11-8 —	Reserved
7 crc_inc_dm	CRC Calculation setting register - 1: CRC includes DM signal - 0: CRC not includes DM signal Present only in designs configured to support DDR4. Programming Mode: Static
6-5 —	Reserved
4 crc_enable	CRC enable - 1: Enable generation of CRC - 0: Disable generation of CRC The setting of this register should match the CRC mode register setting in the DRAM. Programming Mode: Quasi-dynamic Group 2
3-1 —	Reserved
0 parity_enable	C/A Parity enable register - 1: Enable generation of C/A parity and detection of C/A parity error - 0: Disable generation of C/A parity and disable detection of C/A parity error If RCD's parity error detection or DRAM's parity detection is enabled, this register should be 1. Programming Mode: Static

36.14.41 CRC Parity Status (CRCPARSTAT)

Offset

Register	Offset
CRCPARSTAT	CCh

Diagram



Fields

Field	Function
31-17 —	Reserved
16 dfi_alert_err_int	DFI alert error interrupt. If a parity/CRC error is detected on dfi_alert_n, and the interrupt is enabled by CRCPARCTL0.dfi_alert_err_int_en, this interrupt bit will be set. It will remain set until cleared by CRCPARCTL0.dfi_alert_err_int_clr Programming Mode: Static
15-0 dfi_alert_err_cnt	DFI alert error count. If a parity/CRC error is detected on dfi_alert_n, this counter be incremented. This is independent of the setting of CRCPARCTL0.dfi_alert_err_int_en. It will saturate at 0xFFFF, and can be cleared by asserting CRCPARCTL0.dfi_alert_err_cnt_clr. Programming Mode: Static

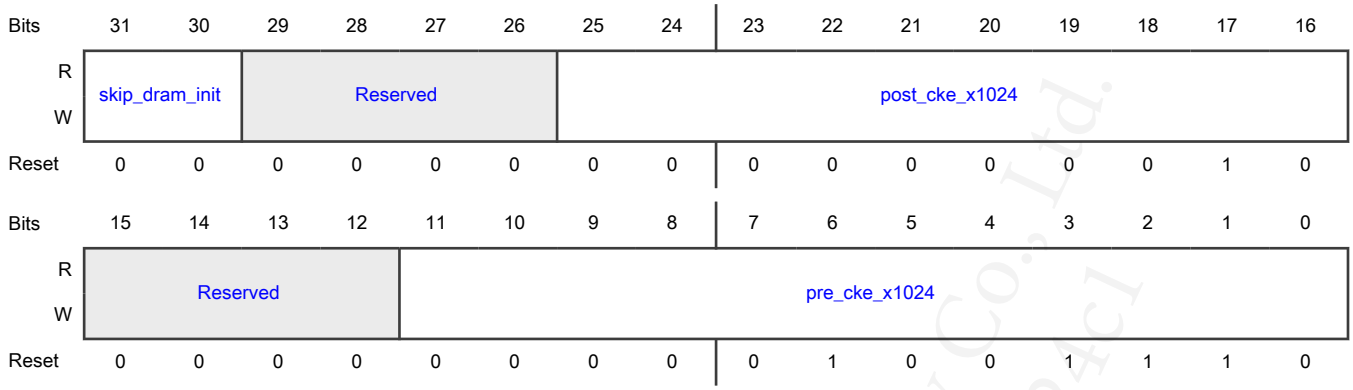
36.14.42 DRAM Initialization 0 (INIT0)

Offset

Register	Offset
INIT0	D0h

Provided by IHS NDA only
COMPANY PROPRIETARY
Beijing CTJ Automotive Technology Co., Ltd.
c8c462e5-4821-4ce8-b251-390d8ab624c1

Diagram



Fields

Field	Function
31-30 skip_dram_init	Skip DRAM Initialization/Reset State Specifies whether to skip the DRAM initialization routine and what state the MC controller should be in on reset exit. Programming Mode: Quasi-dynamic Group 2 00b - Run DRAM initialization routine on power-up. 01b - Skip DRAM initialization routine on power-up and start up in normal mode. 10b - Reserved 11b - Skip DRAM initialization routine on power-up and start up in Self Refresh state.
29-26 —	Reserved
25-16 post_cke_x1024	Post-CKE Interval Specifies the time interval, in multiples of 1024 DFI clock cycles, to wait after driving CKE high to start the DRAM initialization sequence. When the MC is operating at 1:2 frequency ratio, set post_cke_x1024 to RoundUp(JEDEC specification/2). **NOTE** • For DDR2: Typically requires a post_CKE delay of 400 ns, requiring post_cke_x1024 = 2 at all clock speeds. • For LPDDR2 and LPDDR3: Typically requires a post-CKE delay of 200 µs. • For LPDDR4: Typically requires a post_CKE delay of 2 µs. Programming Mode: Static
15-12 —	Reserved
11-0	Pre-CKE Interval

Table continues on the next page...

Table continued from the previous page...

Field	Function
pre_cke_x1024	Specifies the time interval, in multiples of 1024 DFI clock cycles, to wait after reset before driving CKE high to start the DRAM initialization sequence. When the MC is operating at 1:2 frequency ratio, set pre_cke_x1024 to RoundUp(JEDEC specification/2). **NOTE** • For DDR2: Typically requires a pre-CKE delay greater than 200 µs. • For LPDDR2 and LPDDR3: tINIT1 of 100 ns (min). • For LPDDR4: tINIT3 of 2 ms (min). • For DDR3 and DDR4 RDIMMS: Include the time needed to satisfy tSTAB. Programming Mode: Static

36.14.43 DRAM Initialization 1 (INIT1)

Offset

Register	Offset
INIT1	D4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								dram_rstn_x1024							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												pre_ocd_x32			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 —	Reserved
24-16	Reset Signal Assert Interval

Table continues on the next page...

Table continued from the previous page...

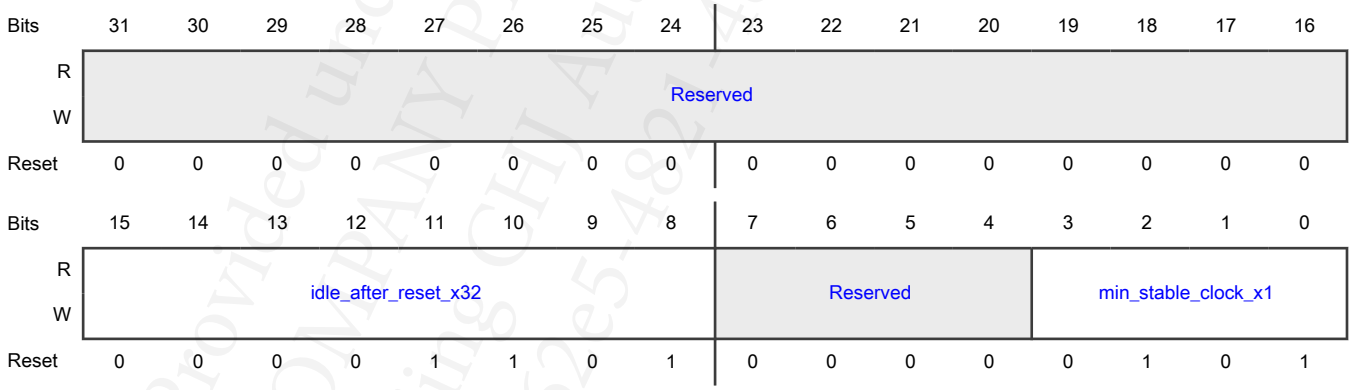
Field	Function
dram_rstn_x1024	Specifies the time interval, in multiples of 1024 DFI clock cycles, to assert the DRAM reset signal during the initialization sequence. **NOTE** • This field applies only to DDR3, DDR4 or LPDDR4 systems. When the MC is operating at 1:2 frequency ratio, set dram_rstn_x1024 to RoundUp(JEDEC specification/2). Programming Mode: Static 000000000b - Invalid value.
15-4 —	Reserved
3-0 pre_ocd_x32	Pre-OCD Interval Specifies the time interval, in multiples of 32 DFI clock cycles, before driving the OCD complete command to DRAM. There is no known specific requirement for this; you may set it to zero. Programming Mode: Static

36.14.44 DRAM Initialization 2 (INIT2)

Offset

Register	Offset
INIT2	D8h

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-3903ab624c1

Fields

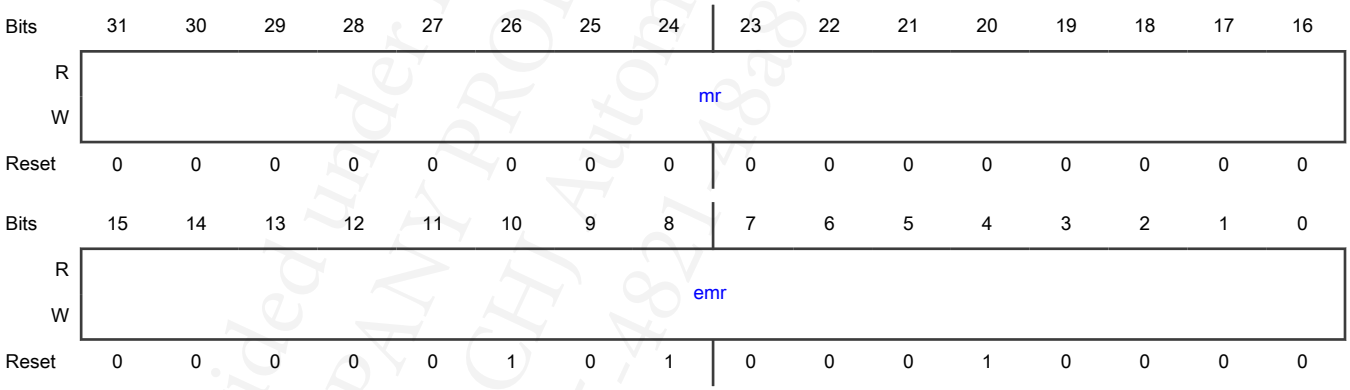
Field	Function
31-16 —	Reserved
15-8 idle_after_reset_x32	Idle time after the reset command, tINIT4. Present only in designs configured to support LPDDR2. When the controller is operating in 1:2 frequency ratio mode, program this to JEDEC spec value divided by 2, and round it up to the next integer value. Unit: Multiples of 32 DFI clock cycles. Programming Mode: Static
7-4 —	Reserved
3-0 min_stable_clock_x1	Time to wait after the first CKE high, tINIT2. Present only in designs configured to support LPDDR2/LPDDR3. LPDDR2/LPDDR3 typically requires 5 x tCK delay. When the controller is operating in 1:2 frequency ratio mode, program this to JEDEC spec value divided by 2, and round it up to the next integer value. Unit: DFI clock cycles. Programming Mode: Static

36.14.45 DRAM Initialization 3 (INIT3)

Offset

Register	Offset
INIT3	DCh

Diagram



Fields

Field	Function
31-16 mr	Mode Register Value Specifies the value to write to the relevant Mode Register, depending on the DRAM type:

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
5c462e5-4821-48a3-b251-390d3...624c2

Table continued from the previous page...

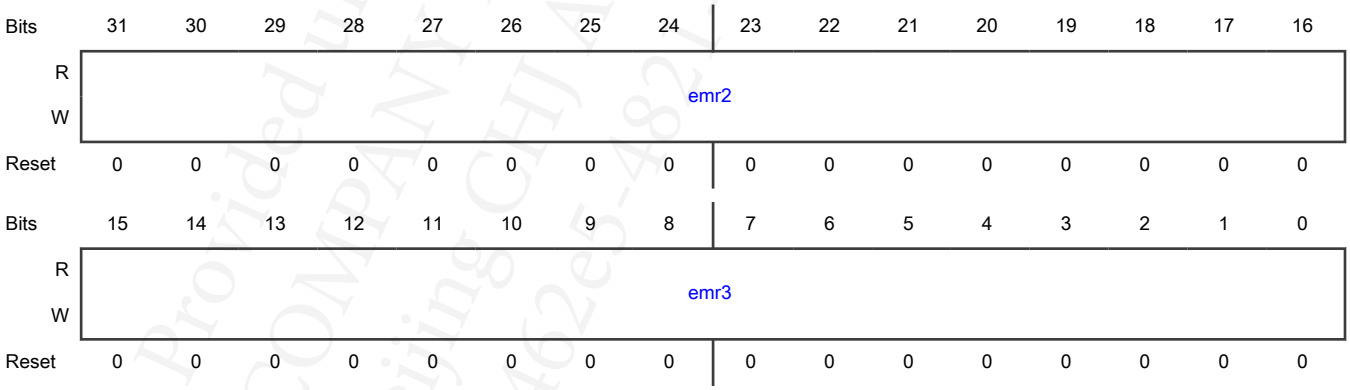
Field	Function
	• For DDR2: Mode Register (MR). Bit 8 is for DLL and its value in this register is ignored; the MC sets it appropriately. • For DDR3 and DDR4: Mode Register 0 (MR0). • For mDDR: MR • For LPDDR2, LPDDR3, and LPDDR4: MR1 Programming Mode: Quasi-dynamic Group 4
15-0 emr	Extended Mode Register Value Specifies the value to write to the relevant Mode Register, depending on the DRAM type: • For DDR2: Extended Mode Register (EMR). Bits 9–7 are for OCD and their value in this register is ignored; the MC sets them appropriately. • For DDR3 or DDR4: Mode Register 1 (MR1). Set bit 7 to 0. If PHY-evaluation mode training is enabled, the MC sets it appropriately during write leveling. • For mDDR: EMR • For LPDDR2, LPDDR3, and LPDDR4: MR2 Programming Mode: Quasi-dynamic Group 4

36.14.46 DRAM Initialization 4 (INIT4)

Offset

Register	Offset
INIT4	E0h

Diagram



Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8ad24c1

Fields

Field	Function
31-16 emr2	Extended Mode Register 2 Value Specifies the value to write to the relevant Mode Register, depending on the DRAM type: • For DDR2: Extended Mode Register 2 (EMR2). • For DDR3 and DDR4: Mode Register 2 (MR2). • For LPDDR2, LPDDR3, and LPDDR4: MR3 • For mDDR: Not used Programming Mode: Quasi-dynamic Group 4
15-0 emr3	Extended Mode Register 3 Value Specifies the value to write to the relevant Mode Register, depending on the DRAM type: • For DDR2: Extended Mode Register 3 (EMR3). • For DDR3 and DDR4: Mode Register 3 (MR3). • For mDDR, LPDDR2, and LPDDR3: Not used • For LPDDR4: MR13 Programming Mode: Quasi-dynamic Group 4

36.14.47 DRAM Initialization 5 (INIT5)

Offset

Register	Offset
INIT5	E4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								dev_zqinit_x32							
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved						max_auto_init_x1024									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Fields

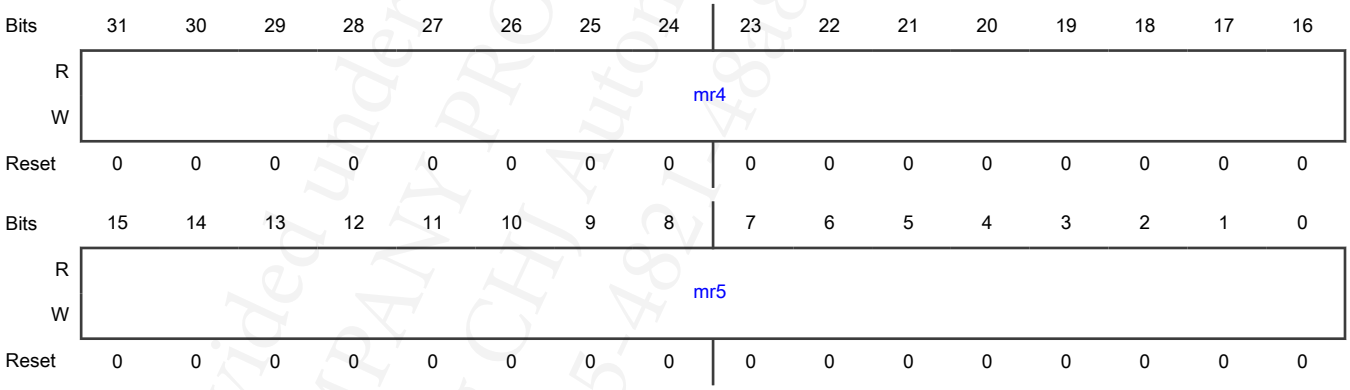
Field	Function
31-24 —	Reserved
23-16 dev_zqinit_x32	ZQ initial calibration, tZQINIT. Present only in designs configured to support DDR3 or DDR4 or LPDDR2/LPDDR3. DDR3 typically requires 512 DRAM clock cycles. DDR4 requires 1024 DRAM clock cycles. LPDDR2/LPDDR3 requires 1 us. When the controller is operating in 1:2 frequency ratio mode, program this to JEDEC spec value divided by 2, and round it up to the next integer value. Unit: Multiples of 32 DFI clock cycles. Programming Mode: Static
15-10 —	Reserved
9-0 max_auto_init_x1024	Maximum duration of the auto initialization, tINIT5. Present only in designs configured to support LPDDR2/LPDDR3. LPDDR2/LPDDR3 typically requires 10 us. Unit: Multiples of 1024 DFI clock cycles. Programming Mode: Static

36.14.48 DRAM Initialization 6 (INIT6)

Offset

Register	Offset
INIT6	E8h

Diagram



Fields

Field	Function
31-16 mr4	Mode Register 4 Value Specifies the value to write to the relevant Mode Register, depending on the DRAM type:

Table continues on the next page...

Table continued from the previous page...

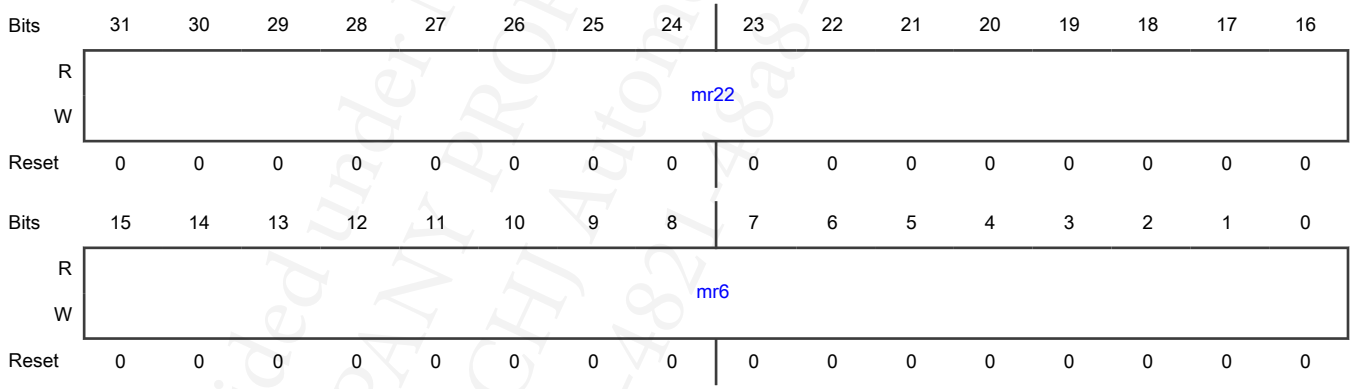
Field	Function
	• For DDR4: Mode Register 4 (MR4) • For LPDDR4: MR11 • For DDR2, DDR3, mDDR, LPPDDR2, and LPDDR3: Not used Programming Mode: Quasi-dynamic Group 4
15-0 mr5	Mode Register 5 Value Specifies the value to write to the relevant Mode Register, depending on the DRAM type: • For DDR4: Mode Register 5 (MR5) • For LPDDR4: MR12 • For DDR2, DDR3, mDDR, LPPDDR2, and LPDDR3: Not used Programming Mode: Quasi-dynamic Group 4

36.14.49 DRAM Initialization 7 (INIT7)

Offset

Register	Offset
INIT7	ECh

Diagram



Fields

Field	Function
31-16 mr22	Mode Register 22 Value Specifies the value to write to the relevant Mode Register, depending on the DRAM type:

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4821-8a8-b251-390a3ab624c1

Table continued from the previous page...

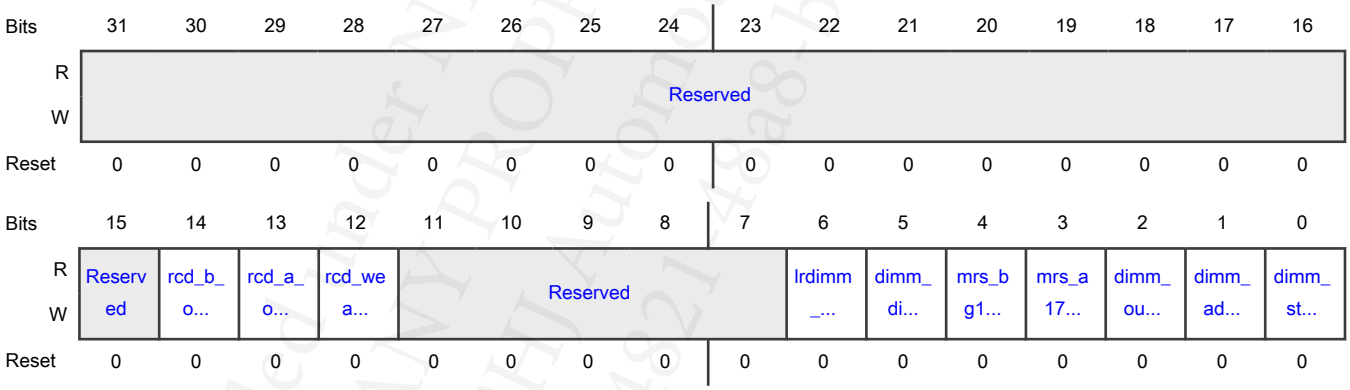
Field	Function
	• For LPDDR4: Mode Register 22 (MR22) • For DDR2, DDR3, DDR4, mDDR, LPPDDR2, and LPDDR3: Not used Programming Mode: Quasi-dynamic Group 4
15-0 mr6	Mode Register 6 Value Specifies the value to write to the relevant Mode Register, depending on the DRAM type: • For DDR4: Mode Register 6 (MR6) • For LPDDR4: MR14 • For DDR2, DDR3, mDDR, LPPDDR2, and LPDDR3: Not used Programming Mode: Quasi-dynamic Group 4

36.14.50 DIMM Control (DIMMCTL)

Offset

Register	Offset
DIMMCTL	F0h

Diagram



Fields

Field	Function
31-15 —	Reserved
14	Disable RCD outputs to B-side DRAMs. This field is used only when the uMCTL2 disables CAL mode. This value is written to F0RC0 DA[3] before and after disabling CAL mode. It is recommended to set

Table continues on the next page...

Table continued from the previous page...

Field	Function
rcd_b_output_disabled	it to ~DIMMCTL.dimm_output_inv_en except for debug. - 1 - Disable B outputs - 0 - Enable B outputs Programming Mode: Static
13 rcd_a_output_disabled	Disable RCD outputs to A-side DRAMs. This field is used only when the uMCTL2 disables CAL mode. This value is written to F0RC0 DA[2] before and after disabling CAL mode. It is recommended to set it to 0 except for debug. - 1 - Disable A outputs - 0 - Enable A outputs Programming Mode: Static
12 rcd_weak_drive	Weak Drive mode to set to the RCD. This field is used only when the uMCTL2 disables CAL mode. When weak drive mode in the RCD is enabled during initialization, this field should be set to 1. When RCD is not used, this field should be set to 0. - 1 - Enable Weak Drive mode - 0 - Disable Weak Drive mode Programming Mode: Static
11-7 —	Reserved
6 lrdimm_bcom_cmd_prot	Protects the timing restrictions (tBCW/tMRC) between consecutive BCOM commands defined in the Data Buffer specification. When using DDR4 LRDIMM, this bit must be set to 1. Otherwise, this bit must be set to 0. Programming Mode: Static
5 dimm_dis_bg_mirroring	Disabling Address Mirroring for BG bits. When this is set to 1, BG0 and BG1 are NOT swapped even if Address Mirroring is enabled. This will be required for DDR4 DIMMs with x16 devices. - 1 - BG0 and BG1 are NOT swapped. - 0 - BG0 and BG1 are swapped if address mirroring is enabled. Programming Mode: Static
4 mrs_bg1_en	Enable for BG1 bit of MRS command. BG1 bit of the mode register address is specified as RFU (Reserved for Future Use) and must be programmed to 0 during MRS. In case where DRAMs which do not have BG1 are attached and both the CA parity and the Output Inversion are enabled, this must be set to 0, so that the calculation of CA parity will not include BG1 bit. Note: This has no effect on the address of any other memory accesses, or of software-driven mode register accesses. If address mirroring is enabled, this is applied to BG1 of even ranks and BG0 of odd ranks. - 1 - Enabled - 0 - Disabled Programming Mode: Static
3 mrs_a17_en	Enable for A17 bit of MRS command. A17 bit of the mode register address is specified as RFU (Reserved for Future Use) and must be programmed to 0 during MRS. In case where DRAMs which do not have A17 are attached as DDR4 RDIMM/LRDIMM and the Output Inversion is enabled, this must be set to 0, so that the calculation of CA parity will not include A17 bit. To keep a consistency with the RCD, DA[3] in F0RC08 of the RCD also needs to be set to 1 (i.e. Disabled) Note: This has no effect on the address of any other memory accesses, or of software-driven mode register accesses. - 1 - Enabled - 0 - Disabled Programming Mode: Static
2 dimm_output_inv_en	Output Inversion Enable (for DDR4 RDIMM/LRDIMM implementations only). DDR4 RDIMM/LRDIMM implements the Output Inversion feature by default, which means that the following address, bank address and bank group bits of B-side DRAMs are inverted: A3-A9, A11, A13, A17, BA0-BA1, BG0-BG1. Setting this bit ensures that, for mode register accesses generated by the uMCTL2 during the automatic initialization routine and enabling of a particular DDR4 feature, separate A-side and B-side mode register accesses are generated. For B-side mode register accesses, these bits are inverted within the uMCTL2 to compensate for this RDIMM/LRDIMM inversion. It is recommended to set this bit always, if using DDR4 RDIMMs/LRDIMMs. Note: This has no effect on the address of any other memory accesses, or of software-driven mode register accesses. - 1 - Implement output inversion for B-side DRAMs. - 0 - Do not implement output inversion for B-side DRAMs. Programming Mode: Static

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 dimm_addr_mirr_en	Address Mirroring Enable (for multi-rank UDIMM implementations and multi-rank DDR4 RDIMM/LRDIMM implementations). Some UDIMMs and DDR4 RDIMMs/LRDIMMs implement address mirroring for odd ranks, which means that the following address, bank address and bank group bits are swapped: (A3, A4), (A5, A6), (A7, A8), (BA0, BA1) and also (A11, A13), (BG0, BG1) for the DDR4. Setting this bit ensures that, for mode register accesses during the automatic initialization routine, these bits are swapped within the uMCTL2 to compensate for this UDIMM/RDIMM/LRDIMM swapping. In addition to the automatic initialization routine, in case of DDR4 UDIMM/RDIMM/LRDIMM, they are swapped during the automatic MRS access to enable/disable of a particular DDR4 feature. Note: This has no effect on the address of any other memory accesses, or of software-driven mode register accesses. This is not supported for mDDR, LPDDR2, LPDDR3 or LPDDR4 SDRAMs. Note: In case of x16 DDR4 DIMMs, BG1 output of MRS for the odd ranks is same as BG0 because BG1 is invalid, hence dimm_dis_bg_mirroring register must be set to 1. - 1 - For odd ranks, implement address mirroring for MRS commands to during initialization and for any automatic DDR4 MRS commands (to be used if UDIMM/RDIMM/LRDIMM implements address mirroring) - 0 - Do not implement address mirroring Programming Mode: Static
0 dimm_stagger_cs_en	Staggering enable for multi-rank accesses (for multi-rank UDIMM, RDIMM and LRDIMM implementations only). This is not supported for mDDR, LPDDR2, LPDDR3 or LPDDR4 SDRAMs. Note: Even if this bit is set it does not take care of software driven MR commands (via MRCTRL0/MRCTRL1), where software is responsible to send them to seperate ranks as appropriate. - 1 - (DDR4) Send MRS commands to each ranks seperately - 1 - (non-DDR4) Send all commands to even and odd ranks seperately - 0 - Do not stagger accesses Programming Mode: Static

36.14.51 Rank Control (RANKCTL)

Offset

Register	Offset
RANKCTL	F4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				diff_rank_wr_gap				diff_rank_rd_gap				max_rank_rd			
Reset	0	0	0	0	0	1	1	0	0	1	1	0	1	1	1	1

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive
c8c46e25-4821-48a8-b251-

Fields

Field	Function
31-12 —	Reserved
11-8 diff_rank_wr_gap	Different rank write gap Specifies the time gap in DFL clock cycles between data responses when performing consecutive writes to differnt ranks. Use this field to switch PHY delays to match the rank requirements. You should consider both the PHY and ODT requirements when specifying this value: • PHY requirement: — tphy_wrcsgap (see the PHY databook for this value) — If CRC is enabled, increase the gap value by 1. — If write preamble is set to 2tCK (DDR4 only), increase the gap value by 1. — If write postamble is set to 1.5tCK (LPDDR4 only), increase the gap value by 1. • ODT requirement: — The value in this register takes care of the ODT switch-off timing requirement when switching ranks during writes. — For LPDDR4: With DQ ODT enabled, the gap value must be a minimum of: ODTLoff – ODTLon – BL/2 + 1 When the MC is operating at: • 1:1 frequency ratio, diff_rank_wr_gap should be the larger of the PHY requirement or the ODT requirement. • 1:2 frequency ratio, diff_rank_wr_gap should be the larger value divided by two and rounded up to the next integer. **NOTE** • This field applies only to multi-rank configurations. • For DDR4-LRDIMM, refer to the TWRRD timing requirements in the JEDEC DDR4 Data Buffer (DDR4DB01) specification. • For other DRAM: the gap value must be a minimum of ODTCFG[wr_odt_hold] – BL/2. Programming Mode: Quasi-dynamic Group 2
7-4 diff_rank_rd_gap	Different rank read gap Specifies the time gap in DFL clock cycles between data responses when performing consecutive reads to differnt ranks. Use this field to switch PHY delays to match the rank requirements. You should consider both the PHY and ODT requirements when specifying this value: • PHY requirement: — tphy_rdcsgap (see the PHY databook for this value) — If read preamble is set to 2tCK (DDR4 only), increase the gap value by 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	— If read postamble is set to 1.5tCK (LPDDR4 only), increase the gap value by 1. • ODT requirement: — The value in this register takes care of the ODT switch-off timing requirement when switching ranks during reads: diff_rank_rd_gap must be a minimum of ODTCFG[rd_odt_hold] – BL/2. When the MC is operating at: • 1:1 frequency ratio, diff_rank_rd_gap should be the larger of the PHY requirement or the ODT requirement • 1:2 frequency ratio, diff_rank_rd_gap should be the larger value divided by two and rounded up to the next integer **NOTE** • This field applies only to multi-rank configurations. • For DDR4-LRDIMM, refer to the TRDRD timing requirements in the JEDEC DDR4 Data Buffer (DDR4DB01) specification. Programming Mode: Quasi-dynamic Group 2
3-0 max_rank_rd	Rank Maximum Reads Specifies the maximum number of reads that can be scheduled consecutively to the same rank. Reads to the same rank can be performed back-to-back. Reads to different ranks require additional gap, which you specify in diff_rank_rd_gap. This is to avoid possible data bus contention as well as to give PHY enough time to switch the delay when changing ranks. The MC arbitrates for bus access on a cycle-by-cycle basis; therefore, after a read is scheduled, there are only a few clock cycles (determined by the value in diff_rank_rd_gap) in which only reads from the same rank are eligible to be scheduled. This prevents reads from other ranks from having fair access to the data bus. After the numbers of consecutive reads specified in this field, the scheduler inserts a delay equal to diff_rank_rd_gap to allow all ranks a fair opportunity to be scheduled. Higher numbers increase bandwidth utilization, lower numbers increase fairness. You can disable this feature by setting this field to 0. When set to 0, the MC stays on the same rank as long as commands are available for it. The minimum programmable value is 0 (feature disabled) and the maximum is Fh. FOR PERFORMANCE ONLY. **NOTE** • Applies to multi-rank configurations. • Affects performance only. Programming Mode: Static

36.14.52 DRAM Timing 0 (DRAMTMG0)

Offset

Register	Offset
DRAMTMG0	100h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	wr2pre							Reserved		t_faw					
Reset	0	0	0	0	1	1	1	1	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved	t_ras_max							Reserved		t_ras_min					
Reset	0	0	0	1	1	0	1	1	0	0	0	0	1	1	1	1

Fields

Field	Function
31 —	Reserved
30-24 wr2pre	Write to precharge minimum interval Specifies the minimum time interval, in DFI clock cycles, between write and precharge to the same bank. Use the following equation to determine the correct value: (WL + BL/2 + tWR) + 15 ns where: • WL is write latency. • BL is burst length. This must match the value in the BL field of the mode register to the DRAM. The MC does not support BST (burst terminate) at this time. • tWR is write recovery time. This comes directly from the DRAM specification. The parenthetical portion of the equation equals approximately eight cycles. Therefore, at 400 MHz, wr2pre should be 14 clock cylces. The value is less at lower frequencies. For LPDDR2, LPDDR3, and LPDDR4, add one extra cycle. When the MC is operating at: • 1:2 frequency ratio and 1T command timing, divide the above value by 2 (no rounding up). • 1:2 frequency ratio and 2T command timing or LPDDR4, divide the above value by 2 and round up. **NOTE** Depending on the PHY, when using LRDIMM you may need to adjust wr2pre to compensate for the extra cycle of latency through the LRDIMM. Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4
23-22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

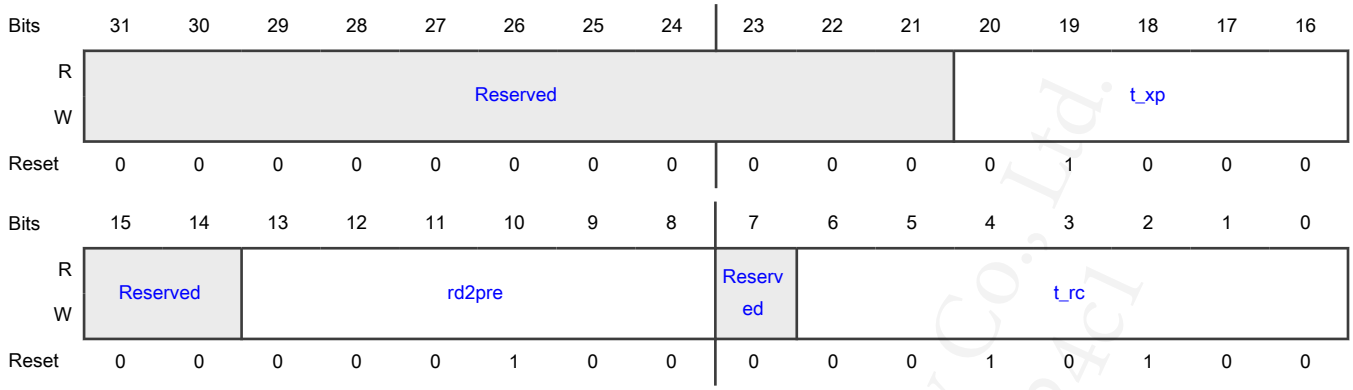
Field	Function
21-16 t_faw	Four active window (tFAW) Specifies the sliding time window, in DFI clock cycles, in which a maximum of 4-bank activates are allowed in an 8-bank configuraiton. When the MC is operating in 1:2 frequency ratio, t_faw should be RoundUp(tFAW/2). In a 4-bank design, set t_faw to 1h independent of the frequency ratio. **NOTE** This field applies only when eight or more banks (or banks x bank groups) are present. Programming Mode: Quasi-dynamic Group 2, Group 4
15 —	Reserved
14-8 t_ras_max	Activate to precharge maximum interval Specifies the maximum time interval, in multiples of 1024 DFI clock cycles, between activate and precharge to the same bank: tRAS(max). This is the maximum time that a page can be kept open. When the MC is operating at: • 1:1 frequency ratio, t_ras_max should be RoundDown(tRAS(max)/tCK/1024). • 1:2 frequency ratio, t_ras_max should be RoundDown((RoundDown(tRAS(max)/tCK/1024) – 1)/2) Programming Mode: Quasi-dynamic Group 2, Group 4 0000000b - Invalid value
7-6 —	Reserved
5-0 t_ras_min	Activate to precharge minimum interval Specifies the minimum time interval, in DFI clock cycles, between activate and precharge to the same bank: tRAS(min). When the MC is operating at: • 1:2 frequency ratio and 1T command timing, t_ras_min should be tRAS(min)/2 (no rounding up) • 1:2 frequency ratio and 2T command timing or LPDDR4 mode, t_ras_min should be RoundUp(tRAS(min)/2) Programming Mode: Quasi-dynamic Group 2, Group 4

36.14.53 DRAM Timing 1 (DRAMTMG1)

Offset

Register	Offset
DRAMTMG1	104h

Diagram



Fields

Field	Function
31-21 —	Reserved
20-16 t_xp	tXP Specifies the minimum time interval, in DFI clock cycles, before any operation after Power Down Exit (PDX). Use the following guidelines to determine the correct value: • DDR3: If using slow power down exit (MR0[12]), set to tXPDLL • DDR4: If using C/A parity, set to (tXP + PL) • LPDDR4: If spec has the tCKELPD parameter, set to max(tXP, tCKELPD) When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4
15-14 —	Reserved
13-8 rd2pre	Read to precharge minimum interval (tRTP) Specifies the minimum time interval, in DFI clock cycles, between read and precharge to the same bank. Use the following equations to determine the correct value: • DDR2: tAL + BL/2 + max(tRTP, 2) – 2 • DDR3: tAL + max(tRTP,4) • DDR4: Maximum of the following equations: — tAL + max(tRTP,4) — RL + BL/2 – tRP (When DDR4 DRAM and ST-MRAM are used simultaneously, use the DRAM tRP value for calculation.)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	— mDDR: BL/2 — LPDDR2-S2: BL/2 + tRTP – 1 — LPDDR2-S4: BL/2 + max(tRTP,2) – 2 — LPDDR3: BL/2 + max(tRTP,4) – 4 — LPDDR4: BL/2 + max(tRTP,8) – 8 When the MC is operating at: • 1:2 frequency ratio and 1T command timing, divide the above value by 2 (no rounding up). • 1:2 frequency ratio and 2T command timing or LPDDR4, divide the above value by 2 and round up. Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4
7 —	Reserved
6-0 t_rc	Same bank activate internal (tRC) Specifies the minimum time interval, in DFI clock cycles, between activates to the same bank. When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4

36.14.54 DRAM Timing 2 (DRAMTMG2)

Offset

Register	Offset
DRAMTMG2	108h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved		write_latency						Reserved		read_latency					
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		rd2wr						Reserved		wr2rd					
Reset	0	0	0	0	0	1	1	0	0	0	0	0	1	1	0	1

Provided under NDA only
COMPANY PROPRIETARY
Beijing eMU Automotive Technology Co., Ltd.
c8c462e5-4214-48a8-b251-390d3ab64c1

Fields

Field	Function
31-30 —	Reserved
29-24 write_latency	Write latency Specifies the time interval, in DFI clock cycles, from a write command to write data on the DRAM interface. You must set this to WL (write latency). When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. **NOTE** • This field does not apply to DDR2 and DDR3 (unless MEMC_TRAINING is set), because the DFI read and write latencies defined in DFITMG0 and DFITMG1 are sufficient for those protocols. • For all protocols, you must also program DFITMG0 and DFITMG1 to control the read and write latencies. Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4
23-22 —	Reserved
21-16 read_latency	Read Latency Specifies the time interval, in DFI clock cycles, from a read command to read data on the DRAM interface. You must set this to RL (read latency). When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. **NOTE** • This field does not apply to DDR2 and DDR3 (unless MEMC_TRAINING is set), because the DFI read and write latencies defined in DFITMG0 and DFITMG1 are sufficient for those protocols. • For all protocols, you must also program DFITMG0 and DFITMG1 to control the read and write latencies. Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4
15-14 —	Reserved
13-8 rd2wr	Read To Write Minimum Time Specifies the minimum time interval, in DFI clock cycles, from a read command to a write command. For DDR4, it's the minimum interval from a read command to a write command for the same bank. This interval includes the time for bus turnaround, and all per-bank, per-rank, and global constraints. Use the following equations to determine the correct value:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• DDR2, DDR3, mDDR: RL + BL/2 + 2 - WL • DDR4: RL + BL/2 + 1 + WR_PREAMBLE - WL • LPDDR2, LPDDR3: RL + BL/2 + RU(tDQSCKmax/tCK) + 1 - WL • LPDDR4 (DQ ODT Disabled): RL + BL/2 + RU(tDQSCKmax/tCK) + WR_PREAMBLE + RD_POSTAMBLE - WL • LPDDR4 (DQ ODT Enabled): RL + BL/2 + RU(tDQSCKmax/tCK) + RD_POSTAMBLE - ODTLon - RU(tODTon(min)/tCK) + 1 Where: • WL is write latency • BL is burst length. Must match the value programmed in the BL bit of the DRAM Mode Register. • RL is read latency (CAS latency) • WR_PREAMBLE is 1 for 1tCK write preamble or 2 for 2tCK write preamble. Unique to DDR4 and LPDDR4. • RD_POSTAMBLE is 0.5 for 0.5tCK read postamble or 1.5 for 1.5tCK read postamble. This is unique to LPDDR4. For LPDDR2, LPDDR3, or LPDDR4, if derating is enabled (DERATEEN[derate_enable] is equal to 1), you should use derated tDQSCKmax. When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4
7-6 — 5-0 wr2rd	Reserved Write To Read Minimum Time Specifies the minimum time interval, in DFI clock cycles, from a write command to a read command. For DDR4, it's the minimum interval from a write command to a read command for the same bank. This interval includes the time for bus turnaround, recovery times, and all per-bank, per-rank, and global constraints. Use the following equations to determine the correct value: • DDR4: CWL + PL + BL/2 + tWTR_L • LPDDR2, LPDDR3, LPDDR4: WL + BL/2 + tWTR + 1 • Others: CWL + BL/2 + tWTR Where: • CWL is CAS write latency • WL is write latency • PL is parity latency

Table continues on the next page...

Table continued from the previous page...

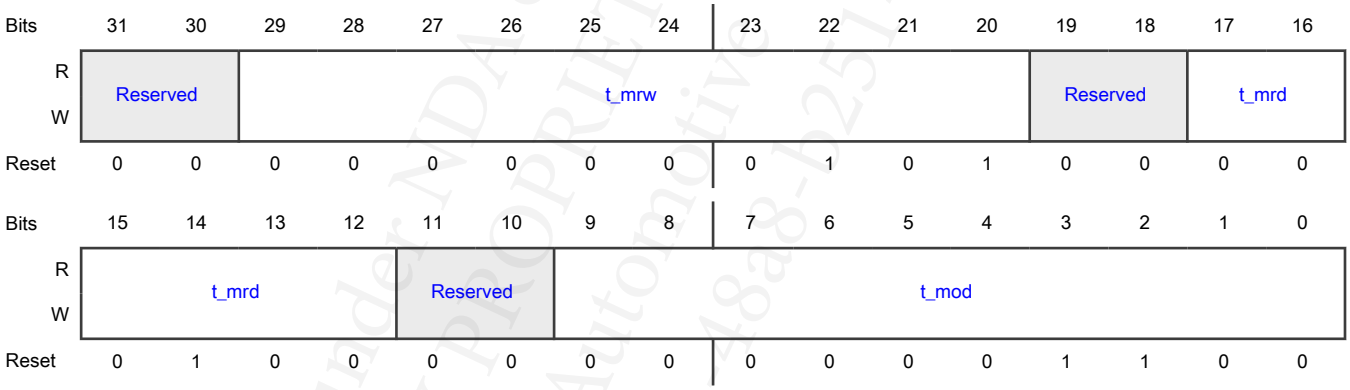
Field	Function
	• BL is burst length. Must match the value programmed in the BL bit of the DRAM Mode Register. • tWTR_L is internal write to read command delay for same bank group (directly from the DRAM specification). For DDR4, if using 2tCK write preamble, increase by one. • tWTR is internal write to read command delay (directly from the DRAM specification). When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4

36.14.55 DRAM Timing 3 (DRAMTMG3)

Offset

Register	Offset
DRAMTMG3	10Ch

Diagram



Fields

Field	Function
31-30 —	Reserved
29-20 t_mrw	tMRW Specifies the time interval, in DFI clock cycles, to wait after a Mode Register write (MRW) or Mode Register read (MRR) for LPDDR2, LPDDR3, or LPDDR4 DRAM. For LPDDR2 DRAM, specifies the time interval from MRW or MRR to all other commands. For LPDDR3 DRAM, specifies the time internal from MRW or MRR to another MRW or MRR.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Depending on the type of DRAM, follow these guidelines: • LPDDR2: typically requires a value of 5 • LPDDR3: typically requires a value of 10 • LPDDR4: Set to the larger of tMRW or TMRWCKEL When the MC is operating at 1:2 frequency ratio, divide value calculated above by 2 and round up to next integer. Programming Mode: Quasi-dynamic Group 2, Group 4
19-18 —	Reserved
17-12 t_mrd	tMRD Specifies the time interval, in DFI clock cycles, to wait after a MR write or read. Depending on the type of DRAM, tMRD is: • DDR2, mDDR: the time from MRS to any command • DDR3, DDR4: the time from MRS command to MRS command • LPDDR2: not used • LPDDR3, LPDDR4: the time from MRS command to non-MRS command Apply the following modifications to tMRD as necessary: • If CAL mode is enabled (DFITMG1[dfi_t_cmd_lat] > 0): add tCAL • For DDR4: If using C/A parity, use tMRD_PAR(tMOD + PL) instead When the MC is operating at 1:2 frequency ratio, use RoundUp(tMRD/2). Programming Mode: Quasi-dynamic Group 2, Group 4
11-10 —	Reserved
9-0 t_mod	tMOD Specifies the time interval, in DFI clock cycles, between a load mode (MR) command and a following non-load mode command. This field applies only to DDR3 and DDR4. Apply the following modifications to tMOD as necessary: • If CAL mode is enabled (DFITMG1[dfi_t_cmd_lat] > 0): add tCAL • For DDR4: — If using C/A parity, tMOD_PAR(tMOD + PL) — If using MPR writes, tMOD + AL

Table continues on the next page...

Table continued from the previous page...

Field	Function
	— If using MPR writes and C/A parity, tMOD_PAR + AL When the MC is operating at 1:1 frequency ratio, use the value calculated above. When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4

36.14.56 DRAM Timing 4 (DRAMTMG4)

Offset

Register	Offset
DRAMTMG4	110h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			t_rcd					Reserved				t_ccd			
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				t_rrd				Reserved				t_rp			
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-29 —	Reserved
28-24 t_rcd	tRCD – tAL Specifies the minimum time interval, in DFI clock cycles, from activate to read or write command to the same bank. When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE The minimum allowed value is 1. Therefore, when the MC is operating at 1:2 frequency ratio, (tRCD – tAL) must be a minimum of 2. Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4
23-20 —	Reserved
19-16 t_ccd	tCCD Specifies the minimum time interval, in DFI clock cycles, between two reads or two writes. For DDR4, use tCCD_L, the minimum time interval between two reads or two writes for the same bank group. When the MC is operating at 1:2 frequency ratio, divide the value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4
15-12 —	Reserved
11-8 t_rrd	tRRD Specifies the minimum time interval, in DFI clock cycles, between activates from bank a to bank b. For DDR4, use tRRD_L, the minimum time interval between activates from bank a to bank b for the same bank group. When the MC is operating at 1:2 frequency ratio, divide the value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4
7-5 —	Reserved
4-0 t_rp	tRP Specifies the minimum time interval, in DFI clock cycles, from precharge to active on the same bank. When the MC is operating at 1:1 frequency ratio, set to RoundUp(tRP/tCK). When the MC is operating at 1:2 frequency ratio, set to RoundDown(RoundUp(tRP/tCK)/2) + 1. For LPDDR4, when the MC is operating at 1:2 frequency ratio, set to RoundDown(RoundUp(tRP/tCK)/2). Programming Mode: Quasi-dynamic Group 2, Group 4

Printed under NDA only
COMPANY PROPRIETARY
Beijing C-I Automotive Technology Co., Ltd.
c8c46e25-4821-48d8-b251-390c3b624c1

36.14.57 DRAM Timing 5 (DRAMTMG5)

Offset

Register	Offset
DRAMTMG5	114h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				t_cksrx				Reserved				t_cksre			
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		t_ckesr						Reserved			t_cke				
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1

Fields

Field	Function
31-28 —	Reserved
27-24 t_cksrx	tCKSRX Specifies the time interval, in DFI clock cycles, to maintain the clock before the Self Refresh Exit (SRX) command. This is the time that CK is maintained as a valid clock before exiting the Self Refresh state. Depending on the type of DRAM, follow these guidelines: • mDDR: 1 • LPDDR2, LPDDR3: 2 • LPDDR4: tCKCKEH • DDR2: 1 • DDR3, DDR4: tCKSRX When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4
23-20 —	Reserved
19-16	tCKSRE

Table continues on the next page...

Table continued from the previous page...

Field	Function
t_cksre	Specifies the time interval, in DFI clock cycles, to delay clock disable after the Self Refresh Entry (SRE) command. This is the time that CK is maintained as a valid clock after entering the Self Refresh state. Depending on the type of DRAM, follow these guidelines: • mDDR: 0 • LPDDR2, LPDDR3: 2 • LPDDR4: tCKELCK • DDR2: 1 • DDR3, DDR4: max (10 ns, 5 tCK) • DDR4 if CRCPARCTL1[caparity_disable_before_sr] == 0: max (10 ns, 5 tCK) + PL (where PL is parity latency) When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4
15-14 —	Reserved
13-8 t_ckesr	tCKESR Specifies the minimum time interval, in DFI clock cycles, for CKE Low (minimum CKE low width) for self refresh or self refresh power down entry to exit. Depending on the type of DRAM, follow these guidelines: • mDDR: set to tRFC • LPDDR2, LPDDR3: set to tCKESR • LPDDR4: set to the larger of tCKE or tSR • DDR2: set to tCKE • DDR3, DDR4: set to tCKE + 1 • DDR4 if CRCPARCTL1[caparity_disable_before_sr] == 0: set to tCKE + 1 + PL (where PL is parity latency) When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4
7-5 —	Reserved
4-0 t_cke	tCKE Specifies the minimum time interval, in DFI clock cycles, for CKE High/Low during power down and self refresh.

Table continues on the next page...

Table continued from the previous page...

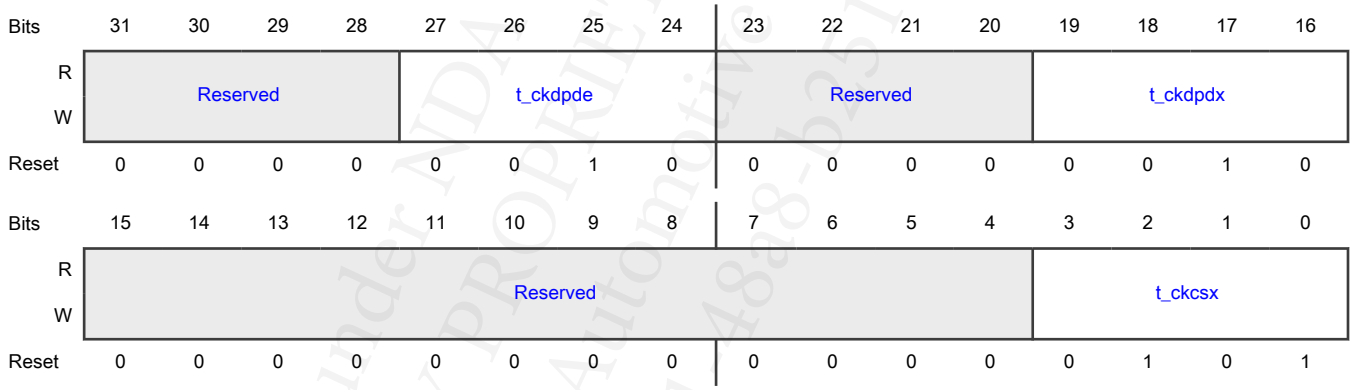
Field	Function
	Depending on the type of DRAM, follow these guidelines: • LPDDR2, LPDDR3: set to the larger of tCKE or tCKESR • LPDDR4: Set to the larger of tCKE or tSR • All other types: set to tCKE When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4

36.14.58 DRAM Timing 6 (DRAMTMG6)

Offset

Register	Offset
DRAMTMG6	118h

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 t_ckdpde	tCKDPDE Specifies the time interval, in DFI clock cycles, to delay clock disable after the Deep Power Down Entry (DPDE) command. This is the time that CK is maintained as a valid clock after entering the Deep Power Down state. Depending on the type of DRAM, follow these guidelines:

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8a62e54821f48a8-b251-390d3ab624d1

Table continued from the previous page...

Field	Function
	• mDDR: 0 • LPDDR2, LPDDR3: 2 • All other types: This field does not apply. When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4
23-20 —	Reserved
19-16 t_ckdpdx	Specifies the time interval, in DFI clock cycles, to maintain CK as a valid clock before the Deep Power Down Exit (DPDX) command. This is the clock stable time before exiting the Deep Power Down state. Depending on the type of DRAM, follow these guidelines: • mDDR: 1 • LPDDR2, LPDDR3: 2 • All other types: This field does not apply. When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4
15-4 —	Reserved
3-0 t_ckcsx	tCKCSX Specifies the time interval, in DFI clock cycles, that CK is maintained as a valid clock before the Clock Stop Exit (CSX) command. This is the clock stable time before the next command after CSX. Depending on the type of DRAM, follow these guidelines: • mDDR: 1 • LPDDR2, LPDDR3, LPDDR4: tXP + 2 • All other types: This field does not apply. When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4

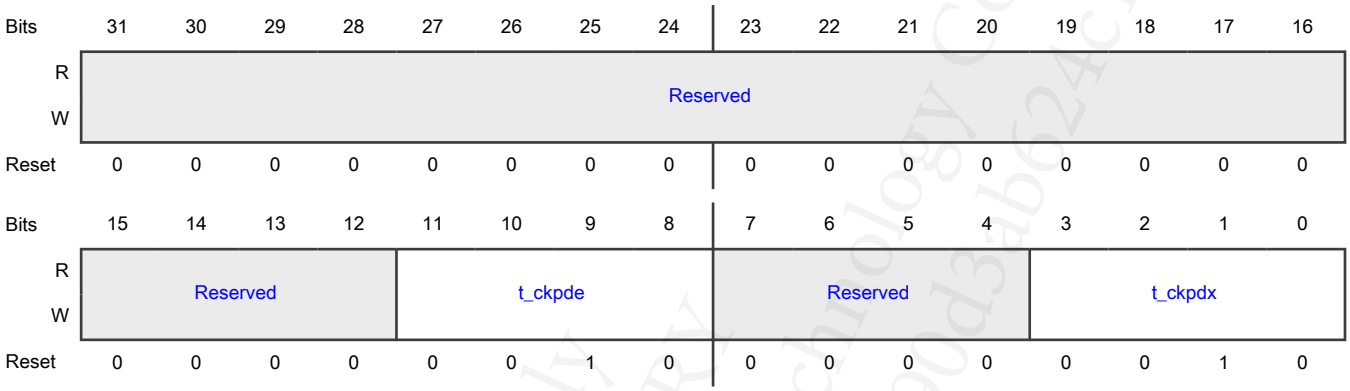
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-4821-4aa8-b251-390d3b624c1

36.14.59 DRAM Timing 7 (DRAMTMG7)

Offset

Register	Offset
DRAMTMG7	11Ch

Diagram



Fields

Field	Function
31-12 —	Reserved
11-8 t_ckpde	tCKPDEX Specifies the time interval, in DFI clock cycles, to maintain CK after the Power Down Entry (PDE) command. This is the time before clock disable after entering the Power Down state. Depending on the type of DRAM, follow these guidelines: • mDDR: 0 • LPDDR2, LPDDR3: 2 • LPDDR4: tCKELCK • DDR2, DDR3, DDR4: set to same value as DRAMTMG5[t_cksre] When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4
7-4 —	Reserved
3-0 t_ckpdx	tCKPDX

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Specifies the time interval, in DFI clock cycles, to maintain CK before the Power Down Exit (PDX) command. This is the time that CK is maintained as a valid clock before exiting the Power Down state. Depending on the type of DRAM, follow these guidelines: • mDDR: 0 • LPDDR2, LPDDR3: 2 • LPDDR4: tCKCKEH • DDR2, DDR3, DDR4: set to same value as DRAMTMG5[t_cksrx] When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up to the next integer. Programming Mode: Quasi-dynamic Group 2, Group 4

36.14.60 DRAM Timing 8 (DRAMTMG8)

Offset

Register	Offset
DRAMTMG8	120h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	t_xs_fast_x32							Reserved	t_xs_abort_x32						
W																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	t_xs_dll_x32							Reserved	t_xs_x32						
W																
Reset	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31 —	Reserved
30-24	tXS_FAST: Exit Self Refresh to ZQCL, ZQCS and MRS (only CL, WR, RTP and Geardown mode). When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided

Table continues on the next page...

Table continued from the previous page...

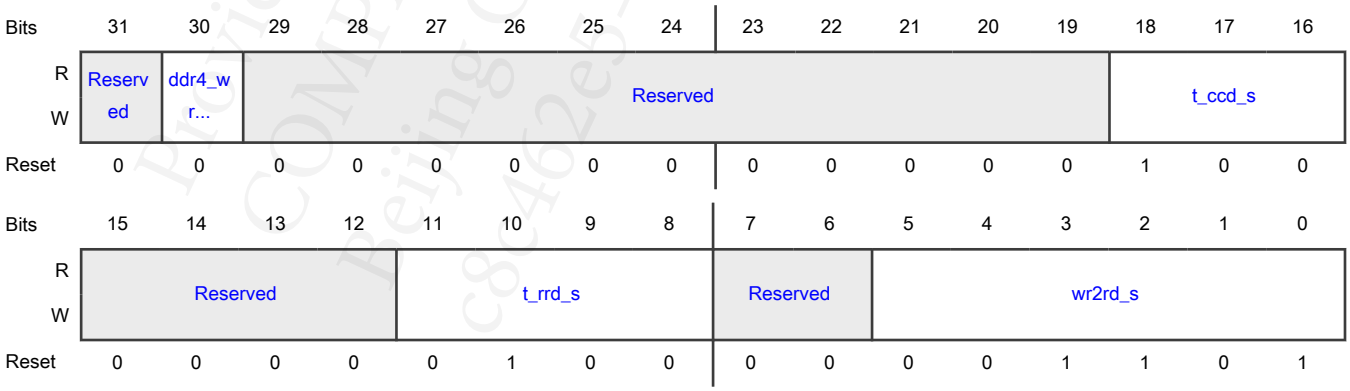
Field	Function
t_xs_fast_x32	by 2 and round up to next integer value. Note: This is applicable to only ZQCL/ZQCS commands. Note: Ensure this is less than or equal to t_xs_x32. Unit: Multiples of 32 DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4
23 —	Reserved
22-16 t_xs_abort_x32	tXS_ABORT: Exit Self Refresh to commands not requiring a locked DLL in Self Refresh Abort. When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided by 2 and round up to next integer value. Note: Ensure this is less than or equal to t_xs_x32. Unit: Multiples of 32 DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4
15 —	Reserved
14-8 t_xs_dll_x32	tXSDLL: Exit Self Refresh to commands requiring a locked DLL. When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided by 2 and round up to next integer value. Note: Used only for DDR2, DDR3 and DDR4 SDRAMs. Unit: Multiples of 32 DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4
7 —	Reserved
6-0 t_xs_x32	tXS: Exit Self Refresh to commands not requiring a locked DLL. When the controller is operating in 1:2 frequency ratio mode, program this to the above value divided by 2 and round up to next integer value. Note: Used only for DDR2, DDR3 and DDR4 SDRAMs. Unit: Multiples of 32 DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4

36.14.61 DRAM Timing 9 (DRAMTMG9)

Offset

Register	Offset
DRAMTMG9	124h

Diagram



Fields

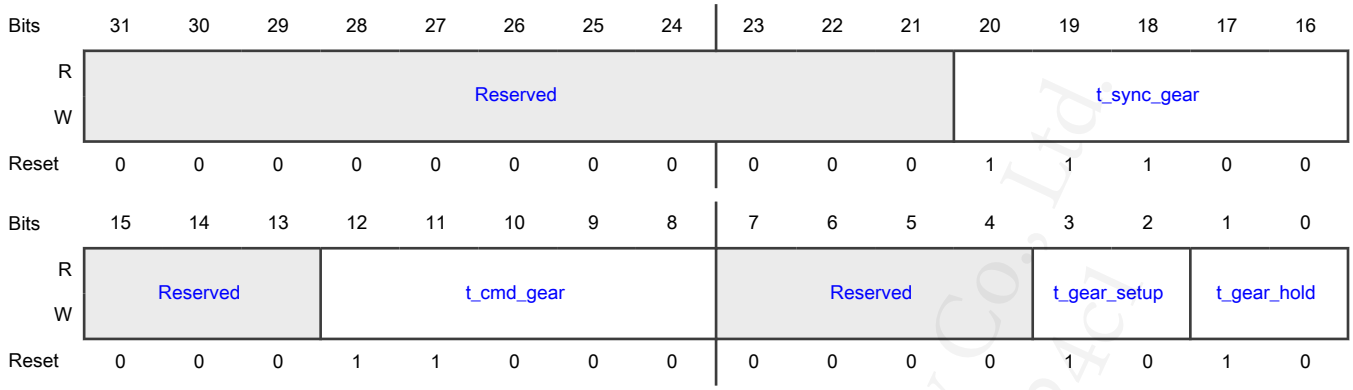
Field	Function
31 —	Reserved
30 ddr4_wr_preamble	DDR4 Write preamble mode - 0: 1tCK preamble - 1: 2tCK preamble Present only with MEMC_FREQ_RATIO=2 Programming Mode: Quasi-dynamic Group 2, Group 4
29-19 —	Reserved
18-16 t_ccd_s	tCCD_S: This is the minimum time between two reads or two writes for different bank group. For bank switching (from bank "a" to bank "b"), the minimum time is this value + 1. When the controller is operating in 1:2 frequency ratio mode, program this to (tCCD_S/2) and round it up to the next integer value. Present only in designs configured to support DDR4. Unit: DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4
15-12 —	Reserved
11-8 t_rrd_s	tRRD_S: Minimum time between activates from bank "a" to bank "b" for different bank group. When the controller is operating in 1:2 frequency ratio mode, program this to (tRRD_S/2) and round it up to the next integer value. Present only in designs configured to support DDR4. Unit: DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4
7-6 —	Reserved
5-0 wr2rd_s	CWL + PL + BL/2 + tWTR_S Minimum time from write command to read command for different bank group. Includes time for bus turnaround, recovery times, and all per-bank, per-rank, and global constraints. Present only in designs configured to support DDR4. Where: - CWL = CAS write latency - PL = Parity latency - BL = burst length. This must match the value programmed in the BL bit of the mode register to the DRAM - tWTR_S = internal write to read command delay for different bank group. This comes directly from the DRAM specification. WTR_S should be increased by one if DDR4 2tCK write preamble is used. When the controller is operating in 1:2 mode, divide the value calculated using the above equation by 2, and round it up to next integer. Unit: DFI clock cycles. Programming Mode: Quasi-dynamic Group 1, Group 2, Group 4

36.14.62 DRAM Timing 10 (DRAMTMG10)

Offset

Register	Offset
DRAMTMG10	128h

Diagram



Fields

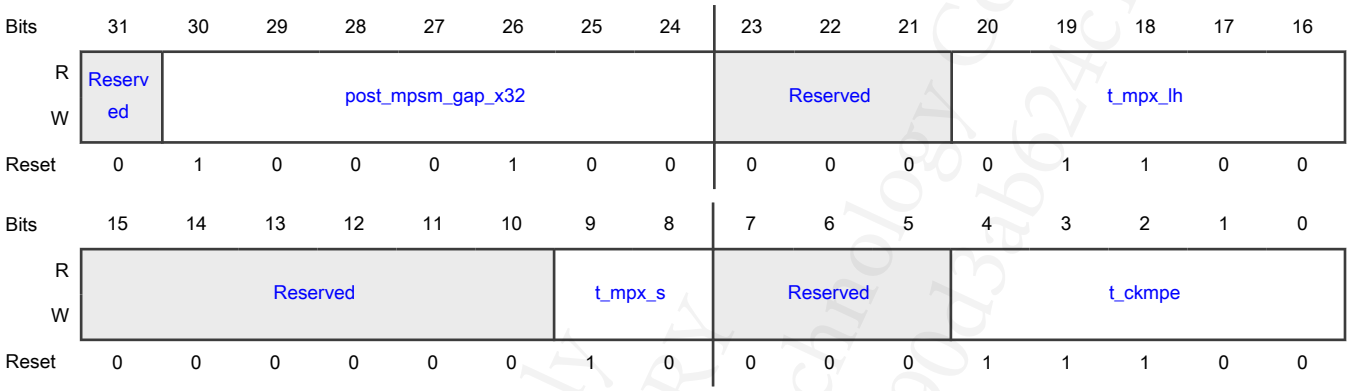
Field	Function
31-21 —	Reserved
20-16 t_sync_gear	Indicates the time between MRS command and the sync pulse time. This must be even number of clocks. For DDR4-2666 and DDR4-3200, this parameter is defined as tMOD(min)+4nCK tMOD(min) is greater of 24nCK or 15ns 15ns / .625ns = 24 Max value for this register is 24+4 = 28 When the controller is operating in 1:2 mode, program this to (tSYNC_GEAR/2) and round it up to the next integer value. Unit: DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4
15-13 —	Reserved
12-8 t_cmd_gear	Sync pulse to first valid command. For DDR4-2666 and DDR4-3200, this parameter is defined as tMOD(min) tMOD(min) is greater of 24nCK or 15ns 15ns / .625ns = 24 Max value for this register is 24 When the controller is operating in 1:2 mode, program this to (tCMD_GEAR/2) and round it up to the next integer value. Unit: DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4
7-4 —	Reserved
3-2 t_gear_setup	Geardown setup time. Minimum value of this register is 1. Zero is invalid. For DDR4-2666 and DDR4-3200, this parameter is defined as 2 clks When the controller is operating in 1:2 frequency ratio mode, program this to (tGEAR_setup/2) and round it up to the next integer value. Unit: DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4
1-0 t_gear_hold	Geardown hold time. Minimum value of this register is 1. Zero is invalid. For DDR4-2666 and DDR4-3200, this parameter is defined as 2 clks When the controller is operating in 1:2 frequency ratio mode, program this to (tGEAR_hold/2) and round it up to the next integer value. Unit: DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4

36.14.63 DRAM Timing 11 (DRAMTMG11)

Offset

Register	Offset
DRAMTMG11	12Ch

Diagram



Fields

Field	Function
31 —	Reserved
30-24 post_mpsm_gap_x32	tXMPDLL: This is the minimum Exit MPSM to commands requiring a locked DLL. When the controller is operating in 1:2 frequency ratio mode, program this to (tXMPDLL/2) and round it up to the next integer value. Present only in designs configured to support DDR4. Unit: Multiples of 32 DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4
23-21 —	Reserved
20-16 t_mpx_lh	tMPX_LH: This is the minimum CS_n Low hold time to CKE rising edge. When the controller is operating in 1:2 frequency ratio mode, program this to RoundUp(tMPX_LH/2)+1. Present only in designs configured to support DDR4. Unit: DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4
15-10 —	Reserved
9-8 t_mpx_s	tMPX_S: Minimum time CS setup time to CKE. When the controller is operating in 1:2 frequency ratio mode, program this to (tMPX_S/2) and round it up to the next integer value. Present only in designs configured to support DDR4. Unit: DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4
7-5	Reserved

Table continues on the next page...

Table continued from the previous page...

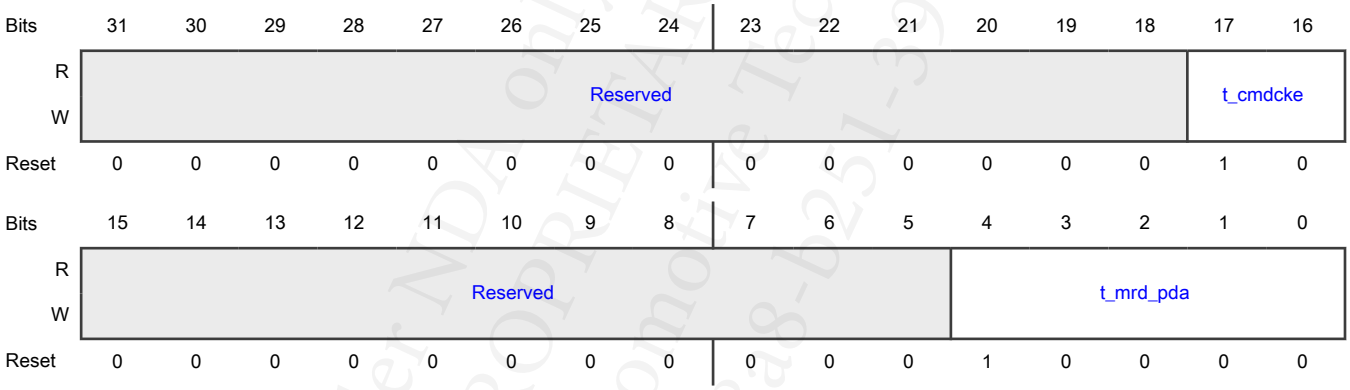
Field	Function
—	
4-0 t_ckmpe	tCKMPE: Minimum valid clock requirement after MPSM entry. Present only in designs configured to support DDR4. When the controller is operating in 1:2 frequency ratio mode, divide the value calculated using the above equation by 2, and round it up to next integer. Unit: DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4

36.14.64 DRAM Timing 12 (DRAMTMG12)

Offset

Register	Offset
DRAMTMG12	130h

Diagram



Fields

Field	Function
31-18 —	Reserved
17-16 t_cmdcke	tCMDCKE Specifies the time interval, in DFI clock cycles, for the delay between a valid command and CKE input low. When the MC is operating at 1:2 frequency ratio, set to RoundUp(max(tESCKE, tCMDCKE)/2). Programming Mode: Quasi-dynamic Group 2, Group 4
15-5 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
4-0 t_mrd_pda	tMRD_PDA Specifies the time interval, in DFI clock cycles, for Mode Register Set command cycle time in PDA mode. When the MC is operating at 1:2 frequency ratio, set to RoundUp(tMRD_PDA/2). Programming Mode: Quasi-dynamic Group 2, Group 4

36.14.65 DRAM Timing 13 (DRAMTMG13)

Offset

Register	Offset
DRAMTMG13	134h

Function
This register applies only to LPDDR4 DRAM.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved	odtloff							Reserved		t_ccd_mw					
Reset	0	0	0	1	1	1	0	0	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved													t_ppd		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31 —	Reserved
30-24 odtloff	tODTLoff Specifies the time interval (latency), in DFI clock cycles, from CAS-2 command to tODTloff reference. When the MC is operating at 1:2 frequency ratio, set to RoundUp(tODTLoff/2). Programming Mode: Quasi-dynamic Group 2, Group 4

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-22 —	Reserved
21-16 t_ccd_mw	tCCDMW Specifies the minimum time interval, in DFI clock cycles, from write or masked write command to masked write command for the same bank. When the MC is operating at 1:2 frequency ratio, set to RoundUp(tCCDMW/2). Programming Mode: Quasi-dynamic Group 2, Group 4
15-3 —	Reserved
2-0 t_ppd	tPPD Specifies the minimum time interval, in DFI clock cycles, from precharge command to precharge command. When the MC is operating at 1:2 frequency ratio, set to RoundUp(tPPD/2). Programming Mode: Quasi-dynamic Group 2, Group 4

36.14.66 DRAM Timing 14 (DRAMTMG14)

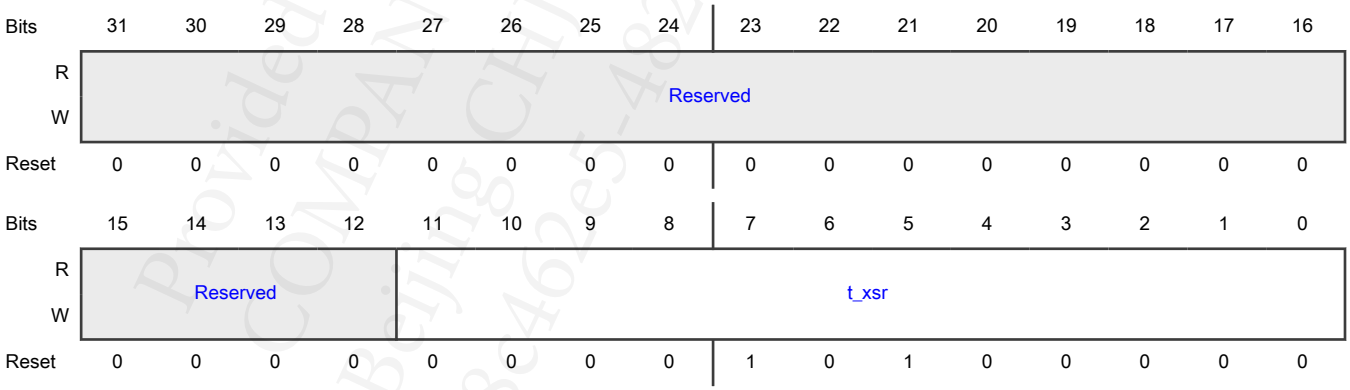
Offset

Register	Offset
DRAMTMG14	138h

Function

This register applies only to mDDR, LPDDR2, LPDDR3, and LPDDR4 DRAM.

Diagram



Provided under NDA of NXP
COMPANY PROPRIETARY
Beijing CHI Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-3901a3ab624c1

Fields

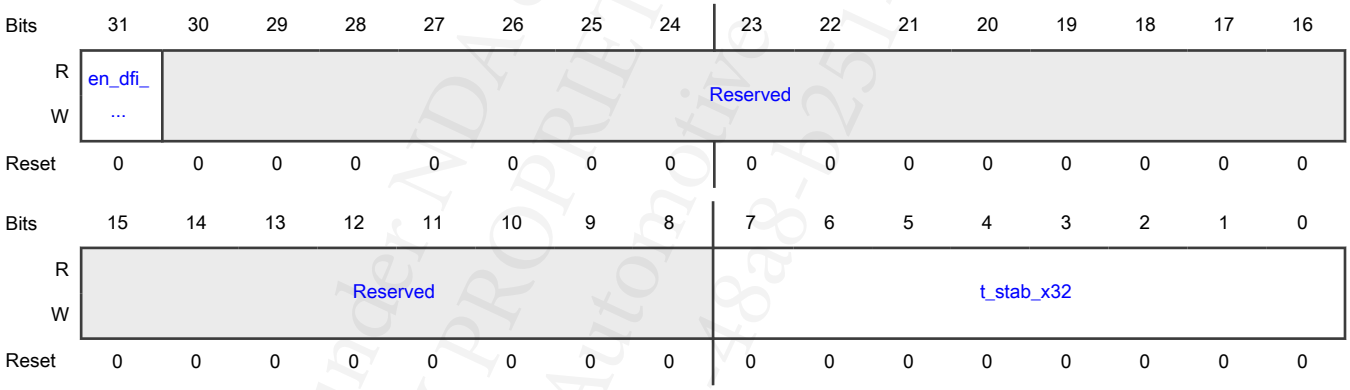
Field	Function
31-12 —	Reserved
11-0 t_xsr	tXSR Specifies the time interval, in DFI clock cycles, from exiting the Self Refresh state to any command. When the MC is operating at 1:2 frequency ratio, divide the calculated value and round up. Programming Mode: Quasi-dynamic Group 2, Group 4

36.14.67 DRAM Timing 15 (DRAMTMG15)

Offset

Register	Offset
DRAMTMG15	13Ch

Diagram



Fields

Field	Function
31 en_dfi_lp_t_stab	- 1 - Enable using tSTAB when exiting DFI LP. Needs to be set when the PHY is stopping the clock during DFI LP to save maximum power. - 0 - Disable using tSTAB when exiting DFI LP Programming Mode: Quasi-dynamic Group 2, Group 4
30-8 —	Reserved
7-0 t_stab_x32	tSTAB: Stabilization time. It is required in the following two cases for DDR3/DDR4 RDIMM : - when exiting power saving mode, if the clock was stopped, after re-enabling it the clock must be stable for a time specified by tSTAB - in the case of input clock frequency change (DDR4) - after issuing control

Table continues on the next page...

Table continued from the previous page...

Field	Function
	words that refers to clock timing (Specification: 6us for DDR3, 5us for DDR4) When the controller is operating in 1:2 frequency ratio mode, program this to recommended value divided by two and round it up to next integer. Unit: Multiples of 32 DFI clock cycles. Programming Mode: Quasi-dynamic Group 2, Group 4

36.14.68 ZQ Control 0 (ZQCTL0)

Offset

Register	Offset
ZQCTL0	180h

Function

This register applies only to DDR3, DDR4, LPDDR2, LPDDR3, and LPDDR4 DRAM.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	dis_aut...	dis_srx...	zq_resi...	dis_mps...	Reserved	t_zq_long_nop										
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						t_zq_short_nop									
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Fields

Field	Function
31 dis_auto_zq	Disable automatic ZQCS|MPC Disables MC generation of the ZQ calibration short (ZQCS | MPC) command. If disabled, software can issue the ZQ calibration command through DBGCMD[zq_calib_short]. If enabled, the MC generates ZQ calibration commands based on ZQCTL1[t_zq_short_interval_x1024]. Programming Mode: Dynamic 0b - Enable 1b - Disable
30	Disable ZQCL|MPC on SR|SRPD exit

Table continues on the next page...

Table continued from the previous page...

Field	Function
dis_srx_zqcl	Disables the issuing of the ZQ calibration (ZQCL|MPC) command on exit from the Self Refresh or Self Refresh Power Down modes. Programming Mode: Quasi-dynamic Group 2, Group 4 0b - Enable 1b - Disable
29 zq_resistor_shared	Shared ZQ resistor Specifies whether to share the ZQ resistor between ranks. If the ZQ resistor is shared, then the MC sends ZQ calibration commands (ZQinit | ZQCL | ZQCS | MPC) to one rank at a time with the necessary time interval (tZQinit | tZQCL | tZQCS | tZQCAL | tZQLAT) to prevent overlap. Programming Mode: Static 0b - No shared 1b - Shared
28 dis_mpsmx_zqcl	Disable ZQCL on MPSM exit Disables the issuing of the ZQ calibration long (ZQCL) command on exit from Maximum Power Saving Mode (MPSM). This field applies only to DDR4 DRAM. **NOTE** If Shared AC mode is enabled (UMCTL2_SHARED_AC == 1), write 1 to this field. Software can send ZQCS after exiting MPSM. Programming Mode: Static 0b - Enable 1b - Disable
27 —	Reserved
26-16 t_zq_long_nop	tZQoper (DDR3, DDR4) | tZQCL (LPDDR2, LPDDR3) | tZQCAL (LPDDR4) Specifies the time interval, in DFI clock cycles, of NOP required after issuing a ZQ calibration long (ZQCL) or ZQ start (MPC) command to the DRAM. When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up. Programming Mode: Static
15-10 —	Reserved
9-0 t_zq_short_nop	tZQCS (DDR3, DDR4, LPDDR2, LPDDR3) | tZQLAT (LPDDR4)

Table continues on the next page...

Table continued from the previous page...

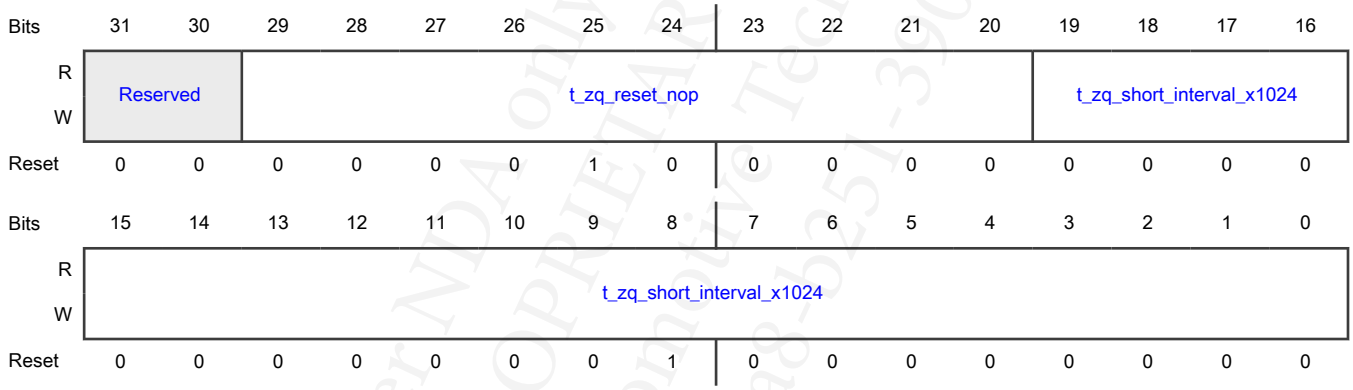
Field	Function
	Specifies the time interval, in DFI clock cycles, of NOP required after issuing a ZQ calibration short (ZQCS) or a ZQ Latch (MPC) command to the DRAM. When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up. Programming Mode: Static

36.14.69 ZQ Control 1 (ZQCTL1)

Offset

Register	Offset
ZQCTL1	184h

Diagram



Fields

Field	Function
31-30 —	Reserved
29-20 t_zq_reset_nop	ZQreset NOP Specifies the time interval, in DFI clock cycles, of NOP required after issuing a ZQ calibration reset (ZQreset) command to DRAM. When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up. This field applies only to LPDDR2, LPDDR3, or LPDDR4 DRAM. Programming Mode: Static
19-0	ZQ calibration short interval

Table continues on the next page...

Table continued from the previous page...

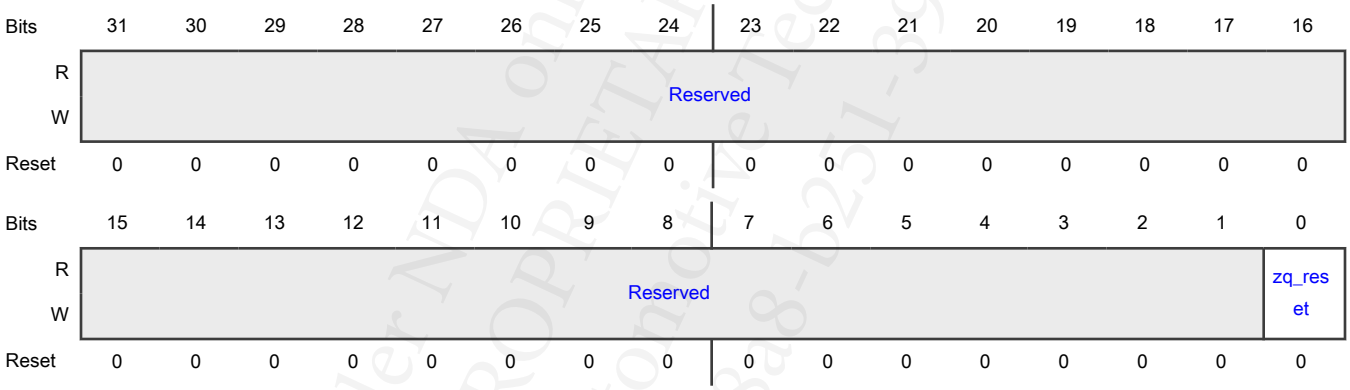
Field	Function
t_zq_short_interval_x1024	Specifies the average time interval, in multiples of 1024 DFI clock cycles, between automatic ZQ calibration (ZQCS | MPC) commands. This field is ignored if automatic ZQ calibration is disabled (ZQCTL0[dis_auto_zq] == 1). This field applies only to DDR3, DDR4, LPDDR2, LPDDR3, or LPDDR4 DRAM. Programming Mode: Static

36.14.70 ZQ Control 2 (ZQCTL2)

Offset

Register	Offset
ZQCTL2	188h

Diagram



Fields

Field	Function
31-1 —	Reserved
0 zq_reset	Setting this register bit to 1 triggers a ZQ Reset operation. When the ZQ Reset operation is complete, the uMCTL2 automatically clears this bit. It is recommended NOT to set this register bit if in Init, in Self-Refresh(except LPDDR4) or SR-Powerdown(LPDDR4) or Deep power-down operating modes. For Self-Refresh(except LPDDR4) or SR-Powerdown(LPDDR4) it will be scheduled after SR(except LPDDR4) or SPRD(LPDDR4) has been exited. For Deep power down, it will not be scheduled, although ZQSTAT.zq_reset_busy will be de-asserted. This is only present for designs supporting LPDDR2/ LPDDR3/LPDDR4 devices. Programming Mode: Dynamic

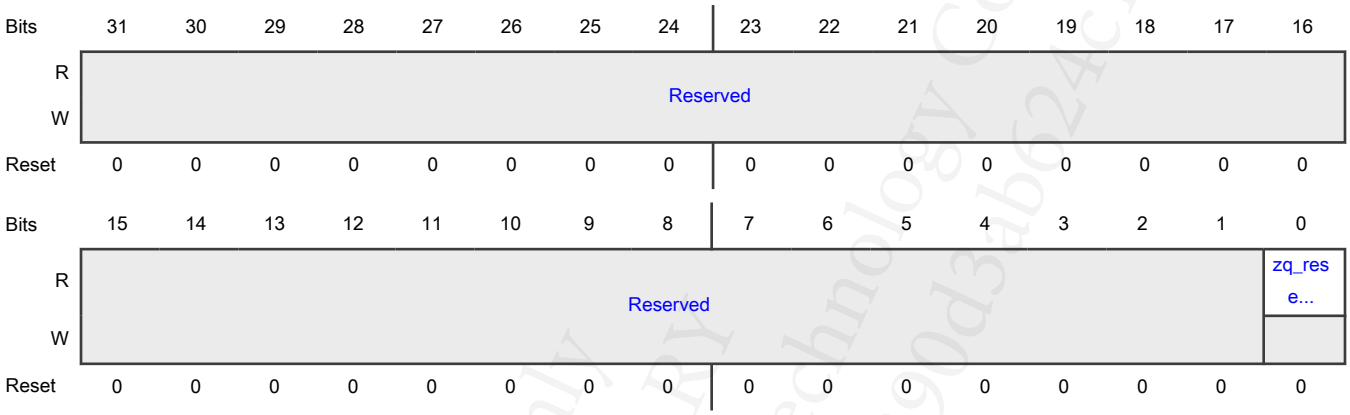
Preliminary NDA only
BOSCH COMPANY PROPRIETARY
c8c0e4821-4848-b251-390d3ab624c1
Bosch Automotive Technology Co., Ltd.

36.14.71 ZQ Status (ZQSTAT)

Offset

Register	Offset
ZQSTAT	18Ch

Diagram



Fields

Field	Function
31-1 —	Reserved
0 zq_reset_busy	SoC core may initiate a ZQ Reset operation only if this signal is low. This signal goes high in the clock after the uMCTL2 accepts the ZQ Reset request. It goes low when the ZQ Reset command is issued to the DRAM and the associated NOP period is over. It is recommended not to perform ZQ Reset commands when this signal is high. - 0 - Indicates that the SoC core can initiate a ZQ Reset operation - 1 - Indicates that ZQ Reset operation is in progress Programming Mode: Dynamic

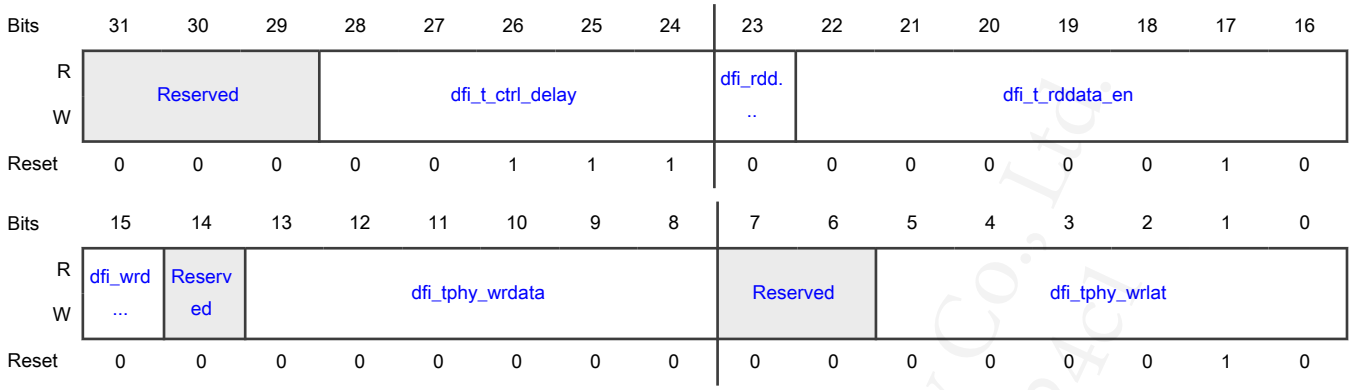
36.14.72 DFI Timing 0 (DFITMG0)

Offset

Register	Offset
DFITMG0	190h

Provided under NDA only
COMPANY CONFIDENTIAL PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4c2-b251-390d34b624c1

Diagram



Fields

Field	Function
31-29 —	Reserved
28-24 dfi_t_ctrl_delay	DFI control delay Specifies the time interval, in DFI clock cycles, between assertion or deassertion of the DFI control signals and the reflection of those control signals at the PHY-DRAM interface. If the DFI clock and the memory clock are not phase-aligned, round this timing parameter up to the next integer. Programming Mode: Quasi-dynamic Group 4 **NOTE** If using RDIMM or LRDIMM, you must increment this value for an extra cycle of latency.
23 dfi_rddata_use_dfi_phy_clk	DFI or DFI PHY clock for read control Selects whether dfi_t_rddata_en represents DFI clock (HDR) or DFI PHY clock (SDR) cycles. Determines whether the MC generates the dfi_rddata_en, dfi_rddata, and dfi_rddata_valid signals based on the DFI clock (HDR) or the DFI PHY clock (SDR). Refer to the PHY specification for the correct value. Programming Mode: Static 0b - DFI clock (HDR) 1b - DFI PHY clock (SDR)
22-16 dfi_t_rddata_en	Time to read data enable Specifies the time interval, in DFI or DFI PHY clock cycles (DFITMG0[dfi_wrdata_use_dfi_phy_clk]), from the assertion of a read command on the DFI interface to the assertion of the dfi_rddata_en signal. This value corresponds to the trddata_en DFI parameter. Refer to the PHY specification for the correct value. Programming Mode: Quasi-dynamic Group 1, Group 4
15	DFI or DFI PHY clock for write control

Table continues on the next page...

Table continued from the previous page...

Field	Function
dfi_wrdata_use_dfi_phy_clk	Selects whether dfi_tphy_wrlat and dfi_tphy_wrdata represent DFI clock (HDR) or DFI PHY clock (SDR) cycles. Determines whether the MC generates the dfi_wrdata_en, dfi_wrdata, and dfi_wrdata_mask signals based on the DFI clock (HDR) or the DFI PHY clock (SDR). Refer to the PHY specification for the correct value. Programming Mode: Static 0b - DFI clock (HDR) 1b - DFI PHY clock (SDR)
14 —	Reserved
13-8 dfi_tphy_wrdata	Write data driven Specifies the time interval, in DFI or DFI PHY clock cycles (DFITMG0[dfi_wrdata_use_dfi_phy_clk]), between assertion of dfi_wrdata_en to the driving of the associated write data on the dfi_wrdata_en signal. This value corresponds to the tphy_wrdata DFI timing parameter. See the PHY specification for the correct value. **NOTE** The maximum supported value is 8. Updating DFITMG0[dfi_tphy_wrlat] or DFITMG0[dfi_wrdata_use_dfi_phy_clk] as part of a frequency change procedure may cause metastability and/or glitches in the dfi_phyupd_ack and dfi_ctrlupd_req signals. To avoid these problems, do this: 1. Write 1b to DFIUPD0[dis_auto_ctrlupd]. 2. Disable the PHY-Update request. Programming mode: Quasi-dynamic Group 4
7-6 —	Reserved
5-0 dfi_tphy_wrlat	Write latency Specifies the time interval, in DFI or DFI PHY clock cycles DFITMG0[dfi_wrdata_use_dfi_phy_clk], from the write command to write data enable (dfi_wrdata_en). This value corresponds to the tphy_wrlat DFI timing parameter. See the PHY specification for the correct value. **NOTE** LPDDR4 does not support values greater than 60. Updating DFITMG0[dfi_tphy_wrlat] or DFITMG0[dfi_wrdata_use_dfi_phy_clk] as part of a frequency change procedure may cause metastability and/or glitches in the dfi_phyupd_ack and dfi_ctrlupd_req signals. To avoid these problems, do this: 1. Write 1'b1 to DFIUPD0[dis_auto_ctrlupd].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	2. Disable the PHY-Update request. Programming mode: Quasi-dynamic Group 1, Group 4

36.14.73 DFI Timing 1 (DFITMG1)

Offset

Register	Offset
DFITMG1	194h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	dfi_t_cmd_lat				Reserved		dfi_t_parin_lat		Reserved			dfi_t_wrdata_delay				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved			dfi_t_dram_clk_disable					Reserved			dfi_t_dram_clk_enable				
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-28 dfi_t_cmd_lat	Command latency Specifies the time interval (latency), in DFI PHY clock cycles, between assertion of the dfi_cs signal and driving of the associated command in CAL mode. The valid range for this value is 0 to 8. This value corresponds to the tCAL parameter. If using PHY CAL mode, set this field to 0. This field applies only to DDR4 DRAM. Programming Mode: Quasi-dynamic Group 2, Group 4
27-26 —	Reserved
25-24 dfi_t_parin_lat	Parity in latency

Table continues on the next page...

Table continued from the previous page...

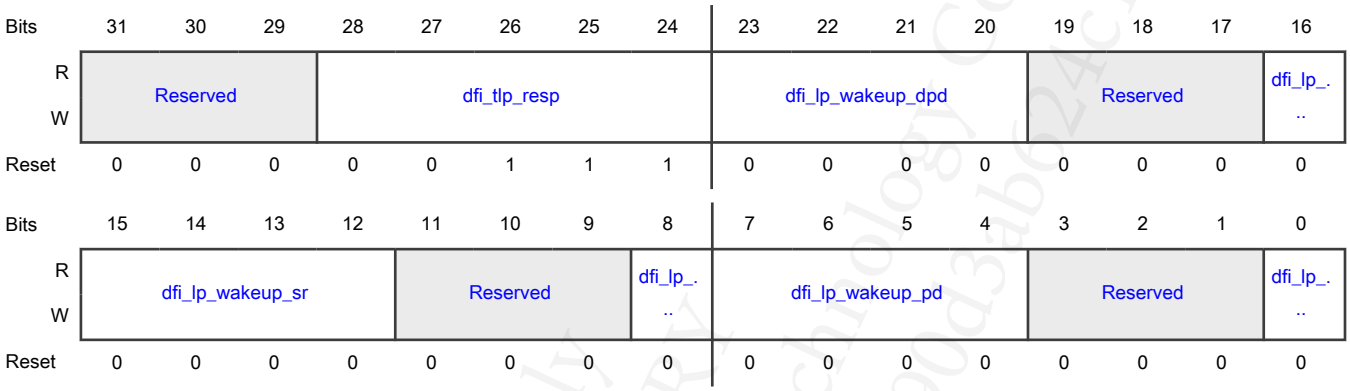
Field	Function
	Specifies the time interval, in DFI PHY clock cycles, between assertion of the dfi_cs signal and driving of the associated dfi_parity_in signal. Programming Mode: Quasi-dynamic Group 4
23-21 —	Reserved
20-16 dfi_t_wrdata_delay	Write delay Specifies the time interval, in DFI clock cycles, between assertion of dfi_wrdata_en and completion of the corresponding write data transfer on the DRAM bus. This value corresponds to the twrdata_delay DFI timing parameter. Refer to the PHY specification for the correct value. When the MC is operating at 1:2 frequency ratio, divide the calculated value by 2 and round up. **NOTE** • If using the DFI PHY clock (DFITMG0[dfi_wrdata_use_dfi_phy_clk]), add 1 to the value. Programming Mode: Quasi-dynamic Group 4
15-13 —	Reserved
12-8 dfi_t_dram_clk_disable	DRAM clock disable latency Specifies the time interval (latency), in DFI clock cycles, from assertion of the dfi_dram_clk_disable signal on the DFI until the clock to the DRAM memory devices at the PHY-DRAM interface maintains a low value. If the DFI clock and the memory clock are not phase-aligned, round this timing parameter up to the next integer. Programming Mode: Quasi-dynamic Group 4
7-5 —	Reserved
4-0 dfi_t_dram_clk_enable	DRAM clock enable latency Specifies the time interval (latency), in DFI clock cycles, from deassertion of the dfi_dram_clk_disable signal on the DFI until the first valid rising edge of the clock to the DRAM memory devices at the PHY-DRAM interface. If the DFI clock and the memory clock are not phase-aligned, round this timing parameter up to the next integer. Programming Mode: Quasi-dynamic Group 4

36.14.74 DFI Low Power Configuration 0 (DFILPCFG0)

Offset

Register	Offset
DFILPCFG0	198h

Diagram



Fields

Field	Function
31-29 —	Reserved
28-24 dfi_tlp_resp	Setting in DFI clock cycles for DFI's tlp_resp time. Same value is used for both Power Down, Self Refresh, Deep Power Down and Maximum Power Saving modes. DFI 2.1 specification onwards, recommends using a fixed value of 7 always. Unit: DFI clock cycles. Programming Mode: Static
23-20 dfi_lp_wakeup_dpd	Value in DFI clock cycles to drive on dfi_lp_wakeup signal when Deep Power Down mode is entered. Determines the DFI's tlp_wakeup time: - 0x0 - 16 cycles - 0x1 - 32 cycles - 0x2 - 64 cycles - 0x3 - 128 cycles - 0x4 - 256 cycles - 0x5 - 512 cycles - 0x6 - 1024 cycles - 0x7 - 2048 cycles - 0x8 - 4096 cycles - 0x9 - 8192 cycles - 0xA - 16384 cycles - 0xB - 32768 cycles - 0xC - 65536 cycles - 0xD - 131072 cycles - 0xE - 262144 cycles - 0xF - Unlimited This is only present for designs supporting mDDR or LPDDR2/LPDDR3 devices. Unit: DFI clock cycles. Programming Mode: Static
19-17 —	Reserved
16 dfi_lp_en_dpd	Enables DFI Low Power interface handshaking during Deep Power Down Entry/Exit. - 0 - Disabled - 1 - Enabled This is only present for designs supporting mDDR or LPDDR2/LPDDR3 devices. Programming Mode: Static
15-12 dfi_lp_wakeup_sr	Value in DFI clock cycles to drive on dfi_lp_wakeup signal when Self Refresh mode is entered. Determines the DFI's tlp_wakeup time: - 0x0 - 16 cycles - 0x1 - 32 cycles - 0x2 - 64 cycles - 0x3 - 128 cycles - 0x4 - 256 cycles - 0x5 - 512 cycles - 0x6 - 1024 cycles - 0x7 - 2048 cycles - 0x8 - 4096 cycles - 0x9 - 8192 cycles - 0xA - 16384 cycles - 0xB - 32768 cycles - 0xC - 65536 cycles - 0xD - 131072 cycles - 0xE - 262144 cycles - 0xF - Unlimited Unit: DFI clock cycles. Programming Mode: Static

Table continues on the next page...

Table continued from the previous page...

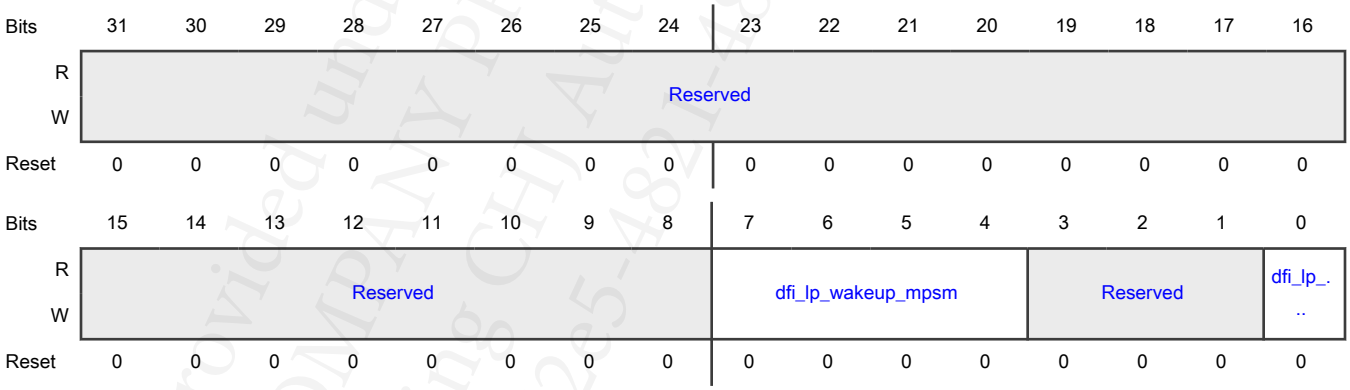
Field	Function
11-9 —	Reserved
8 dfi_lp_en_sr	Enables DFI Low Power interface handshaking during Self Refresh Entry/Exit. - 0 - Disabled - 1 - Enabled Programming Mode: Static
7-4 dfi_lp_wakeup_pd	Value in DFI clock cycles to drive on dfi_lp_wakeup signal when Power Down mode is entered. Determines the DFI's tlp_wakeup time: - 0x0 - 16 cycles - 0x1 - 32 cycles - 0x2 - 64 cycles - 0x3 - 128 cycles - 0x4 - 256 cycles - 0x5 - 512 cycles - 0x6 - 1024 cycles - 0x7 - 2048 cycles - 0x8 - 4096 cycles - 0x9 - 8192 cycles - 0xA - 16384 cycles - 0xB - 32768 cycles - 0xC - 65536 cycles - 0xD - 131072 cycles - 0xE - 262144 cycles - 0xF - Unlimited Unit: DFI clock cycles. Programming Mode: Static
3-1 —	Reserved
0 dfi_lp_en_pd	Enables DFI Low Power interface handshaking during Power Down Entry/Exit. - 0 - Disabled - 1 - Enabled Programming Mode: Static

36.14.75 DFI Low Power Configuration 1 (DFILPCFG1)

Offset

Register	Offset
DFILPCFG1	19Ch

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-b251-39013ab6...

Fields

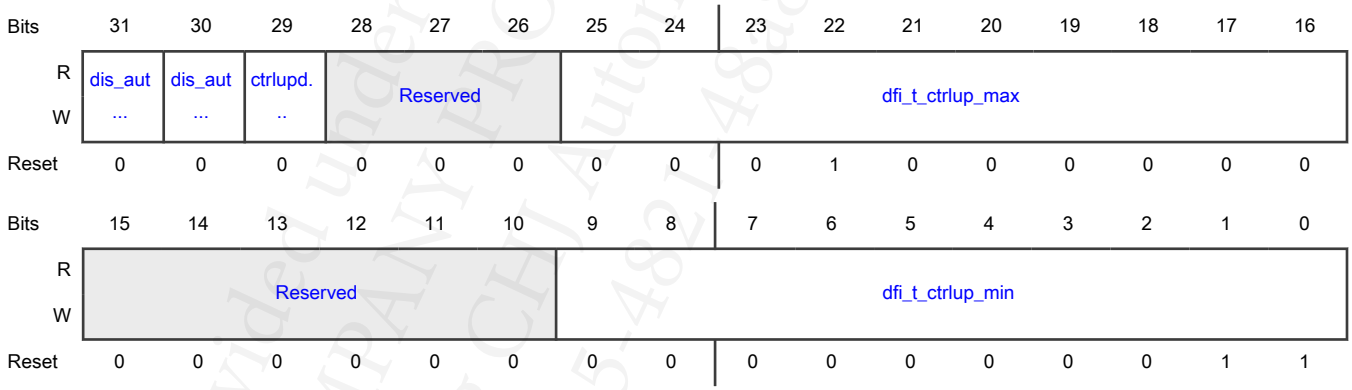
Field	Function
31-8 —	Reserved
7-4 dfi_lp_wakeup_mpsm	Value in DFI clock cycles to drive on dfi_lp_wakeup signal when Maximum Power Saving Mode is entered. Determines the DFI's tlp_wakeup time: - 0x0 - 16 cycles - 0x1 - 32 cycles - 0x2 - 64 cycles - 0x3 - 128 cycles - 0x4 - 256 cycles - 0x5 - 512 cycles - 0x6 - 1024 cycles - 0x7 - 2048 cycles - 0x8 - 4096 cycles - 0x9 - 8192 cycles - 0xA - 16384 cycles - 0xB - 32768 cycles - 0xC - 65536 cycles - 0xD - 131072 cycles - 0xE - 262144 cycles - 0xF - Unlimited This is only present for designs supporting DDR4 devices. Unit: DFI clock cycles. Programming Mode: Static
3-1 —	Reserved
0 dfi_lp_en_mpsm	Enables DFI Low Power interface handshaking during Maximum Power Saving Mode Entry/Exit. - 0 - Disabled - 1 - Enabled This is only present for designs supporting DDR4 devices. Programming Mode: Static

36.14.76 DFI Update 0 (DFIUPD0)

Offset

Register	Offset
DFIUPD0	1A0h

Diagram



Fields

Field	Function
31 dis_auto_ctrlupd	Disable automatic update request

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Disables automatic MC generation of the update request. When disabled, the core must issue the update request (DBGCMD[ctrlupd]). Programming Mode: Quasi-dynamic Group 2 0b - Enable 1b - Disable
30 dis_auto_ctrlupd_srx	Disable automatic update request on SRX Disables automatic MC generation of the update request on exit from Self Refresh state. Programming Mode: Static 0b - Enable 1b - Disable
29 ctrlupd_pre_srx	Update request on SRX Specifies whether the MC sends an update request before or after existing the Self Refresh state. If automatic update on Self Refresh exit is disabled (dis_auto_ctrlupd_srx == 1), this field has no effect. Programming Mode: Static 0b - After 1b - After
28-26 —	Reserved
25-16 dfi_t_ctrlup_max	Maximum update request time Specifies the maximum time interval, in DFI clock cycles, that the MC controller can assert the udate request (dfi_ctrlupd_req) signal. The lowest valid value is 40h. Programming Mode: Static
15-10 —	Reserved
9-0 dfi_t_ctrlup_min	Update request latency Specifies the minimum time interval, in DFI clock cycles, for assertion of the MC-initiated update request (dfi_ctrlupd_req) signal. The MC expects the PHY to respond within this interval. If the PHY does not respond, the MC deasserts the update request after dfi_t_ctrlup_min plus 2 cycles. The lowest valid value is 3. Programming Mode: Static

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e5-4821-48a8-b251-39d3ab624c1

36.14.77 DFI Update 1 (DFIUPD1)

Offset

Register	Offset
DFIUPD1	1A4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								dfi_t_ctrlupd_interval_min_x1024							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								dfi_t_ctrlupd_interval_max_x1024							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-24 —	Reserved
23-16 dfi_t_ctrlupd_interval_min_x1024	Minimum update interval Specifies the minimum time interval, in multiples of 1024 DFI clock cycles, between MC-initiated DFI update requests (generated when the MC is idle). This timer resets with each update request; when the timer expires, the MC sends an update request (dfi_ctrlupd_req) and blocks traffic until it receives the update acknowledgement (dfi_ctrlupd_ackx). Set this value higher to reduce the frequency of update requests, which can have a small impact on the latency of the first read request when the MC is idle. The minimum allowed value is 1. **NOTE** The value must be less than the value in dfi_t_ctrlupd_interval_max_x1024. Programming Mode: Static
15-8 —	Reserved
7-0	Maximum update interval

Table continues on the next page...

Table continued from the previous page...

Field	Function
dfi_t_ctrlupd_interval_max_x1024	Specifies the maximum time interval, in multiples of 1024 DFI clock cycles, between MC-initiated DFI update requests (generated when the MC is idle). This timer resets with each update request; when the timer expires, the MC sends an update request (dfi_ctrlupd_req) and blocks traffic until it receives the update acknowledgement (dfi_ctrlupd_ackx). PHY can use this idle time to recalibrate the delay lines to the DLLs. It can also use the time to reset the PHY FIFO counters in the event of data capture errors. Updates are needed to maintain calibration over PVT, but frequent updates may impact performance. The minimum allowed value is 1. **NOTE** The value must be greater than the value in dfi_t_ctrlupd_interval_min_x1024. Programming Mode: Static

36.14.78 DFI Update 2 (DFIUPD2)

Offset

Register	Offset
DFIUPD2	1A8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	dfi_phy ...	Reserved														
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 dfi_phyupd_en	PHY update acknowledge enable Enables support for acknowledgement of PHY-initiated udpate requests. Programming Mode: Static

Table continues on the next page...

Table continued from the previous page...

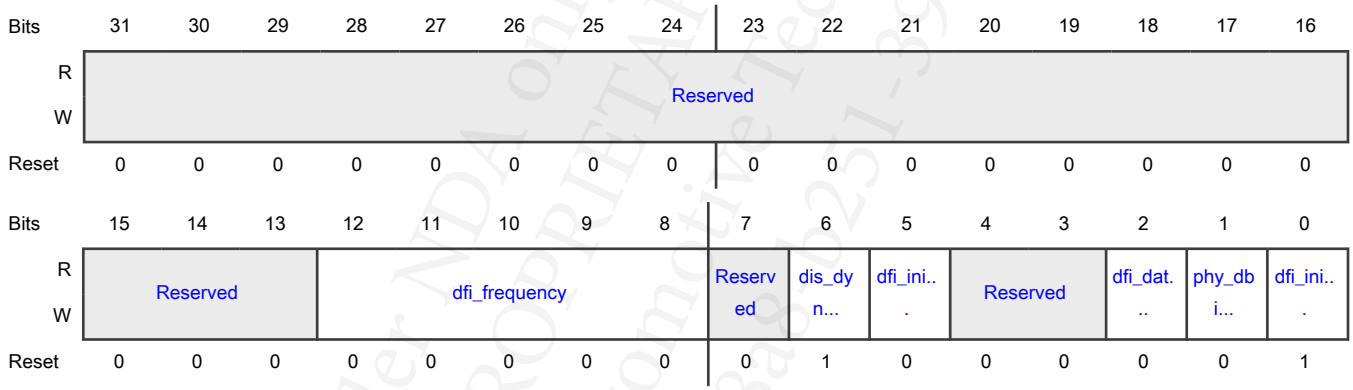
Field	Function
	0b - Disable 1b - Enable
30-0 —	Reserved

36.14.79 DFI Miscellaneous Control (DFIMISC)

Offset

Register	Offset
DFIMISC	1B0h

Diagram



Fields

Field	Function
31-13 —	Reserved
12-8 dfi_frequency	Operating frequency Specifies the operating frequency of the system. The PHY defines the number of supported frequencies and the mapping of signal values to clock frequencies. Programming Mode: Quasi-dynamic Group 1
7 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 dis_dyn_adr_tri	Disable Dynamic Tristating Disables PHY-specific Dynamic Tristating. If Dynamic Tristating is enabled, the MC issues a special IDLE command on the DFI while dfi_cs is inactive. If the PHY detects the special IDLE command, it makes the command/address bus high impedance (Hi-Z). Dynamic Tristating works only if the MC is operating at 1:2 frequency ratio, regardless of MSTR[en_2t_timing_mode]. Therefore, it does not work if either of the following conditions is true: • MEMC_FREQ_RATIO == 1 • MEMC_PROG_FREQ_RATIO == 1 AND MSTR[frequency_ratio] == 1 The special IDLE command means the following codes with the case where all the dfi_cs is 1: • (phase 0 and 1) dfi_ras_n=1 • (phase 0 and 1) dfi_cas_n= 1 • (phase 0 and 1) dfi_we_n= 1 • (phase 0 and 1) dfi_bank [0]= 0 • (phase 0 and 1) dfi_act_n= 1 Programming Mode: Quasi-dynamic Group 3 0b - Enable 1b - Disable
5 dfi_init_start	PHY init start Triggers the PHY init start request. if " DFIMISC[dfi_init_start] is triggered ,while dfi_phymstr_req (PHY request control of DFI interface) is active, it can result in the PHY Master request being acknowledged(dfi_phymstr_ack=1) while dfi_init_start is active. This simultaneously occurring of both these signals is illegal for the Synopsys PHYs. This can lead to unexpected behavior / protocol violation. Therefore, before writing 1 to "DFIMISC[dfi_init_start]" ,write 0 to DFIPHYMSTR[dfi_phymstr_en] to disable the PHY Master. Programming Mode: Quasi-dynamic Group 3
4-3 —	Reserved
2 dfi_data_cs_polarity	Chip select polarity Specifies the polarity of the dfi_wrdata_cs and dfi_rddata_cs signals. Programming Mode: Static 0b - Active low 1b - Active high
1	DBI mode

Table continues on the next page...

Table continued from the previous page...

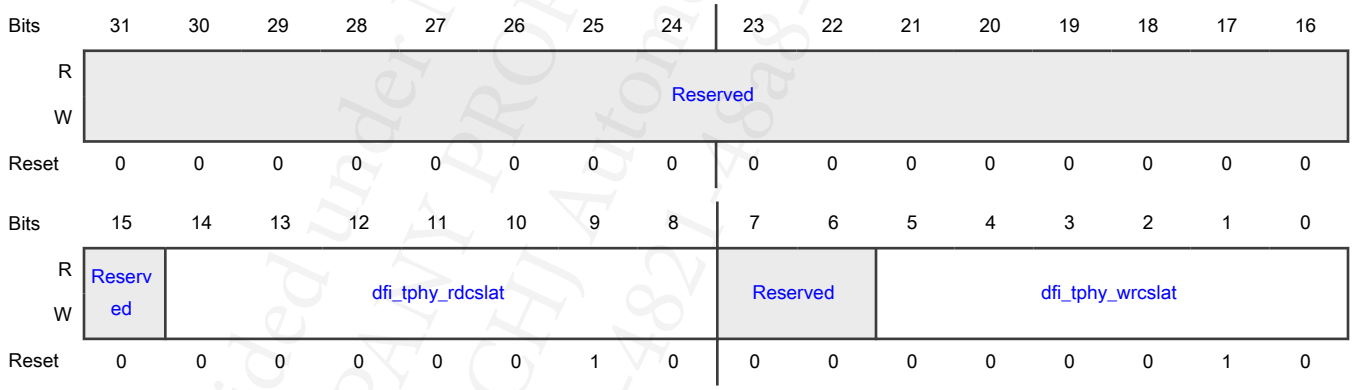
Field	Function
phy_dbi_mode	Specifies whether data bus inversion (DBI) is implemented in the MC or PHY. This field applies only to DDR4 and LPDDR4 DRAM. Programming Mode: Static 0b - MC 1b - PHY
0 dfi_init_complete_en	PHY initialization complete enable Enables the triggering of DRAM initialization on assertion of the dfi_init_complete signal. Programming Mode: Quasi-dynamic Group 3 0b - Disable 1b - Enable

36.14.80 DFI Timing 2 (DFITMG2)

Offset

Register	Offset
DFITMG2	1B4h

Diagram



Fields

Field	Function
31-15 —	Reserved
14-8	DFI read to chip select latency

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4821-48a8-b251-390a3ab694c1

Table continued from the previous page...

Field	Function
dfi_tphy_rdcslat	Specifies the time interval, in DFI PHY clock cycles, between sending a read command on the DFI control interface and assertion of the dfi_rddata_cs signal. This value corresponds to the tphy_rdcslat DFI timing parameter. Programming Mode: Quasi-dynamic Group 2, Group 4
7-6 —	Reserved
5-0 dfi_tphy_wrcslat	DFI write to chip select latency Specifies the time interval (latency), in DFI PHY clock cycles, between sending a write command on the DFI control interface and assertion of the associated dfi_wrdata_cs signal. This value corresponds to the tphy_wrcslat DFI timing parameter. Refer to the PHY specification for the correct value. Programming Mode: Quasi-dynamic Group 2, Group 4

36.14.81 DFI Timing 3 (DFITMG3)

Offset

Register	Offset
DFITMG3	1B8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								dfi_t_geardown_delay							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5 —	Reserved

Table continues on the next page...

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4848-b251-390d3ab624c1

Table continued from the previous page...

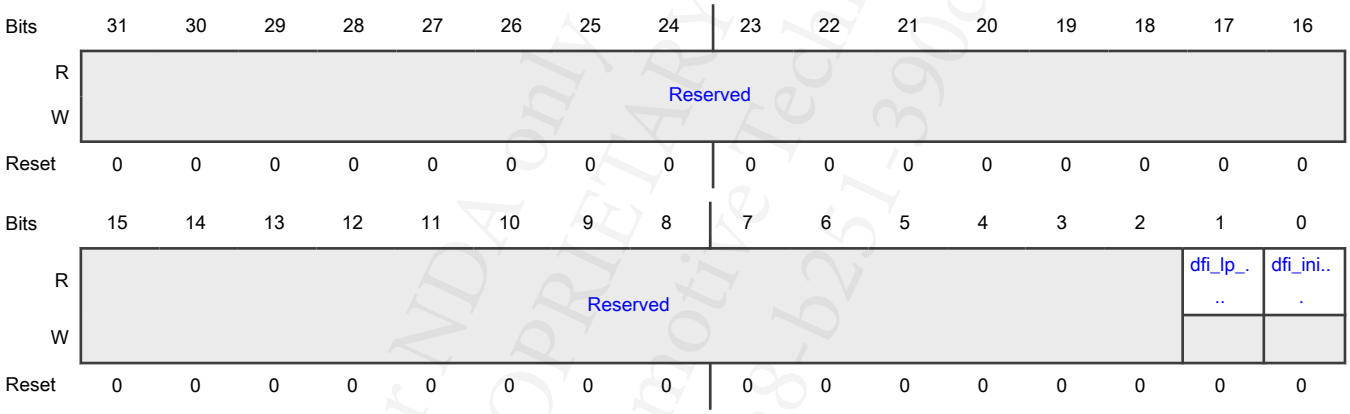
Field	Function
4-0 dfi_t_geardown_delay	The delay from dfi_geardown_en assertion to the time of the PHY being ready to receive commands. Refer to PHY specification for correct value. When the controller is operating in 1:2 frequency ratio mode, program this to (tgeardown_delay/2) and round it up to the next integer value. Unit: DFI clock cycles. Programming Mode: Static

36.14.82 DFI Status (DFISTAT)

Offset

Register	Offset
DFISTAT	1BCh

Diagram



Fields

Field	Function
31-2 —	Reserved
1 dfi_lp_ack	Stores the value of the dfi_lp_ack input to the controller. Programming Mode: Dynamic
0 dfi_init_complete	The status flag register which announces when the DFI initialization has been completed. The DFI INIT triggered by dfi_init_start signal and then the dfi_init_complete flag is polled to know when the initialization is done. Programming Mode: Dynamic

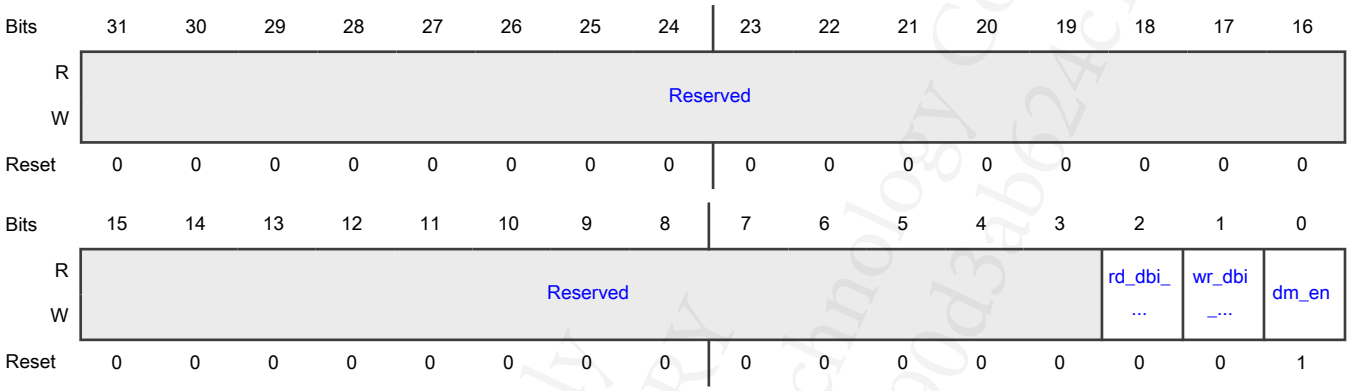
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-1821-48a8-b251-390d3ab624c1

36.14.83 DM/DBI Control (DBICTL)

Offset

Register	Offset
DBICTL	1C0h

Diagram



Fields

Field	Function
31-3 —	Reserved
2 rd_dbi_en	Read DBI enable Enables read data bus inversion (DBI) in the MC. Depending on the type of DRAM, follow these guidelines to set this value to the corresponding DRAM Mode Register: • DDR4: Set to the same value as MR5[A12]. When using x4 devices, set this signal to 0. • LPDDR4: Set to the same value as MR3[6]. Programming Mode: Quasi-dynamic Group 1 0b - Disable 1b - Enable
1 wr_dbi_en	Write DBI enable Enables write data bus inversion (DBI) in the MC. Depending on the type of DRAM, follow these guidelines to set this value to the corresponding DRAM Mode Register: • DDR4: Set to the same value as MR5[A11]. When using x4 devices, set this signal to 0. • LPDDR4: Set to the same value as MR3[7].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Programming Mode: Quasi-dynamic Group 1 0b - Disable 1b - Enable
0 dm_en	Data mask enable Enables data mask. Depending on the type of DRAM, follow these guidelines to set this value to the corresponding DRAM Mode Register: • DDR4: Set to the same value as MR5[A10]. When using x4 devices, set this signal to 0. • LPDDR4: Set to the inverted value of MR13[5], which has the opposite polarity of this signal. Programming Mode: Static 0b - Disable 1b - Enable

36.14.84 DFI PHY Master (DFIPHYMSTR)

Offset

Register	Offset
DFIPHYMSTR	1C4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved															dfi_phy ...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-1	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a8-b251-390d3a0624c1

Table continued from the previous page...

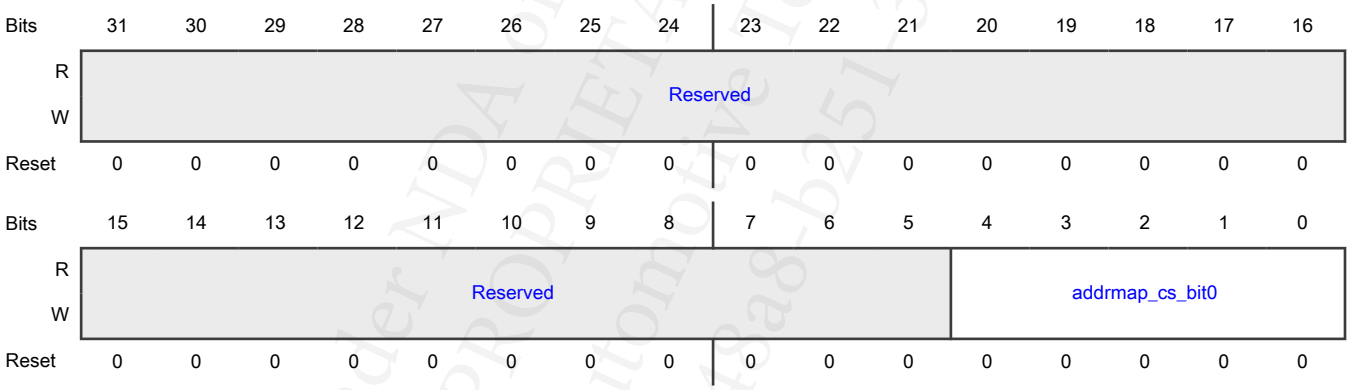
Field	Function
—	
0 dfi_phymstr_en	Enables the PHY Master Interface. Programming Mode: Static 0b - Disable 1b - Enable

36.14.85 Address Map 0 (ADDRMAP0)

Offset

Register	Offset
ADDRMAP0	200h

Diagram



Fields

Field	Function
31-5 —	Reserved
4-0 addrmap_cs_bit0	Rank address bit 0 Selects the host interface (HIF) address bit to use as the rank address bit 0. The valid range is 0 to 29, and 31. The internal base is 6. The selected HIF address bit is determined by adding the internal base to the value of this field. If you set this to 31, then the rank address bit 0 is set to 0. Programming Mode: Static

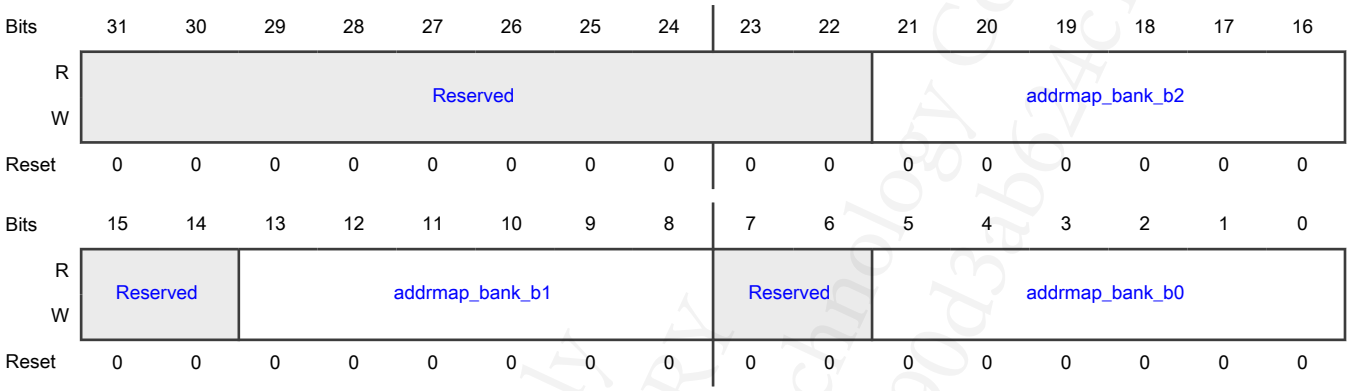
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c52e5-4821-43a8-b251-390d3ab624c1

36.14.86 Address Map 1 (ADDRMAP1)

Offset

Register	Offset
ADDRMAP1	204h

Diagram



Fields

Field	Function
31-22 —	Reserved
21-16 addrmap_bank_b2	Bank address bit 2 Selects the host interface (HIF) address bit to use as the bank address bit 2. The valid range is 0 to 32, and 63. The internal base is 4. The selected HIF address bit is determined by adding the internal base to the value of this field. If you set this to 63, then the bank address bit 2 is set to 0.
15-14 —	Reserved
13-8 addrmap_bank_b1	Bank address bit 1 Selects the host interface (HIF) address bit to use as the bank address bit 1. The valid range is 0 to 32, and 63. The internal base is 3. The selected HIF address bit is determined by adding the internal base to the value of this field. If you set this to 63, then the bank address bit 1 is set to 0. Programming Mode: Static
7-6	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5-0 addrmap_bank_b0	Bank address bit 0 Selects the host interface (HIF) address bit to use as the bank address bit 0. The valid range is 0 to 32, and 63. The internal base is 2. The selected HIF address bit is determined by adding the internal base to the value of this field. If you set this to 63, then the bank address bit 0 is set to 0. Programming Mode: Static

36.14.87 Address Map 2 (ADDRMAP2)

Offset

Register	Offset
ADDRMAP2	208h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				addrmap_col_b5				Reserved				addrmap_col_b4			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				addrmap_col_b3				Reserved				addrmap_col_b2			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 addrmap_col_b5	Column address bit 5 - Full bus width mode: Selects the HIF address bit used as column address bit 5. - Half bus width mode: Selects the HIF address bit used as column address bit 6. - Quarter bus width mode: Selects the HIF address bit used as column address bit 7 . Valid Range: 0 to 7, and 15 Internal Base: 5 The selected HIF

Table continues on the next page...

Table continued from the previous page...

Field	Function
	address bit is determined by adding the internal base to the value of this field. If unused, set to 15 and then this column address bit is set to 0. Programming Mode: Static
23-20 —	Reserved
19-16 addrmap_col_b4	Column address bit 4 - Full bus width mode: Selects the HIF address bit used as column address bit 4. - Half bus width mode: Selects the HIF address bit used as column address bit 5. - Quarter bus width mode: Selects the HIF address bit used as column address bit 6. Valid Range: 0 to 7, and 15 Internal Base: 4 The selected HIF address bit is determined by adding the internal base to the value of this field. If unused, set to 15 and then this column address bit is set to 0. Programming Mode: Static
15-13 —	Reserved
12-8 addrmap_col_b3	Column address bit 3 - Full bus width mode: Selects the HIF address bit used as column address bit 3. - Half bus width mode: Selects the HIF address bit used as column address bit 4. - Quarter bus width mode: Selects the HIF address bit used as column address bit 5. Valid Range: 0 to 7, x. x indicate a valid value in inline ECC configuration. Internal Base: 3 The selected HIF address bit is determined by adding the internal base to the value of this field. Note, if UMCTL2_INCL_ARB=1, MEMC_BURST_LENGTH=16, Full bus width (MSTR.data_bus_width=00) and BL16 (MSTR.burst_rdwr=1000), it is recommended to program this to 0. In Inline ECC configuration (MEMC_INLINE_ECC=1) and ECC is enabled (ECCCFG0.ecc_mode>0), if MEMC_BURST_LENGTH is 16, number of column address is 10 and work on half bus width mode, column bit 8, column bit 7 and column bit 3 must map to the highest 3 valid HIF address bits. This register need be set to x. (x = the highest valid HIF address bit - internal base - 2) Programming Mode: Static
7-4 —	Reserved
3-0 addrmap_col_b2	Column address bit 2 Depending on the bus width mode, selects the host interface (HIF) address bit to use as: • Full bus width: column address bit 2 • Half bus width: column address bit 3 • Quarter bus width: column address bit 4 The valid range is 0 to 7. The internal base is 2. The selected HIF address bit is determined by adding the internal base to the value of this field. If the MC includes a multi-port arbiter (UMCTL2_INCL_ARB == 1) and the burst length is 8 (MEMC_BURST_LENGTH == 8), you must set this field to 0 unless: • Half or Quarter bus width (MSTR[data_bus_width] != 00) AND

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Burst length expansion mode enabled (PCCFG[bl_exp_mode] == 1) AND • Either: — DDR4: ADDRMAP8[addrmap_bg_b0] == 0 — LPDDR4: ADDRMAP1[addrmap_bank_b0] == 0 If the MC includes a multi-port arbiter (UMCTL2_INCL_ARB == 1) and the burst length (BL) is 16 (MEMC_BURST_LENGTH == 16), you must set this field to 0 unless: • Half or Quarter bus width (MSTR[data_bus_width] != 00) AND • BL expansion mode enabled (PCCFG[bl_exp_mode] == 1) AND • DDR4: ADDRMAP8[addrmap_bg_b0] == 0 Otherwise, follow these recommendations: • If using Full bus width (MSTR[data_bus_width] == 00) AND BL is 8 (MEMC_BURST_LENGTH == 8), set to 0 so HIF[2] maps to column address bit 2. • If using Full bus width (MSTR[data_bus_width] == 00) AND BL is 16 (MEMC_BURST_LENGTH == 16), set to 0 so HIF[2] maps to column address bit 2. • If using Half bus width (MSTR[data_bus_width] == 01) AND BL is 16 (MEMC_BURST_LENGTH == 16), set to 0 so HIF[2] maps to column address bit 3. Programming Mode: Static

36.14.88 Address Map 3 (ADDRMAP3)

Offset

Register	Offset
ADDRMAP3	20Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved			addrmap_col_b9					Reserved			addrmap_col_b8				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved			addrmap_col_b7					Reserved			addrmap_col_b6				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c84a2e5-4821-48a8-b251-6da3ab624c1

Fields

Field	Function
31-29 —	Reserved
28-24 addrmap_col_b9	Column address bit 9 Depending on the bus width mode, selects the host interface (HIF) address bit to use as: • Full bus width: column address bit 9 • Half bus width: column address bit 11 (for LPDDR2 and LPDDR3: bit 10) • Quarter bus width: column address bit 13 (for LPDDR2 and LPDDR3: bit 11) The valid values are 0 to 7, 31, and x, where x is a valid value in the inline ECC configuration. The internal base is 9. The MC determines the selected HIF address bit by adding the internal base to the value of this field. If not used, write 31 (11111b) to this field. In this case, the HIF address bit to use is 0. If the following conditions apply, the highest three column address bits must map to the highest three valid HIF address bits: • Inline ECC configuration (MEMC_INLINE_ECC == 1) • ECC enabled (ECCCFG0[ecc_mode]>0) If column bit 9 is the highest column address bit, you must set this value to the highest valid HIF address bit minus the internal base (highest HIF bit – 9) If column bit 9 is the second highest column address bit, you must set this value to the second highest valid HIF address bit minus the internal base minus one (highest HIF bit – 9 – 1) If column bit 9 is the third highest column address bit, you must set this value to the highest valid HIF address bit minus the internal base minus two (highest HIF bit – 9 – 2) **NOTE** For DDR2, DDR3, and mDDR (per the JEDEC specifications), column address bit 10 is reserved for indicating auto-precharge, and therefore no source address bit can be mapped to column address bit 10. LPDDR2 and LPDDR3 have a dedicated bit in the CA bus for auto-precharge, so these DRAM types use column address bit 10. Programming Mode: Static
23-21 —	Reserved
20-16 addrmap_col_b8	Column address bit 8 Depending on the bus width mode, selects the host interface (HIF) address bit to use as: • Full bus width: column address bit 8 • Half bus width: column address bit 9 • Quarter bus width: column address bit 11 (for LPDDR2 and LPDDR3: bit 10)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The valid values are 0 to 7, 31, and x, where x is a valid value in the inline ECC configuration. The internal base is 8. The MC determines the selected HIF address bit by adding the internal base to the value of this field. If not used, write 31 (11111b) to this field. In this case, the HIF address bit to use is 0. If the following conditions apply, the highest three column address bits must map to the highest three valid HIF address bits: • Inline ECC configuration (MEMC_INLINE_ECC == 1) • ECC enabled (ECCCFG0[ecc_mode]>0) If column bit 8 is the second highest column address bit, you must set this value to the second highest valid HIF address bit minus the internal base minus one (highest HIF bit – 8 – 1) If column bit 8 is the third highest column address bit, you must set this value to the highest valid HIF address bit minus the internal base minus two (highest HIF bit – 8 – 2) **NOTE** For DDR2, DDR3, and mDDR (per the JEDEC specifications), column address bit 10 is reserved for indicating auto-precharge, and therefore no source address bit can be mapped to column address bit 10. LPDDR2 and LPDDR3 have a dedicated bit in the CA bus for auto-precharge, so these DRAM types use column address bit 10. Programming Mode: Static
15-13 —	Reserved
12-8 addrmap_col_b7	Column address bit 7 Depending on the bus width mode, selects the host interface (HIF) address bit to use as: • Full bus width: column address bit 7 • Half bus width: column address bit 8 • Quarter bus width: column address bit 9 The valid values are 0 to 7, 31, and x, where x is a valid value in the inline ECC configuration. The internal base is 7. The MC determines the selected HIF address bit by adding the internal base to the value of this field. If not used, write 31 (11111b) to this field. In this case, the HIF address bit to use is 0. If the following conditions apply, the highest three column address bits must map to the highest three valid HIF address bits: • Inline ECC configuration (MEMC_INLINE_ECC == 1) • ECC enabled (ECCCFG0[ecc_mode]>0) If column bit 7 is the third highest column address bit, you must set this value to the highest valid HIF address bit minus the internal base minus two (highest HIF bit – 7 – 2) Programming Mode: Static

Table continues on the next page...

Table continued from the previous page...

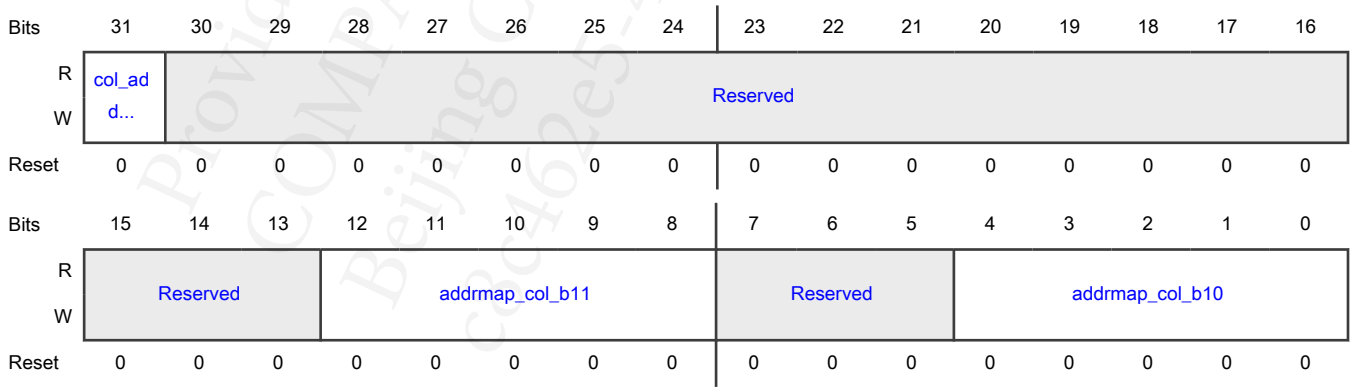
Field	Function
7-5 —	Reserved
4-0 addrmap_col_b6	Column address bit 6 Depending on the bus width mode, selects the host interface (HIF) address bit to use as: • Full bus width: column address bit 6 • Half bus width: column address bit 7 • Quarter bus width: column address bit 8 The valid values are 0 to 7, 31, and x, where x is a valid value in the inline ECC configuration. The internal base is 6. The MC determines the selected HIF address bit by adding the internal base to the value of this field. If not used, write 31 (11111b) to this field. In this case, the HIF address bit to use is 0. If all of the following conditions apply, column bits 8, 7, and 6 must map to the highest three valid HIF address bits. Therefore, you must set this value to the highest valid HIF address bit minus the internal base minus two (highest HIF bit – 6 – 2): • Inline ECC configuration (MEMC_INLINE_ECC == 1) • ECC enabled (ECCCFG0[ecc_mode]>0) • Number of column address bits == 10 • Half bus width Programming Mode: Static

36.14.89 Address Map 4 (ADDRMAP4)

Offset

Register	Offset
ADDRMAP4	210h

Diagram



Fields

Field	Function
31 col_addr_shift	Column address shift 2 Shifts the column address 2 bits to the left (a lower HIF address) in specific cases as required by inline ECC configuration. **NOTE** If inline ECC is not enabled (ECCCFG0[ecc_mode] == 0), set to 0. Programming Mode: Static 0b - No shift 1b - Shift
30-13 —	Reserved
12-8 addrmap_col_b11	Column address bit 11 Depending on the bus width mode, selects the host interface (HIF) address bit to use as: • Full bus width: column address bit 13 (for LPDDR2 and LPDDR3: bit 11) • Half bus width: Not used (see instruction below for "If not used") • Quarter bus width: Not used (see instruction below for "If not used") The valid values are 0 to 7, 31, and x, where x is a valid value in the inline ECC configuration. The internal base is 11. The MC determines the selected HIF address bit by adding the internal base to the value of this field. If not used, write 31 (11111b) to this field. In this case, the HIF address bit to use is 0. If the following conditions apply, the highest three column address bits must map to the highest three valid HIF address bits: • Inline ECC configuration (MEMC_INLINE_ECC == 1) • ECC enabled (ECCCFG0[ecc_mode]>0) If column bit 11 is the highest column address bit, you must set this value to the highest valid HIF address bit minus the internal base (highest HIF bit – 11) If column bit 11 is the second highest column address bit, you must set this value to the second highest valid HIF address bit minus the internal base minus one (highest HIF bit – 11 – 1) If column bit 11 is the third highest column address bit, you must set this value to the highest valid HIF address bit minus the internal base minus two (highest HIF bit – 11 – 2) **NOTE** For DDR2, DDR3, and mDDR (per the JEDEC specifications), column address bit 10 is reserved for indicating auto-precharge, and therefore no source address bit can be mapped to column address bit 10. LPDDR2 and LPDDR3 have a dedicated bit in the CA bus for auto-precharge, so these DRAM types use column address bit 10. Programming Mode: Static

Table continues on the next page...

Table continued from the previous page...

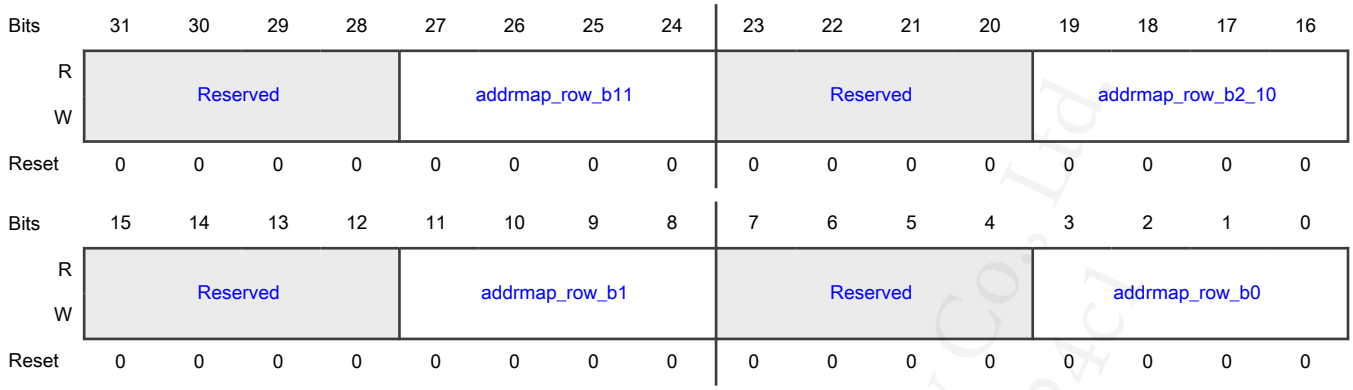
Field	Function
7-5 —	Reserved
4-0 addrmap_col_b10	Column address bit 10 Depending on the bus width mode, selects the host interface (HIF) address bit to use as: • Full bus width: column address bit 11 (for LPDDR2 and LPDDR3: bit 10) • Half bus width: column address bit 13 (for LPDDR2 and LPDDR3: bit 11) • Quarter bus width: Not used (see instruction below for "If not used") The valid values are 0 to 7, 31, and x, where x is a valid value in the inline ECC configuration. The internal base is 10. The MC determines the selected HIF address bit by adding the internal base to the value of this field. If not used, write 31 (11111b) to this field. In this case, the HIF address bit to use is 0. If the following conditions apply, the highest three column address bits must map to the highest three valid HIF address bits: • Inline ECC configuration (MEMC_INLINE_ECC == 1) • ECC enabled (ECCCFG0[ecc_mode]>0) If column bit 10 is the highest column address bit, you must set this value to the highest valid HIF address bit minus the internal base (highest HIF bit – 10) If column bit 10 is the second highest column address bit, you must set this value to the second highest valid HIF address bit minus the internal base minus one (highest HIF bit – 10 – 1) If column bit 10 is the third highest column address bit, you must set this value to the highest valid HIF address bit minus the internal base minus two (highest HIF bit – 10 – 2) **NOTE** For DDR2, DDR3, and mDDR (per the JEDEC specifications), column address bit 10 is reserved for indicating auto-precharge, and therefore no source address bit can be mapped to column address bit 10. LPDDR2 and LPDDR3 have a dedicated bit in the CA bus for auto-precharge, so these DRAM types use column address bit 10. Programming Mode: Static

36.14.90 Address Map 5 (ADDRMAP5)

Offset

Register	Offset
ADDRMAP5	214h

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 addrmap_row_b11	Row address bit 11 Selects the HIF address bits to use as row address bit 11. The valid values are 0 to 11, and 15. The internal base is 17. The MC determines the selected HIF address bit by adding the internal base to the value of this field. If not used, write 15 (1111b) to this field. In this case, the HIF address bit to use is 0. Programming Mode: Static
23-20 —	Reserved
19-16 addrmap_row_b2_10	Row address bits 2 to 10 Selects the HIF address bits to use as row address bits 2 to 10. The valid values are 0 to 11, and 15. The internal base is (address bit + 6). For example, the base for bit 2 is (2 + 6 = 8. For bit 10, the base is (10 + 6 = 16. The MC determines the selected HIF address bit by adding the internal base to the value of this field. When set to 15, the MC obtains the row address bits 2 to 10 from ADDRMAP9, ADDRMAP10, and ADDRMAP11. Programming Mode: Static
15-12 —	Reserved
11-8 addrmap_row_b1	Row address bit 1 Selects the HIF address bits to use as row address bit 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	The valid values are 0 to 11. The internal base is 7. The MC determines the selected HIF address bit by adding the internal base to the value of this field. Programming Mode: Static
7-4 —	Reserved
3-0 addrmap_row_b0	Row address bit 0 Selects the HIF address bits to use as row address bit 0. The valid values are 0 to 11. The internal base is 6. The MC determines the selected HIF address bit by adding the internal base to the value of this field. Programming Mode: Static

36.14.91 Address Map 6 (ADDRMAP6)

Offset

Register	Offset
ADDRMAP6	218h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	lpddr3_...	lpddr4_6gb_12gb...		Reserved	addrmap_row_b15				Reserved				addrmap_row_b14			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				addrmap_row_b13				Reserved				addrmap_row_b12			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 lpddr3_6gb_12gb	Indicates that a 6 GB or 12 GB LPDDR3 DRAM device is being used. This field applies only to LPDDR3 DRAM. Programming Mode: Static

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8a62e5-4821-8ab-b251-390d8ab624c1

Table continued from the previous page...

Field	Function
	0b - No LPDDR3 DRAM. All addresses are valid. 1b - 6 GB or 12 GB LPDDR3 DRAM is present. Every address with row[14:13] == 11b is invalid
30-29 lpddr4_6gb_12gb_24gb	LPDDR4 type Specifies the type of LPDDR4 DRAM being used. This field applies only to LPDDR4 DRAM. Programming Mode: Static 00b - No LPDDR4 DRAM. All addresses are valid. 01b - 6 GB LPDDR4 DRAM. Every address with row[14:13] == 11b is invalid 10b - 12 GB LPDDR4 DRAM. Every address with row[15:14] == 11b is invalid 11b - 24 GB LPDDR4 DRAM. Every address with row[16:15] == 11b is invalid
28 —	Reserved
27-24 addrmap_row_b15	Row address bit 15 Selects the HIF address bits to use as row address bit 15. The valid values are 0 to 11, and 15. The internal base is 21. The MC determines the selected HIF address bit by adding the internal base to the value of this field. If not used, write 15 (1111b) to this field. In this case, the HIF address bit to use is 0. Programming Mode: Static
23-20 —	Reserved
19-16 addrmap_row_b14	Row address bit 14 Selects the HIF address bits to use as row address bit 14. The valid values are 0 to 11, and 15. The internal base is 20. The MC determines the selected HIF address bit by adding the internal base to the value of this field. If not used, write 15 (1111b) to this field. In this case, the HIF address bit to use is 0. Programming Mode: Static
15-12 —	Reserved
11-8 addrmap_row_b13	Row address bit 13 Selects the HIF address bits to use as row address bit 13. The valid values are 0 to 11, and 15. The internal base is 19. The MC determines the selected HIF address bit by adding the internal base to the value of this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If not used, write 15 (1111b) to this field. In this case, the HIF address bit to use is 0. Programming Mode: Static
7-4 —	Reserved
3-0 addrmap_row_b12	Row address bit 12 Selects the HIF address bits to use as row address bit 12. The valid values are 0 to 11, and 15. The internal base is 18. The MC determines the selected HIF address bit by adding the internal base to the value of this field. If not used, write 15 (1111b) to this field. In this case, the HIF address bit to use is 0. Programming Mode: Static

36.14.92 Address Map 7 (ADDRMAP7)

Offset

Register	Offset
ADDRMAP7	21Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				addrmap_row_b17				Reserved				addrmap_row_b16			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEC Automotive Technology Co., Ltd.
8c462e5-4821-8a8-b251-390a3ab624c1

Table continued from the previous page...

Field	Function
11-8 addrmap_row_b17	Selects the HIF address bit used as row address bit 17. Valid Range: 0 to 11, and 15 Internal Base: 23 The selected HIF address bit is determined by adding the internal base to the value of this field. If unused, set to 15 and then row address bit 17 is set to 0. Programming Mode: Static
7-4 —	Reserved
3-0 addrmap_row_b16	Selects the HIF address bit used as row address bit 16. Valid Range: 0 to 11, and 15 Internal Base: 22 The selected HIF address bit is determined by adding the internal base to the value of this field. If unused, set to 15 and then row address bit 16 is set to 0. Programming Mode: Static

36.14.93 Address Map 8 (ADDRMAP8)

Offset

Register	Offset
ADDRMAP8	220h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		addrmap_bg_b1						Reserved		addrmap_bg_b0					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-14 —	Reserved
13-8 addrmap_bg_b1	Selects the HIF address bits used as bank group address bit 1. Valid Range: 0 to 32, and 63 Internal Base: 3 The selected HIF address bit for each of the bank group address bits is determined by adding the internal base to the value of this field. If unused, set to 63 and then bank group address bit 1 is set to 0. Programming Mode: Static

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-6 —	Reserved
5-0 addrmap_bg_b0	Selects the HIF address bits used as bank group address bit 0. Valid Range: 0 to 32, and 63 Internal Base: 2 The selected HIF address bit for each of the bank group address bits is determined by adding the internal base to the value of this field. If unused, set to 63 and then bank group address bit 0 is set to 0. Programming Mode: Static

36.14.94 Address Map 9 (ADDRMAP9)

Offset

Register	Offset
ADDRMAP9	224h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				addrmap_row_b5				Reserved				addrmap_row_b4			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				addrmap_row_b3				Reserved				addrmap_row_b2			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-24 addrmap_row_b5	Selects the HIF address bits used as row address bit 5. Valid Range: 0 to 11 Internal Base: 11 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15. Programming Mode: Static
23-20 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
19-16 addrmap_row_b4	Selects the HIF address bits used as row address bit 4. Valid Range: 0 to 11 Internal Base: 10 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15. Programming Mode: Static
15-12 —	Reserved
11-8 addrmap_row_b3	Selects the HIF address bits used as row address bit 3. Valid Range: 0 to 11 Internal Base: 9 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15. Programming Mode: Static
7-4 —	Reserved
3-0 addrmap_row_b2	Selects the HIF address bits used as row address bit 2. Valid Range: 0 to 11 Internal Base: 8 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15. Programming Mode: Static

36.14.95 Address Map 10 (ADDRMAP10)

Offset

Register	Offset
ADDRMAP10	228h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved				addrmap_row_b9				Reserved				addrmap_row_b8			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved				addrmap_row_b7				Reserved				addrmap_row_b6			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA of NXP
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-4821-48a8-b251-3b13ab0fc1c1

Fields

Field	Function
31-28 —	Reserved
27-24 addrmap_row_b9	Selects the HIF address bits used as row address bit 9. Valid Range: 0 to 11 Internal Base: 15 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15. Programming Mode: Static
23-20 —	Reserved
19-16 addrmap_row_b8	Selects the HIF address bits used as row address bit 8. Valid Range: 0 to 11 Internal Base: 14 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15. Programming Mode: Static
15-12 —	Reserved
11-8 addrmap_row_b7	Selects the HIF address bits used as row address bit 7. Valid Range: 0 to 11 Internal Base: 13 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15. Programming Mode: Static
7-4 —	Reserved
3-0 addrmap_row_b6	Selects the HIF address bits used as row address bit 6. Valid Range: 0 to 11 Internal Base: 12 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15. Programming Mode: Static

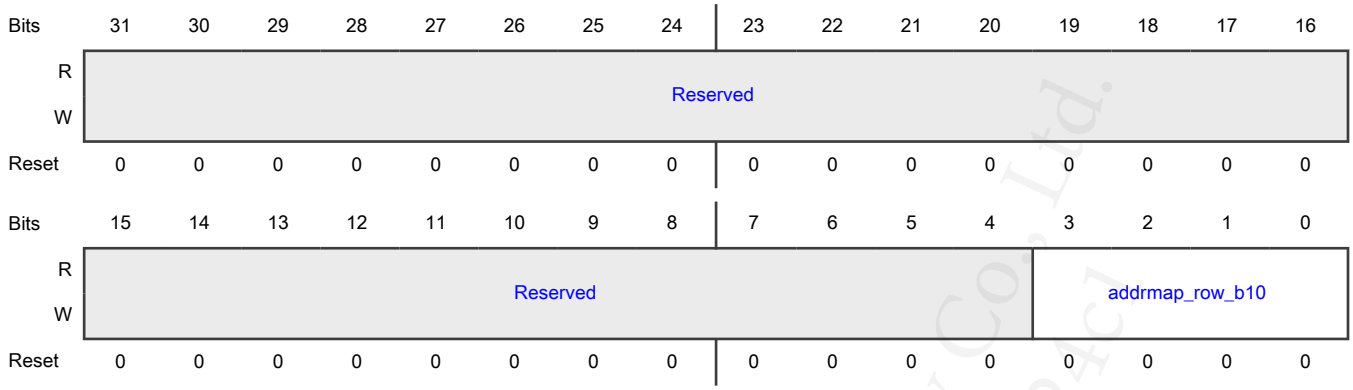
36.14.96 Address Map 11 (ADDRMAP11)

Offset

Register	Offset
ADDRMAP11	22Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48...b25190d3...624c2

Diagram



Fields

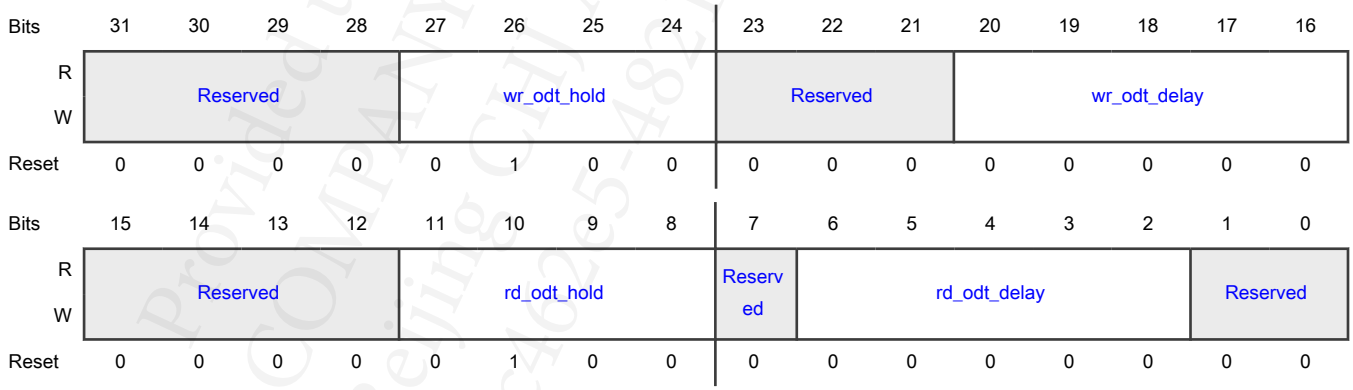
Field	Function
31-4 —	Reserved
3-0 addrmap_row_b10	Selects the HIF address bits used as row address bit 10. Valid Range: 0 to 11 Internal Base: 16 The selected HIF address bit for each of the row address bits is determined by adding the internal base to the value of this field. This register field is used only when ADDRMAP5.addrmap_row_b2_10 is set to value 15. Programming Mode: Static

36.14.97 ODT Configuration (ODTCFG)

Offset

Register	Offset
ODTCFG	240h

Diagram



Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c16254-821-48a8-b251-90d3ab624c1

Fields

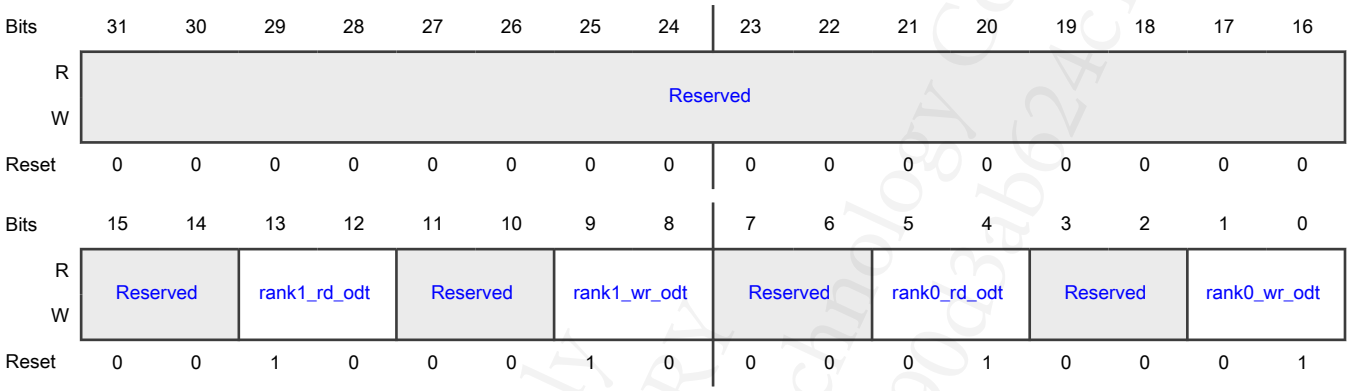
Field	Function
31-28 —	Reserved
27-24 wr_odt_hold	DFI PHY clock cycles to hold ODT for a write command. The minimum supported value is 2. Recommended values: DDR2: - BL8: 0x5 (DDR2-400/533/667), 0x6 (DDR2-800), 0x7 (DDR2-1066) - BL4: 0x3 (DDR2-400/533/667), 0x4 (DDR2-800), 0x5 (DDR2-1066) DDR3: - BL8: 0x6 DDR4: - BL8: 5 + WR_PREAMBLE + CRC_MODE WR_PREAMBLE = 1 (1tCK write preamble), 2 (2tCK write preamble) CRC_MODE = 0 (not CRC mode), 1 (CRC mode) LPDDR3: - BL8: 7 + RU(tODTon(max)/tCK) Unit: DFI PHY clock cycles. Programming Mode: Quasi-dynamic Group 1, Group 4
23-21 —	Reserved
20-16 wr_odt_delay	The delay, in DFI PHY clock cycles, from issuing a write command to setting ODT values associated with that command. ODT setting must remain constant for the entire time that DQS is driven by the uMCTL2. Recommended values: DDR2: - CWL + AL - 3 (DDR2-400/533/667), CWL + AL - 4 (DDR2-800), CWL + AL - 5 (DDR2-1066) If (CWL + AL - 3 < 0), uMCTL2 does not support ODT for write operation. DDR3: - 0x0 DDR4: - DFITMG1.dfi_t_cmd_lat (to adjust for CAL mode) LPDDR3: - WL - 1 - RU(tODTon(max)/tCK)) Unit: DFI PHY clock cycles. Programming Mode: Quasi-dynamic Group 1, Group 4
15-12 —	Reserved
11-8 rd_odt_hold	DFI PHY clock cycles to hold ODT for a read command. The minimum supported value is 2. Recommended values: DDR2: - BL8: 0x6 (not DDR2-1066), 0x7 (DDR2-1066) - BL4: 0x4 (not DDR2-1066), 0x5 (DDR2-1066) DDR3: - BL8 - 0x6 DDR4: - BL8: 5 + RD_PREAMBLE RD_PREAMBLE = 1 (1tCK read preamble), 2 (2tCK read preamble) LPDDR3: - BL8: 5 + RU(tDQSCK(max)/tCK) - RD(tDQSCK(min)/tCK) + RU(tODTon(max)/tCK) Unit: DFI PHY clock cycles. Programming Mode: Quasi-dynamic Group 1, Group 4
7 —	Reserved
6-2 rd_odt_delay	The delay, in DFI PHY clock cycles, from issuing a read command to setting ODT values associated with that command. ODT setting must remain constant for the entire time that DQS is driven by the uMCTL2. Recommended values: DDR2: - CL + AL - 4 (not DDR2-1066), CL + AL - 5 (DDR2-1066) If (CL + AL - 4 < 0), uMCTL2 does not support ODT for read operation. DDR3: - CL - CWL DDR4: - CL - CWL - RD_PREAMBLE + WR_PREAMBLE + DFITMG1.dfi_t_cmd_lat (to adjust for CAL mode) WR_PREAMBLE = 1 (1tCK write preamble), 2 (2tCK write preamble) RD_PREAMBLE = 1 (1tCK read preamble), 2 (2tCK read preamble) If (CL - CWL - RD_PREAMBLE + WR_PREAMBLE) < 0, uMCTL2 does not support ODT for read operation. LPDDR3: - RL + RD(tDQSCK(min)/tCK) - 1 - RU(tODTon(max)/tCK) Unit: DFI PHY clock cycles. Programming Mode: Quasi-dynamic Group 1, Group 4
1-0 —	Reserved

36.14.98 ODT/Rank Map (ODTMAP)

Offset

Register	Offset
ODTMAP	244h

Diagram



Fields

Field	Function
31-14 —	Reserved
13-12 rank1_rd_odt	ODTs for read from rank 1 Turns on remote on-die terminations (ODTs) in the DRAM for a read from rank 1. Each rank has a remote ODT in the DRAM. Set a bit to 1 to turn on the ODT for the corresponding rank during a read from rank 1. Programming Mode: Static 00b - No ODTs during read from rank 1 01b - Rank 0 ODT on during read from rank 1 10b - Rank 1 ODT on during read from rank 1 11b - Both rank 0 and 1 ODTs on during read from rank 1
11-10 —	Reserved
9-8 rank1_wr_odt	ODTs for write to rank 1 Turns on remote on-die terminations (ODTs) in the DRAM for a write to rank 1. Each rank has a remote ODT in the DRAM. Set a bit to 1 to turn on the ODT for the corresponding rank during a write to rank 1. Programming Mode: Static

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - No ODTs during write to rank 1 01b - Rank 0 ODT on during write to rank 1 10b - Rank 1 ODT on during write to rank 1 11b - Both rank 0 and 1 ODTs on during write to rank 1
7-6 —	Reserved
5-4 rank0_rd_odt	ODTs for read from rank 0 Turns on remote on-die terminations (ODTs) in the DRAM for a read from rank 0. Each rank has a remote ODT in the DRAM. Set a bit to 1 to turn on the ODT for the corresponding rank during a read from rank 0. Programming Mode: Static 00b - No ODTs during read from rank 0 01b - Rank 0 ODT on during read from rank 0 10b - Rank 1 ODT on during read from rank 0 11b - Both rank 0 and 1 ODTs on during read from rank 0
3-2 —	Reserved
1-0 rank0_wr_odt	ODTs for write to rank 0 Turns on remote on-die terminations (ODTs) in the DRAM for a write to rank 0. Each rank has a remote ODT in the DRAM. Set a bit to 1 to turn on the ODT for the corresponding rank during a write to rank 0. Programming Mode: Static 00b - No ODTs during write to rank 0 01b - Rank 0 ODT on during write to rank 0 10b - Rank 1 ODT on during write to rank 0 11b - Both rank 0 and 1 ODTs on during write to rank 0

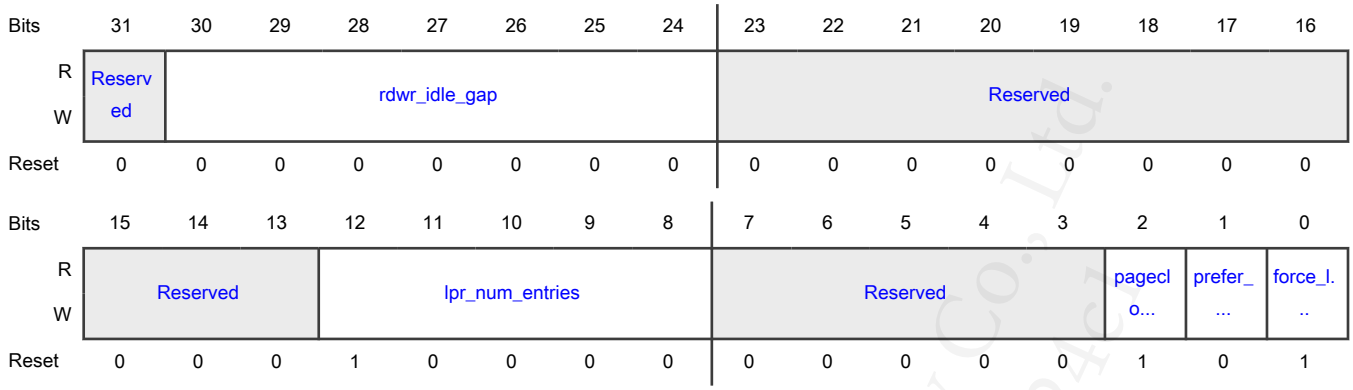
36.14.99 Scheduler Control (SCHED)

Offset

Register	Offset
SCHED	250h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3a624c1

Diagram



Fields

Field	Function
31 —	Reserved
30-24 rdwr_idle_gap	Specifies the time interval, in DFI clock cycles, for the preferred transaction store to be empty before switching to the alternate transaction store, if the alternate store is not empty. The default preferred transaction store is the read (both low- and high-priority) transaction store. The alternative transaction store is the write transaction store. If prefer_write is set to 1, then the preferred and alternative stores are reversed: write is preferred and read is alternate. When this field is set to 0x00, transaction store switching happens immediately when the switching conditions become true. **NOTE** Affects performance only. Programming Mode: Static
23-13 —	Reserved
12-8 lpr_num_entries	Low-priority transaction store entries Specifies the number of entries in the low-priority transaction store minus 1. The number of entries available for the high-priority transaction store is (MEMC_NO_OF_ENTRY – (lpr_num_entries + 1)). Set to the maximum value to allocate all entries to the low-priority transaction store. Setting to 0 allocates one entry to the low-priority transaction store and the rest to the high-priority transaction store.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE In ECC configurations, the numbers of Write credits and Low Priority Read credits issued are both one less than in non-ECC configurations. The MC reserves one entry each in the Write and Low Priority Read CAMS to store RMW requests resulting from single-bit error correction operation. Programming Mode: Static
7-3 —	Reserved
2 pageclose	Paging policy Specifies the paging policy for precharge operations. See Intelligent precharges for a complete explanation of the intertaction between pageclose and SCHED1[pageclose_timer]. **NOTE** Affects performance only. Programming Mode: Quasi-dynamic Group 3
1 prefer_write	Prefer writes over reads Tells the bank selector to prefer writes over reads. **NOTE** For debug only. Programming Mode: Static 0b - No preference 1b - Prefer writes over reads
0 force_low_pri_n	Force low priority Forces all incoming transactions to low priority. The MC treats all High Priorority Read (HPR) and Variable Priority Read (VPR) commands as Low Priority Read (LPR) commands, and all Variable Priority Write (VPW) commands as Normal Priority Write (NPW) commands. This is an active low signal. Forcing incoming transactions to low priority implicitly turns off the Bypass path for read commands. Affects performance only. Programming Mode: Static 0b - Force low priority 1b - Do not force low priority

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c40e5-4837-48a8-9251-390d3b624c1

36.14.100 Scheduler Control 1 (SCHED1)

Offset

Register	Offset
SCHED1	254h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								pageclose_timer							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

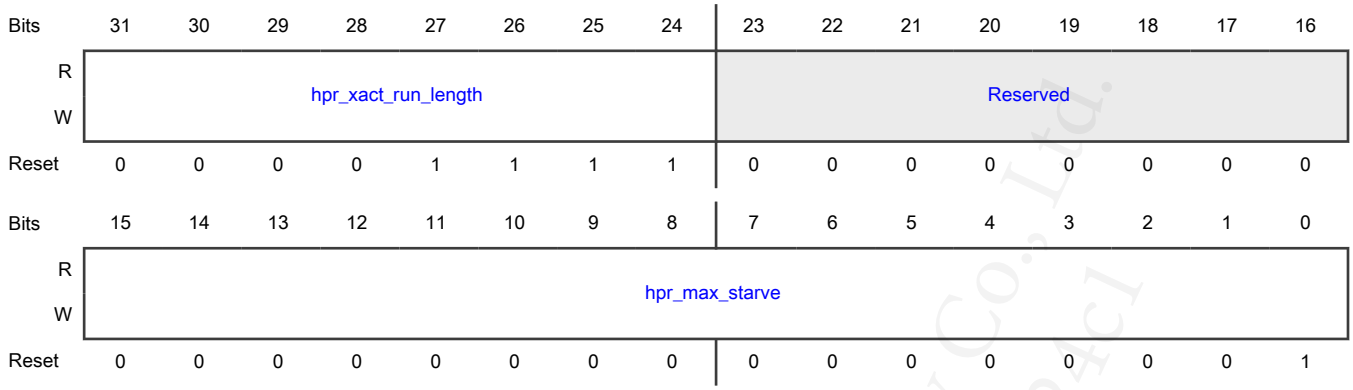
Field	Function
31-8 —	Reserved
7-0 pageclose_timer	Specifies the time interval, in DFI clock cycles, that a bank is kept open while there are page hit transactions in the CAM for that bank. See Intelligent precharges for a complete explanation of the intertaction between pageclose_timer and SCHED[pageclose]. **NOTE** Applies only if SCHED[pageclose] == 1. Programming Mode: Static

36.14.101 High Priority Read CAM 1 (PERFHPR1)

Offset

Register	Offset
PERFHPR1	25Ch

Diagram



Fields

Field	Function
31-24 hpr_xact_run_length	HPR transactions after critical Specifies the number of transactions serviced after the HPR queue goes critical, as long as transactions are available. In other words, the number of transactions serviced after the HPR queue goes critical is the smaller of: • This value • The number of available transactions **NOTE** Affects performance only. Programming Mode: Quasi-dynamic Group 3
23-16 —	Reserved
15-0 hpr_max_starve	HPR maximimum starve interval Specifies the time interval, in DFI clock cycles, that the High Priority Read (HPR) queue can be starved before it goes critical. The minimum practical value for normal operation is 1h; a value of 0h disables the starvation function and results in excessive latencies. **NOTE** Affects performance only. Programming Mode: Quasi-dynamic Group 3

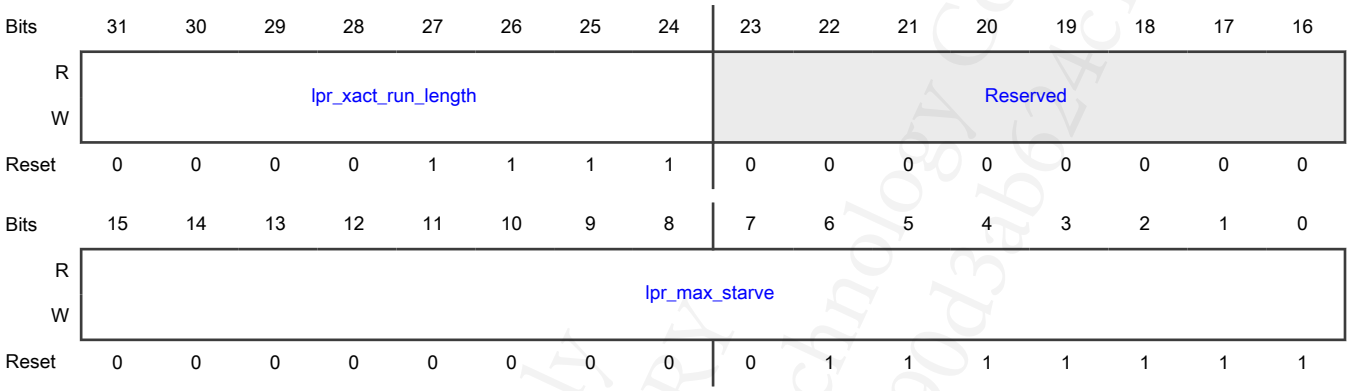
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c46205-4891-48a8-b251-39d3ab624c1

36.14.102 Low Priority Read CAM 1 (PERFLPR1)

Offset

Register	Offset
PERFLPR1	264h

Diagram



Fields

Field	Function
31-24 lpr_xact_run_length	LPR transactions after critical Specifies the number of transactions serviced after the LPR queue goes critical, as long as transactions are available. In other words, the number of transactions serviced after the LPR queue goes critical is the smaller of: • This value • The number of available transactions **NOTE** Affects performance only. Programming Mode: Quasi-dynamic Group 3
23-16 —	Reserved
15-0 lpr_max_starve	LPR maximimum starve interval Specifies the time interval, in DFI clock cycles, that the Low Priority Read (LPR) queue can be starved before it goes critical. The minimum practical value for normal operation is 1h; a value of 0h disables the starvation function and results in excessive latencies.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Affects performance only. Programming Mode: Quasi-dynamic Group 3

36.14.103 Write CAM 1 (PERFWR1)

Offset

Register	Offset
PERFWR1	26Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	w_xact_run_length								Reserved							
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	w_max_starve															
Reset	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1

Fields

Field	Function
31-24 w_xact_run_length	WR transactions after critical Specifies the number of transactions serviced after the WR queue goes critical, as long as transactions are available. In other words, the number of transactions serviced after the LPR queue goes critical is the smaller of: • This value • The number of available transactions **NOTE** Affects performance only. Programming Mode: Quasi-dynamic Group 3
23-16	Reserved

Table continues on the next page...

Table continued from the previous page...

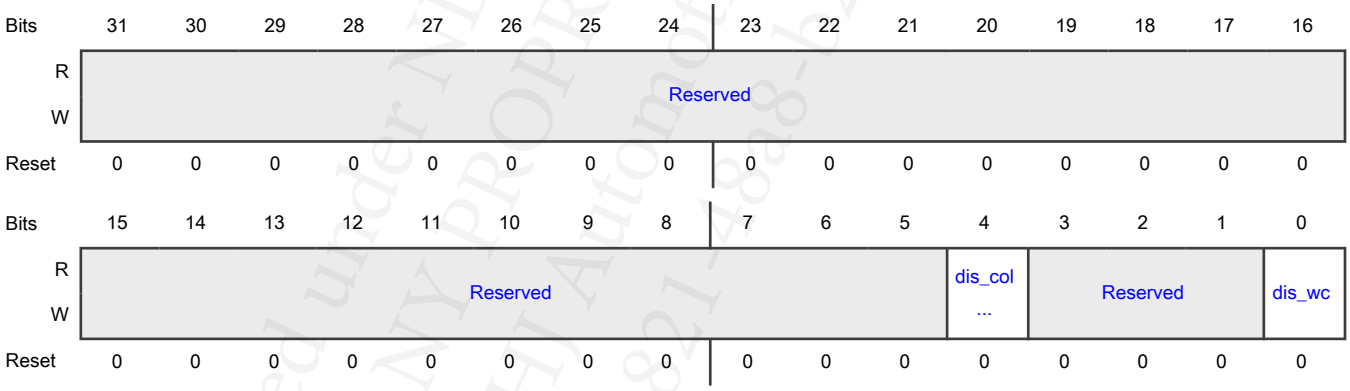
Field	Function
—	
15-0 w_max_starve	WR maximimum starve interval Specifies the time interval, in DFI clock cycles, that the Write (WR) queue can be starved before it goes critical. The minimum practical value for normal operation is 1h; a value of 0h disables the starvation function and results in excessive latencies. **NOTE** Affects performance only. Programming Mode: Quasi-dynamic Group 3

36.14.104 Debug 0 (DBG0)

Offset

Register	Offset
DBG0	300h

Diagram



Fields

Field	Function
31-5 —	Reserved
4	When this is set to '0', auto-precharge is disabled for the flushed command in a collision case. Collision cases are write followed by read to same address, read followed by write to same address, or write

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
ce462e5-4821-48f8-b251-390d3ab624d1

Table continued from the previous page...

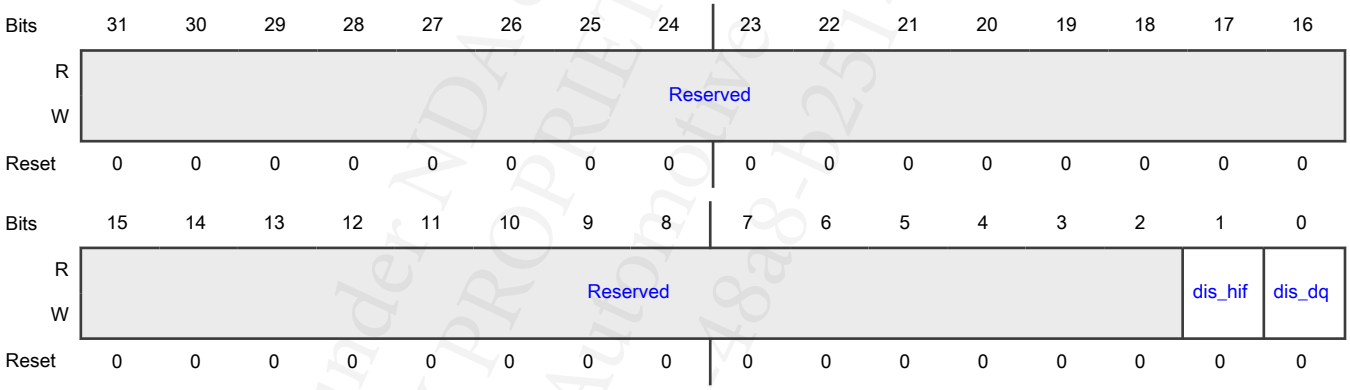
Field	Function
dis_collision_page_opt	followed by write to same address with DBG0.dis_wc bit = 1 (where same address comparisons exclude the two address bits representing critical word). FOR DEBUG ONLY. Programming Mode: Static
3-1 —	Reserved
0 dis_wc	When 1, disable write combine. FOR DEBUG ONLY Programming Mode: Static

36.14.105 Debug 1 (DBG1)

Offset

Register	Offset
DBG1	304h

Diagram



Fields

Field	Function
31-2 —	Reserved
1 dis_hif	When 1, uMCTL2 asserts the HIF command signal hif_cmd_stall. uMCTL2 will ignore the hif_cmd_valid and all other associated request signals. This bit is intended to be switched on-the-fly. Programming Mode: Dynamic
0 dis_dq	When 1, uMCTL2 will not de-queue any transactions from the CAM. Bypass is also disabled. All transactions are queued in the CAM. No reads or writes are issued to DRAM as long as this is asserted. This bit may be used to prevent reads or writes being issued by the uMCTL2, which

Table continues on the next page...

Table continued from the previous page...

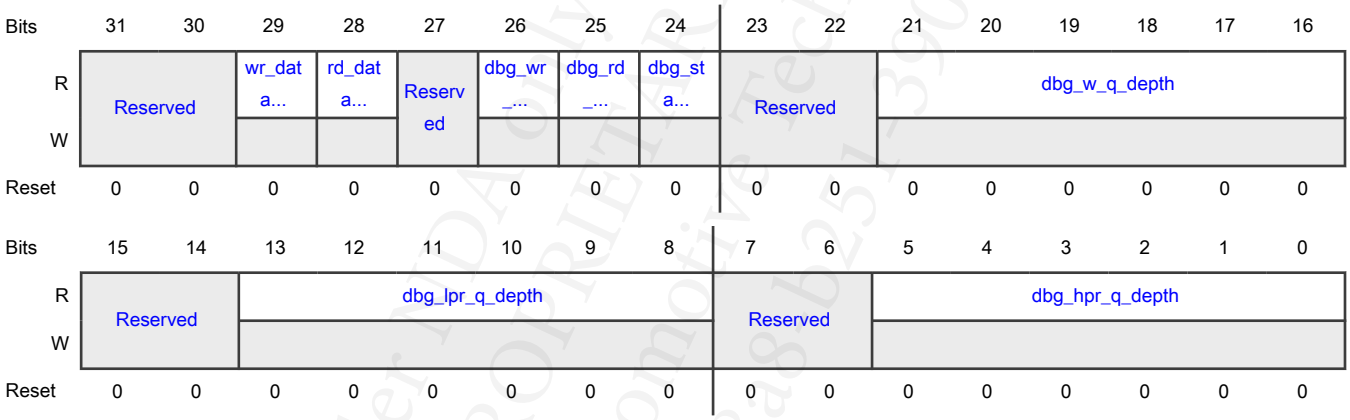
Field	Function
	makes it safe to modify certain register fields associated with reads and writes (see User Guide for details). After setting this bit, it is strongly recommended to poll DBGCAM.wr_data_pipeline_empty and DBGCAM.rd_data_pipeline_empty, before making changes to any registers which affect reads and writes. This will ensure that the relevant logic in the MC is idle. This bit is intended to be switched on-the-fly. Programming Mode: Dynamic

36.14.106 CAM Debug (DBGCAM)

Offset

Register	Offset
DBGCAM	308h

Diagram



Fields

Field	Function
31-30 —	Reserved
29 wr_data_pipeline_empty	This bit indicates that the write data pipeline on the DFI interface is empty. This register is intended to be polled at least twice after setting DBG1.dis_dq, to ensure that all remaining commands/data have completed. Programming Mode: Dynamic
28 rd_data_pipeline_empty	This bit indicates that the read data pipeline on the DFI interface is empty. This register is intended to be polled at least twice after setting DBG1.dis_dq, to ensure that all remaining commands/data have completed. Programming Mode: Dynamic
27	Reserved

Table continues on the next page...

Table continued from the previous page...

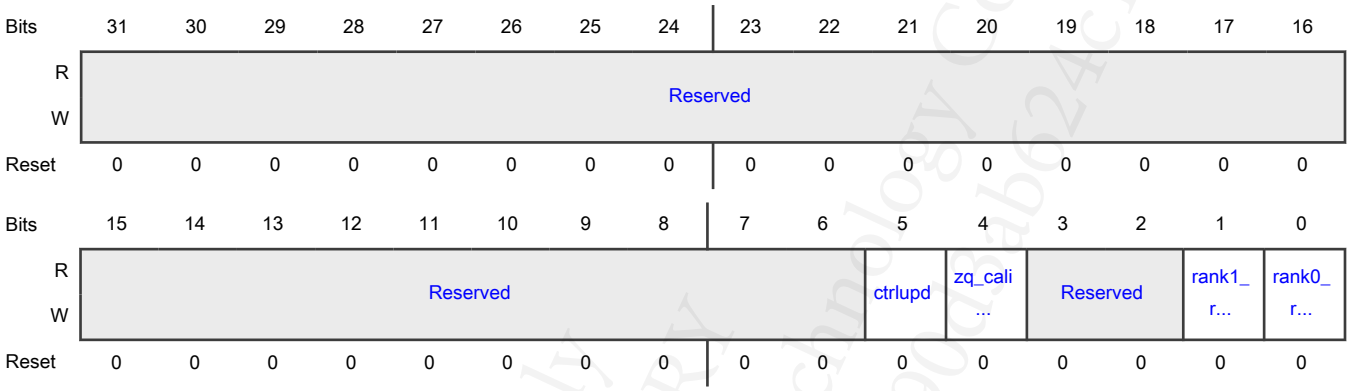
Field	Function
—	
26 dbg_wr_q_empty	This field is used for debug purposes. When its value is 1, all the write command queues and write data buffers inside the memory controller are empty. With the field's value as 1, if the controller enters Self-Refresh mode using the low-power entry sequence, it is expected to have executed all the commands in its queues, with the write and read data drained. However, you can write 1 both to this field and the DBGCAM.[dbg_rd_q_empty] field only when there are no reads and writes in the DDR controller. Therefore, you need to monitor both these fields simultaneously to get the status of the pending transactions in uMCTL2.
25 dbg_rd_q_empty	This field is used for debug purposes. When its value is 1, all the read command queues and read data buffers inside the memory controller are empty. With the field's value as 1, if the controller enters Self-Refresh mode using the low-power entry sequence, it is expected to have executed all the commands in its queues, with the write and read data drained. However, you can write 1 both to this field and the DBGCAM.[dbg_wr_q_empty] field only when there are no reads and writes in the DDR controller. Therefore, you need to monitor both these fields simultaneously to get the status of the pending transactions in uMCTL2.
24 dbg_stall	Stall FOR DEBUG ONLY Programming Mode: Dynamic
23-22 —	Reserved
21-16 dbg_w_q_depth	Write queue depth The last entry of WR queue is reserved for ECC SCRUB operation. This entry is not included in the calculation of the queue depth. FOR DEBUG ONLY Programming Mode: Dynamic
15-14 —	Reserved
13-8 dbg_lpr_q_depth	Low priority read queue depth The last entry of Lpr queue is reserved for ECC SCRUB operation. This entry is not included in the calculation of the queue depth. FOR DEBUG ONLY Programming Mode: Dynamic
7-6 —	Reserved
5-0 dbg_hpr_q_depth	High priority read queue depth FOR DEBUG ONLY Programming Mode: Dynamic

36.14.107 Command Debug (DBGCMD)

Offset

Register	Offset
DBGCMD	30Ch

Diagram



Fields

Field	Function
31-6 —	Reserved
5 ctrlupd	Setting this register bit to 1 indicates to the uMCTL2 to issue a dfi_ctrlupd_req to the PHY. When this request is stored in the uMCTL2, the bit is automatically cleared. This operation must only be performed when DFIUPD0.dis_auto_ctrlupd=1. Programming Mode: Dynamic
4 zq_calib_short	Setting this register bit to 1 indicates to the uMCTL2 to issue a ZQCS (ZQ calibration short)/MPC(ZQ calibration) command to the DRAM. When this request is stored in the uMCTL2, the bit is automatically cleared. This operation can be performed only when ZQCTL0.dis_auto_zq=1. It is recommended NOT to set this register bit if in Init, in Self-Refresh(except LPDDR4) or SR-Powerdown(LPDDR4) or Deep power-down operating modes or Maximum Power Saving Mode. For Self-Refresh(except LPDDR4) or SR-Powerdown(LPDDR4) it will be scheduled after SR(except LPDDR4) or SPRD(LPDDR4) has been exited. For Deep power down and Maximum Power Saving Mode, it will not be scheduled, although DBGSTAT.zq_calib_short_busy will be de-asserted. Programming Mode: Dynamic
3-2 —	Reserved
1 rank1_refresh	Setting this register bit to 1 indicates to the uMCTL2 to issue a refresh to rank 1. Writing to this bit causes DBGSTAT.rank1_refresh_busy to be set. When DBGSTAT.rank1_refresh_busy is cleared, the command has been stored in uMCTL2. For 3DS configuration, refresh is sent to rank index 1. This operation can be performed only when RFSHCTL3.dis_auto_refresh=1. It is recommended NOT to set this register bit if in Init or Deep power-down operating modes or Maximum Power Saving Mode. Programming Mode: Dynamic

Table continues on the next page...

Table continued from the previous page...

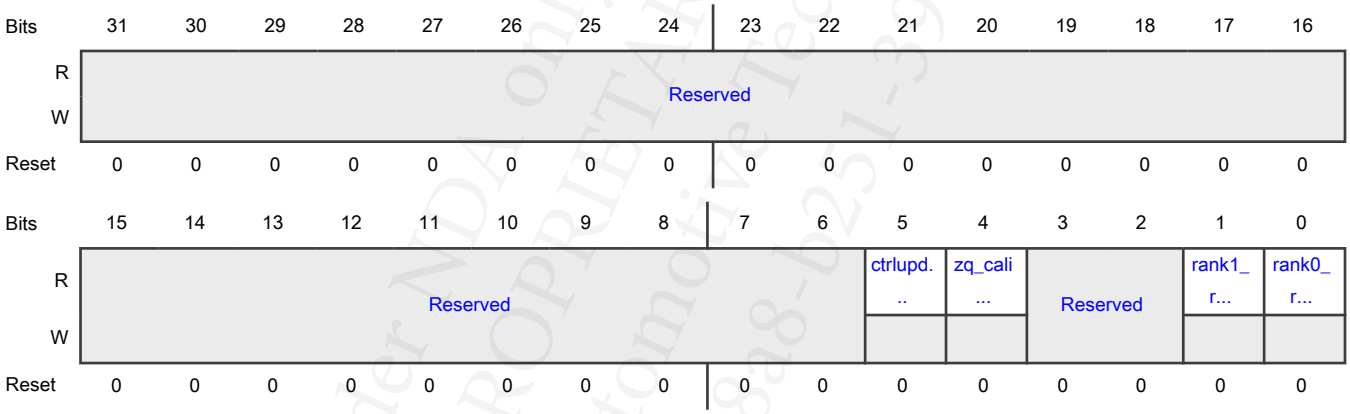
Field	Function
0 rank0_refresh	Setting this register bit to 1 indicates to the uMCTL2 to issue a refresh to rank 0. Writing to this bit causes DBGSTAT.rank0_refresh_busy to be set. When DBGSTAT.rank0_refresh_busy is cleared, the command has been stored in uMCTL2. For 3DS configuration, refresh is sent to rank index 0. This operation can be performed only when RFSHCTL3.dis_auto_refresh=1. It is recommended NOT to set this register bit if in Init or Deep power-down operating modes or Maximum Power Saving Mode. Programming Mode: Dynamic

36.14.108 Status Debug (DBGSTAT)

Offset

Register	Offset
DBGSTAT	310h

Diagram



Fields

Field	Function
31-6 —	Reserved
5 ctrlupd_busy	SoC core may initiate a ctrlupd operation only if this signal is low. This signal goes high in the clock after the uMCTL2 accepts the ctrlupd request. It goes low when the ctrlupd operation is initiated in the uMCTL2. It is recommended not to perform ctrlupd operations when this signal is high. - 0 - Indicates that the SoC core can initiate a ctrlupd operation - 1 - Indicates that ctrlupd operation has not been initiated yet in the uMCTL2 Programming Mode: Dynamic
4 zq_calib_short_busy	SoC core may initiate a ZQCS (ZQ calibration short) operation only if this signal is low. This signal goes high in the clock after the uMCTL2 accepts the ZQCS request. It goes low when the ZQCS operation is initiated in the uMCTL2. It is recommended not to perform ZQCS operations when this signal is high. - 0

Table continues on the next page...

Table continued from the previous page...

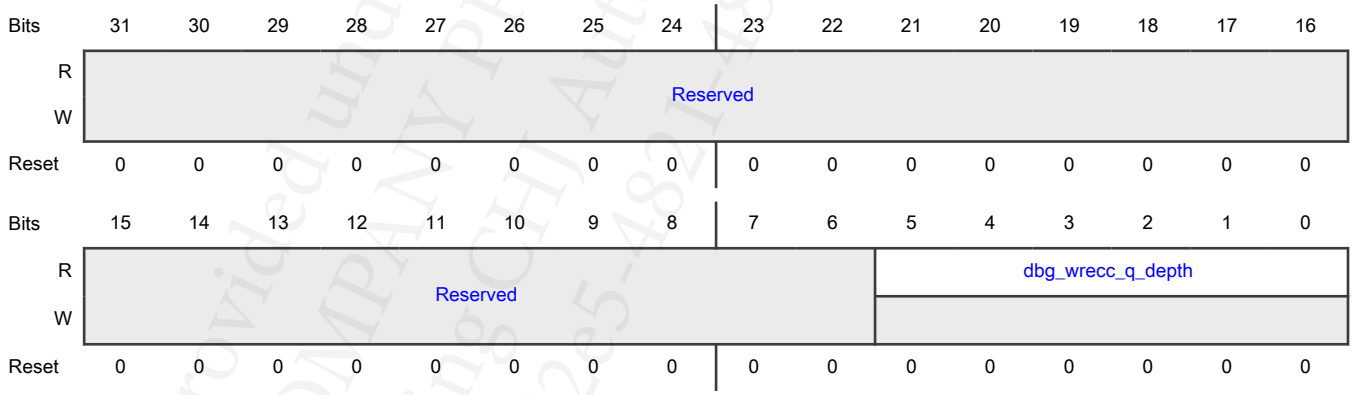
Field	Function
	- Indicates that the SoC core can initiate a ZQCS operation - 1 - Indicates that ZQCS operation has not been initiated yet in the uMCTL2 Programming Mode: Dynamic
3-2 —	Reserved
1 rank1_refresh_busy	SoC core may initiate a rank1_refresh operation (refresh operation to rank 1) only if this signal is low. This signal goes high in the clock after DBGCMD.rank1_refresh is set to one. It goes low when the rank1_refresh operation is stored in the uMCTL2. It is recommended not to perform rank1_refresh operations when this signal is high. - 0 - Indicates that the SoC core can initiate a rank1_refresh operation - 1 - Indicates that rank1_refresh operation has not been stored yet in the uMCTL2 Programming Mode: Dynamic
0 rank0_refresh_busy	SoC core may initiate a rank0_refresh operation (refresh operation to rank 0) only if this signal is low. This signal goes high in the clock after DBGCMD.rank0_refresh is set to one. It goes low when the rank0_refresh operation is stored in the uMCTL2. It is recommended not to perform rank0_refresh operations when this signal is high. - 0 - Indicates that the SoC core can initiate a rank0_refresh operation - 1 - Indicates that rank0_refresh operation has not been stored yet in the uMCTL2 Programming Mode: Dynamic

36.14.109 CAM Debug 1 (DBGCAM1)

Offset

Register	Offset
DBGCAM1	318h

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-3901...

Fields

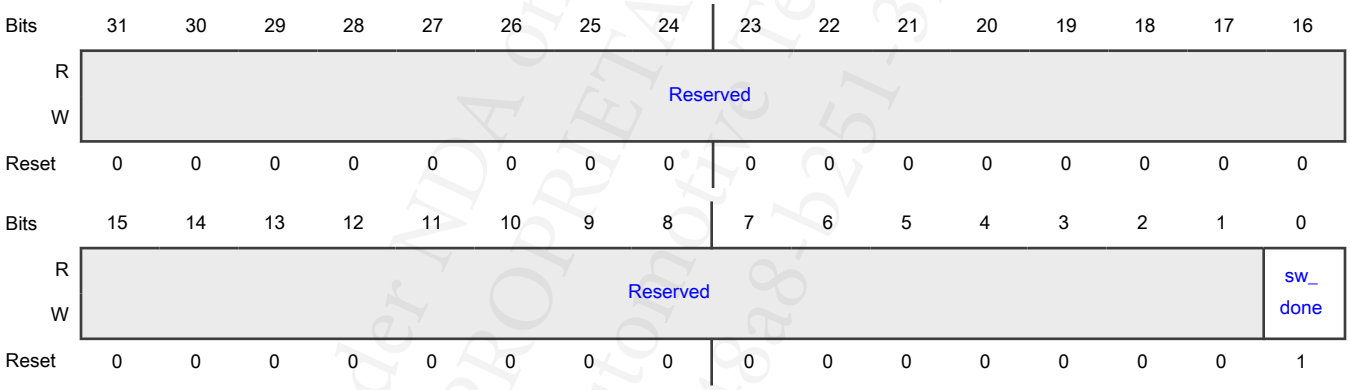
Field	Function
31-6 —	Reserved
5-0 dbg_wrecc_q_depth	Write ECC queue depth FOR DEBUG ONLY Programming Mode: Dynamic

36.14.110 Software Register Programming Control Enable (SWCTL)

Offset

Register	Offset
SWCTL	320h

Diagram



Fields

Field	Function
31-1 —	Reserved
0 sw_done	Enable quasi-dynamic register programming outside reset. Program register to 0 to enable quasi-dynamic programming. Set back register to 1 once programming is done. Programming Mode: Dynamic

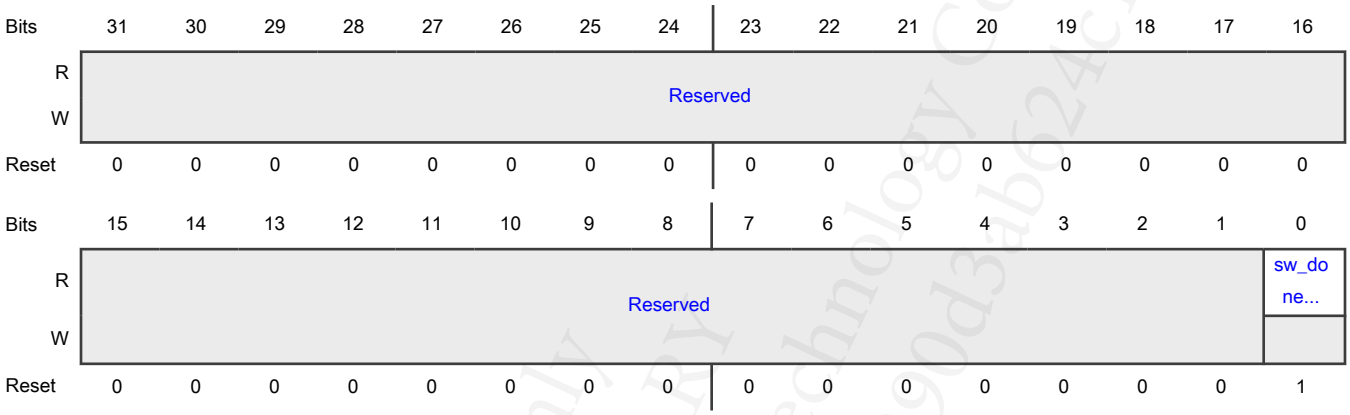
Printed under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462f5-4821-43a8-b251-390d3ab624c1

36.14.111 Software Register Programming Control Status (SWSTAT)

Offset

Register	Offset
SWSTAT	324h

Diagram



Fields

Field	Function
31-1 —	Reserved
0 sw_done_ack	Register programming done. This register is the echo of SWCTL.sw_done. Wait for sw_done value 1 to propagate to sw_done_ack at the end of the programming sequence to ensure that the correct registers values are propagated to the destination clock domains. Programming Mode: Static

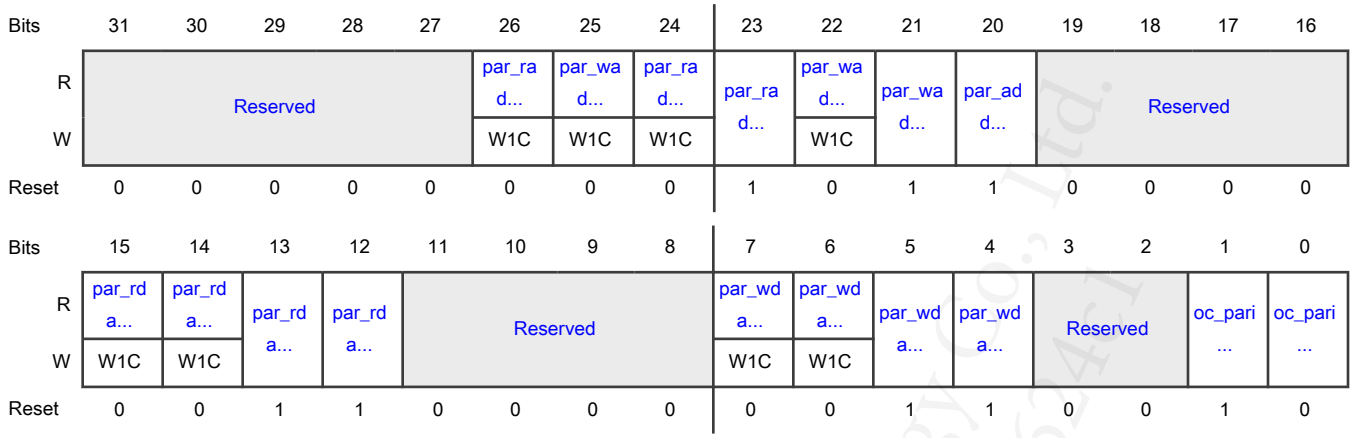
36.14.112 On-Chip Parity Configuration 0 (OCPARCFG0)

Offset

Register	Offset
OCPARCFG0	330h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4c0-b251-390d34b624c1

Diagram



Fields

Field	Function
31-27 —	Reserved
26 par_raddr_err_intr_force	Interrupt force bit for all par_raddr_err_intr. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
25 par_waddr_err_intr_force	Interrupt force bit for all par_waddr_err_intr. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
24 par_raddr_err_intr_clr	Interrupt clear bit for all par_raddr_err_intr. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
23 par_raddr_err_intr_en	Enables interrupt generation, if set to 1, for all ports, on signal par_raddr_err_intr_n upon detection of parity error on the AXI interface. Programming Mode: Dynamic
22 par_waddr_err_intr_clr	Interrupt clear bit for all par_waddr_err_intr. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
21 par_waddr_err_intr_en	Enables interrupt generation, if set to 1, for all ports, on signal par_waddr_err_intr_n upon detection of parity error on the AXI interface. Programming Mode: Dynamic
20	Enables SLVERR generation on read response or write response when address parity error is detected at the AXI interface. Programming Mode: Quasi-dynamic Group 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
par_addr_slverr_en	
19-16 —	Reserved
15 par_rdata_err_intr_force	Interrupt force bit for all par_rdata_err_intr_n and par_rdata_in_err_ecc_intr (Inline-ECC only). uMCTL2 automatically clears this bit. Programming Mode: Dynamic
14 par_rdata_err_intr_clr	Interrupt clear bit for par_rdata_err_intr. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
13 par_rdata_err_intr_en	Enables interrupt generation, if set to 1, for all ports, on signal par_rdata_err_intr_n upon detection of parity error at the AXI interface. Programming Mode: Dynamic
12 par_rdata_slverr_en	Enables SLVERR generation on read response when read data parity error is detected at the AXI interface. Programming Mode: Quasi-dynamic Group 3
11-8 —	Reserved
7 par_wdata_err_intr_force	Interrupt force bit for par_wdata_err_intr. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
6 par_wdata_err_intr_clr	Interrupt clear bit for par_wdata_err_intr. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
5 par_wdata_slverr_en	Enables SLVERR generation on write response when write data parity error is detected at the AXI interface. Programming Mode: Quasi-dynamic Group 3
4 par_wdata_err_intr_en	Enables write data interrupt generation (par_wdata_err_intr) upon detection of parity error at the AXI or DFI interface. Programming Mode: Dynamic
3-2 —	Reserved

Table continues on the next page...

Table continued from the previous page...

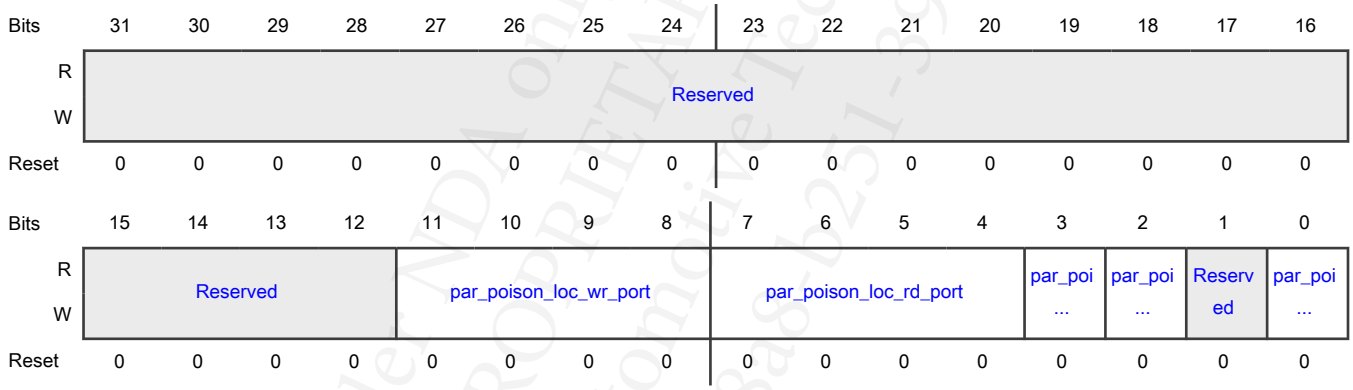
Field	Function
1 oc_parity_type	Parity type: - 0: Even parity - 1: Odd parity Programming Mode: Quasi-dynamic Group 3
0 oc_parity_en	Parity enable register. Enables On-Chip parity for all interfaces. Programming Mode: Quasi-dynamic Group 3

36.14.113 On-Chip Parity Configuration 1 (OCPARCFG1)

Offset

Register	Offset
OCPARCFG1	334h

Diagram



Fields

Field	Function
31-12 —	Reserved
11-8 par_poison_loc_wr_port	Enables parity poisoning on write data at the AXI interface before the input parity check logic. The value specifies the binary encoded port number of the AXI interface to be injected with parity error. Programming Mode: Quasi-dynamic Group 3
7-4 par_poison_loc_rd_port	Enables parity poisoning on read data at the AXI interface after the parity check logic. The value specifies the binary encoded port number of the AXI interface to be injected with parity error. Error can be injected to one port at a time. An error injected here is not logged and does not trigger SLVERR or interrupt by the controller. Programming Mode: Quasi-dynamic Group 3

Table continues on the next page...

Table continued from the previous page...

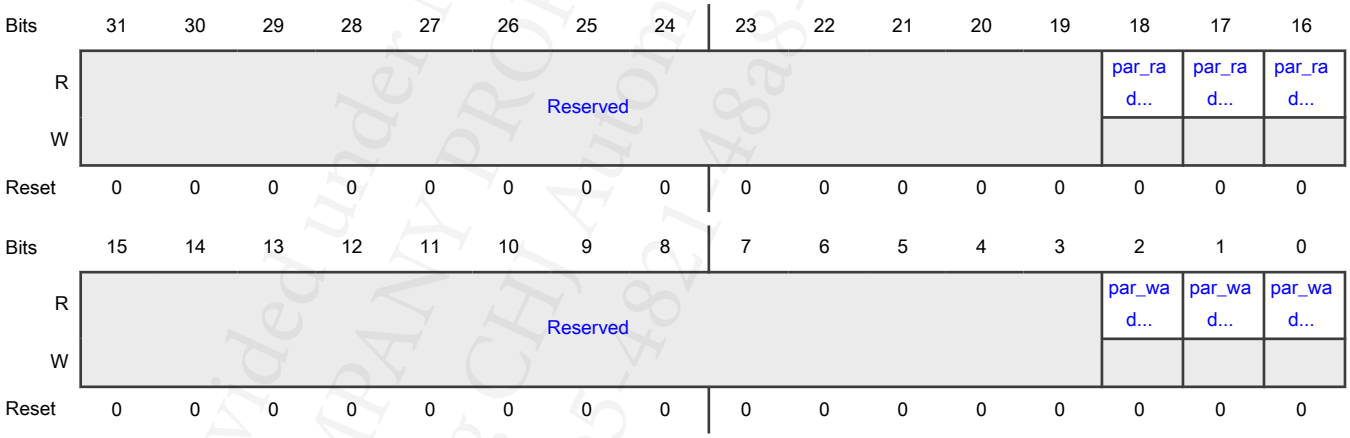
Field	Function
3 par_poison_loc_rd_iecc_type	Selects which parity to poison at the DFI when inline ECC is enabled. If this register is set to 0, parity error is injected on the first read data going through the ECC path; if this register is set to 1, parity error is injected on the first read data going through the data path. Programming Mode: Quasi-dynamic Group 3
2 par_poison_loc_rd_dfi	Enables parity poisoning on read data at the DFI interface after the parity generation logic, and when MEMC_INLINE_ECC=1 enables poisoning of ECC word after the ECC encoder at the write data interface at the DFI. Programming Mode: Quasi-dynamic Group 3
1 —	Reserved
0 par_poison_en	Enables on-chip parity poisoning on the data interfaces. Programming Mode: Quasi-dynamic Group 3

36.14.114 On-Chip Parity Status 0 (OCPARSTAT0)

Offset

Register	Offset
OCPARSTAT0	338h

Diagram



Fields

Field	Function
31-19 —	Reserved

Table continues on the next page...

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3a624c2

Table continued from the previous page...

Field	Function
18 par_raddr_err_intr_2	Read address parity error interrupt for port 2. This interrupt is asserted when an on-chip read address parity error occurred on the corresponding AXI port's read address channel. Programming Mode: Static
17 par_raddr_err_intr_1	Read address parity error interrupt for port 1. This interrupt is asserted when an on-chip read address parity error occurred on the corresponding AXI port's read address channel. Programming Mode: Static
16 par_raddr_err_intr_0	Read address parity error interrupt for port 0. This interrupt is asserted when an on-chip read address parity error occurred on the corresponding AXI port's read address channel. Programming Mode: Static
15-3 —	Reserved
2 par_waddr_err_intr_2	Write address parity error interrupt for port 2. This interrupt is asserted when an on-chip write address parity error occurred on the corresponding AXI port's write address channel. Programming Mode: Static
1 par_waddr_err_intr_1	Write address parity error interrupt for port 1. This interrupt is asserted when an on-chip write address parity error occurred on the corresponding AXI port's write address channel. Programming Mode: Static
0 par_waddr_err_intr_0	Write address parity error interrupt for port 0. This interrupt is asserted when an on-chip write address parity error occurred on the corresponding AXI port's write address channel. Programming Mode: Static

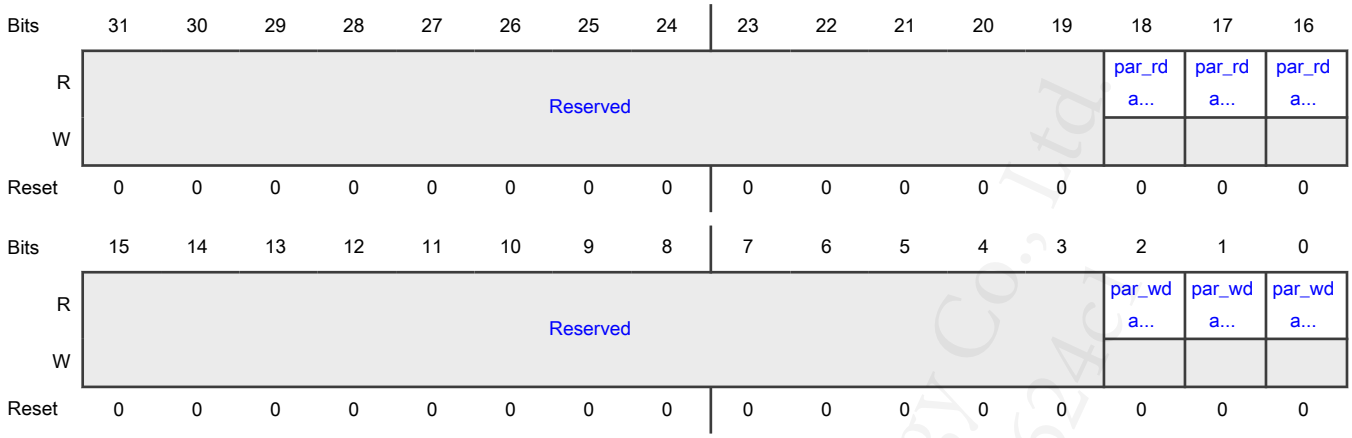
36.14.115 On-Chip Parity Status 1 (OCPARSTAT1)

Offset

Register	Offset
OCPARSTAT1	33Ch

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4888-9253-39d3ab6e4c3

Diagram



Fields

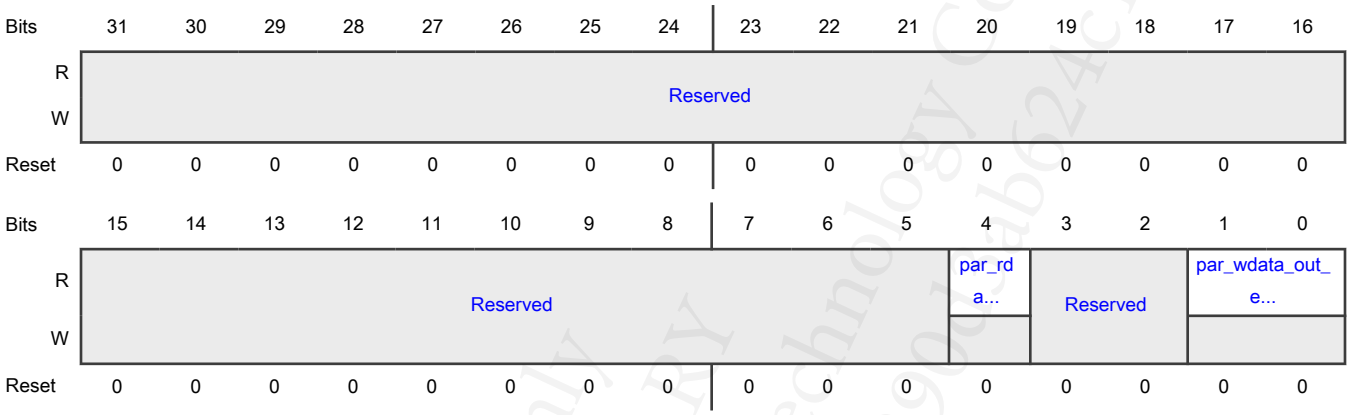
Field	Function
31-19 —	Reserved
18 par_rdata_err_intr_2	Read data parity error interrupt for port 2. This interrupt is asserted when an on-chip read data parity error occurred on the corresponding AXI port's read data channel. Bit 0 corresponds to Port 0, and so on. Cleared by register par_rdata_err_intr_clr. Programming Mode: Static
17 par_rdata_err_intr_1	Read data parity error interrupt for port 1. This interrupt is asserted when an on-chip read data parity error occurred on the corresponding AXI port's read data channel. Bit 0 corresponds to Port 0, and so on. Cleared by register par_rdata_err_intr_clr. Programming Mode: Static
16 par_rdata_err_intr_0	Read data parity error interrupt for port 0. This interrupt is asserted when an on-chip read data parity error occurred on the corresponding AXI port's read data channel. Bit 0 corresponds to Port 0, and so on. Cleared by register par_rdata_err_intr_clr. Programming Mode: Static
15-3 —	Reserved
2 par_wdata_in_err_intr_2	Write data parity error interrupt on input for port 2. This interrupt is asserted when an on-chip write data parity error occurred on the corresponding AXI port's write data channel. Bit 0 corresponds to Port 0, and so on. Cleared by register par_wdata_err_intr_clr. Programming Mode: Static
1 par_wdata_in_err_intr_1	Write data parity error interrupt on input for port 1. This interrupt is asserted when an on-chip write data parity error occurred on the corresponding AXI port's write data channel. Bit 0 corresponds to Port 0, and so on. Cleared by register par_wdata_err_intr_clr. Programming Mode: Static
0 par_wdata_in_err_intr_0	Write data parity error interrupt on input for port 0. This interrupt is asserted when an on-chip write data parity error occurred on the corresponding AXI port's write data channel. Bit 0 corresponds to Port 0, and so on. Cleared by register par_wdata_err_intr_clr. Programming Mode: Static

36.14.116 On-Chip Parity Status 2 (OCPARSTAT2)

Offset

Register	Offset
OCPARSTAT2	340h

Diagram



Fields

Field	Function
31-5 —	Reserved
4 par_rdata_in_err_ecc_intr	Interrupt on ECC data going into inline ECC decoder. Cleared by par_rdata_err_intr_clr. Programming Mode: Static
3-2 —	Reserved
1-0 par_wdata_out_err_intr	Write data parity error interrupt on output. Cleared by register par_wdata_err_intr_clr. Programming Mode: Static

36.14.117 AXI Poison Configuration (POISONCFG)

Offset

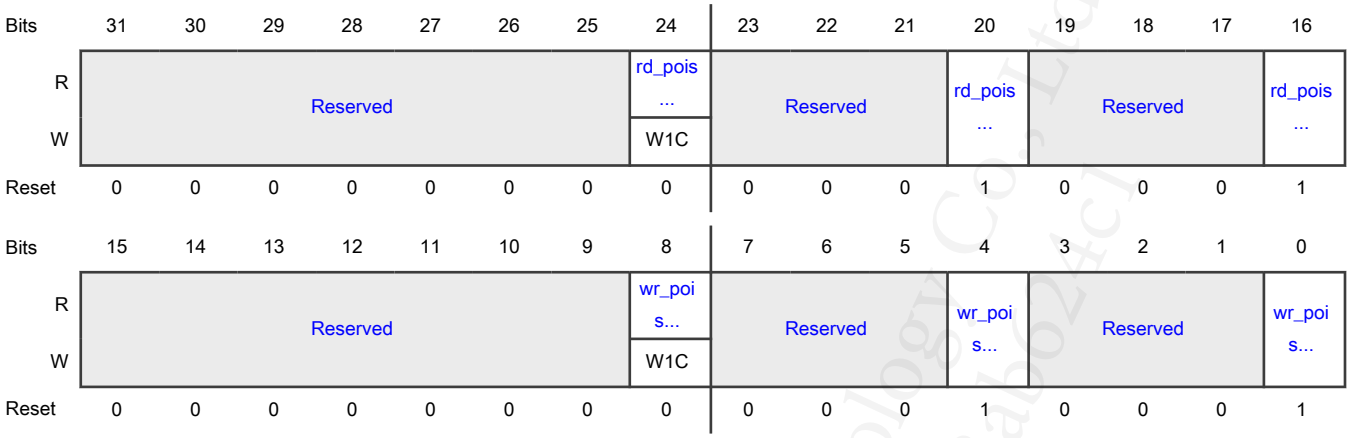
Register	Offset
POISONCFG	36Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CFL Automotive Technology Co., Ltd.
c8c4625-4821-48a0-b251-390d4b624c1

Function

Common for all AXI ports

Diagram



Fields

Field	Function
31-25 —	Reserved
24 rd_poison_intr_clr	Read poison interrupt clear Clears read transaction poisoning interrupts. Allow two to three clock cycles for the core to clear the interrupts. The MC clears this bit automatically. Programming Mode: Dynamic 0b - No effect 1b - Clear
23-21 —	Reserved
20 rd_poison_intr_en	Read poison interrupt enable Enables interrupts for read transaction poisoning. Programming Mode: Dynamic 0b - Disable 1b - Enable
19-17 —	Reserved
16	Read SLVERR enable Enables the AXI slave error (SLVERR) response for read transaction poisoning.

Table continues on the next page...

Table continued from the previous page...

Field	Function
rd_poison_slverr_en	Programming Mode: Dynamic 0b - Disable 1b - Enable
15-9 —	Reserved
8 wr_poison_intr_clr	Write poison interrupt clear Clears write transaction poisoning interrupts. Allow two to three clock cycles for the core to clear the interrupts. The MC clears this bit automatically. Programming Mode: Dynamic 0b - No effect 1b - Clear
7-5 —	Reserved
4 wr_poison_intr_en	Write poison interrupt enable Enables interrupts for write transaction poisoning. Programming Mode: Dynamic 0b - Disable 1b - Enable
3-1 —	Reserved
0 wr_poison_slverr_en	Write SLVERR enable Enables the AXI slave error (SLVERR) response for write transaction poisoning. Programming Mode: Dynamic 0b - Disable 1b - Enable

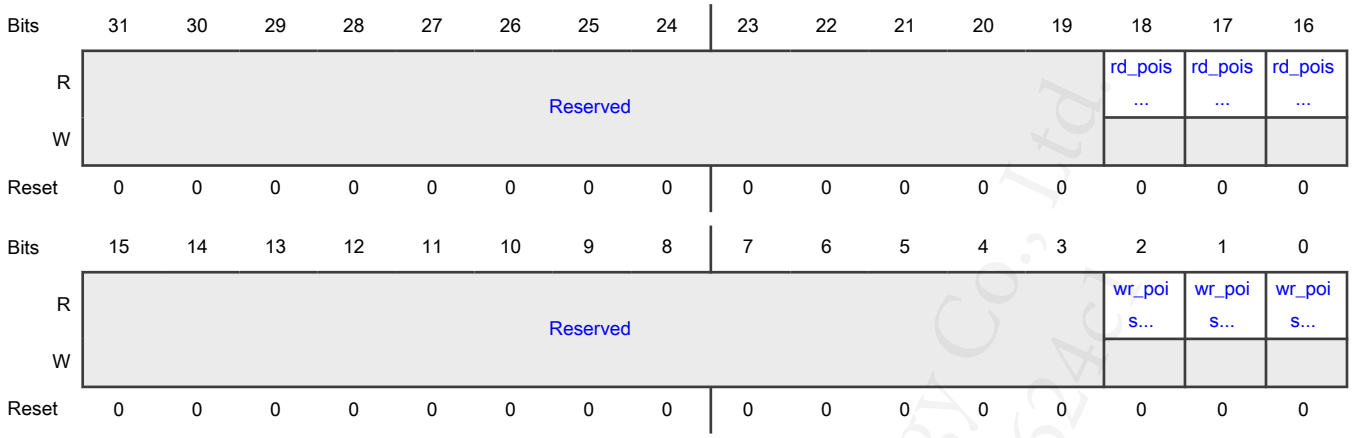
36.14.118 AXI Poison Status (POISONSTAT)

Offset

Register	Offset
POISONSTAT	370h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-4821-48a8-b251-390d3ab624c1

Diagram



Fields

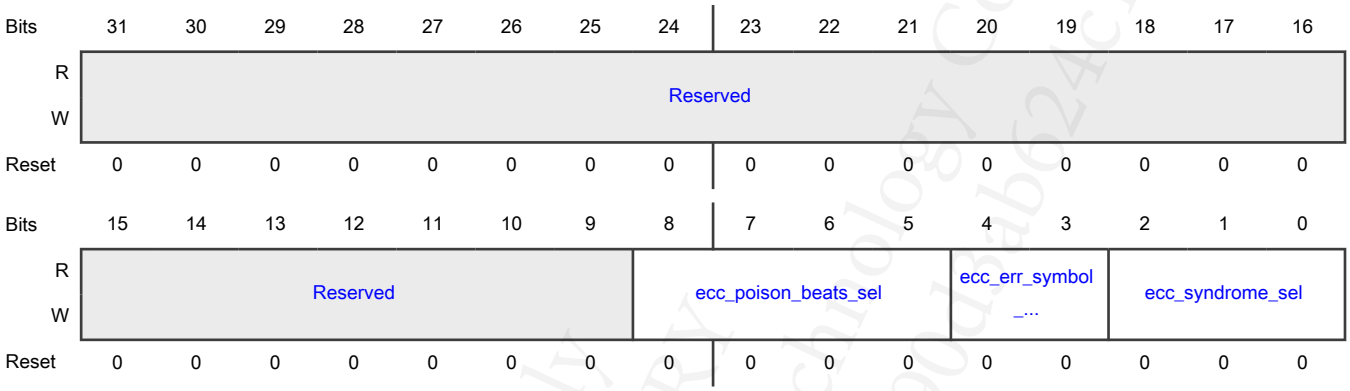
Field	Function
31-19 —	Reserved
18 rd_poison_intr_2	Read transaction poisoning error interrupt for port 2. This register is a APB clock copy (double register synchronizer) of the interrupt asserted when a transaction is poisoned on the corresponding AXI port's read address channel. Bit 0 corresponds to Port 0, and so on. Interrupt is cleared by register rd_poison_intr_clr, then value propagated to APB clock. Programming Mode: Dynamic
17 rd_poison_intr_1	Read transaction poisoning error interrupt for port 1. This register is a APB clock copy (double register synchronizer) of the interrupt asserted when a transaction is poisoned on the corresponding AXI port's read address channel. Bit 0 corresponds to Port 0, and so on. Interrupt is cleared by register rd_poison_intr_clr, then value propagated to APB clock. Programming Mode: Dynamic
16 rd_poison_intr_0	Read transaction poisoning error interrupt for port 0. This register is a APB clock copy (double register synchronizer) of the interrupt asserted when a transaction is poisoned on the corresponding AXI port's read address channel. Bit 0 corresponds to Port 0, and so on. Interrupt is cleared by register rd_poison_intr_clr, then value propagated to APB clock. Programming Mode: Dynamic
15-3 —	Reserved
2 wr_poison_intr_2	Write transaction poisoning error interrupt for port 2. This register is a APB clock copy (double register synchronizer) of the interrupt asserted when a transaction is poisoned on the corresponding AXI port's write address channel. Bit 0 corresponds to Port 0, and so on. Interrupt is cleared by register wr_poison_intr_clr, then value propagated to APB clock. Programming Mode: Dynamic
1 wr_poison_intr_1	Write transaction poisoning error interrupt for port 1. This register is a APB clock copy (double register synchronizer) of the interrupt asserted when a transaction is poisoned on the corresponding AXI port's write address channel. Bit 0 corresponds to Port 0, and so on. Interrupt is cleared by register wr_poison_intr_clr, then value propagated to APB clock. Programming Mode: Dynamic
0 wr_poison_intr_0	Write transaction poisoning error interrupt for port 0. This register is a APB clock copy (double register synchronizer) of the interrupt asserted when a transaction is poisoned on the corresponding AXI port's write address channel. Bit 0 corresponds to Port 0, and so on. Interrupt is cleared by register wr_poison_intr_clr, then value propagated to APB clock. Programming Mode: Dynamic

36.14.119 Advanced ECC Index (ADVECCINDEX)

Offset

Register	Offset
ADVECCINDEX	374h

Diagram



Fields

Field	Function
31-9 —	Reserved
8-5 ecc_poison_beats_sel	Selector of which DRAM beat's poison pattern will be set by ECCPOISONPAT0/1/2 registers. For frequency ratio 1:1 mode, 2 DRAM beats can be poisoned. Set ecc_poison_beats_sel to 0 and given ECCPOISONPAT0/1/2 to set 1st beat's poison pattern; set ecc_poison_beats_sel to 1 and given ECCPOISONPAT0/1/2 to set 2nd beat's poison pattern. For frequency ratio 1:2 mode, 4 DRAM beats can be poisoned. Set ecc_poison_beats_sel from 0~3 to set 1st to 4th beat's poison pattern. The other value is reserved. Programming Mode: Quasi-dynamic Group 1
4-3 ecc_err_symbol_sel	ECC error symbol select Selector of which error symbol's status output to ADVECCSTAT.advecc_err_symbol_pos and advecc_err_symbol_bits. The default is first error symbol. The value should be less than ADVECCSTAT.advecc_num_err_symbol. Programming Mode: Quasi-dynamic Group 1
2-0 ecc_syndrome_sel	Selector of which DRAM beat data output to ECCCSYN0/1/2 as well as ECCUCSYN. In Advanced ECC, the syndrome consist of number of DRAM beats. This register select which beats of data will be output to ECCCSYN0/1/2 and ECCUCSYN0/1/2 registers. Programming Mode: Quasi-dynamic Group 1

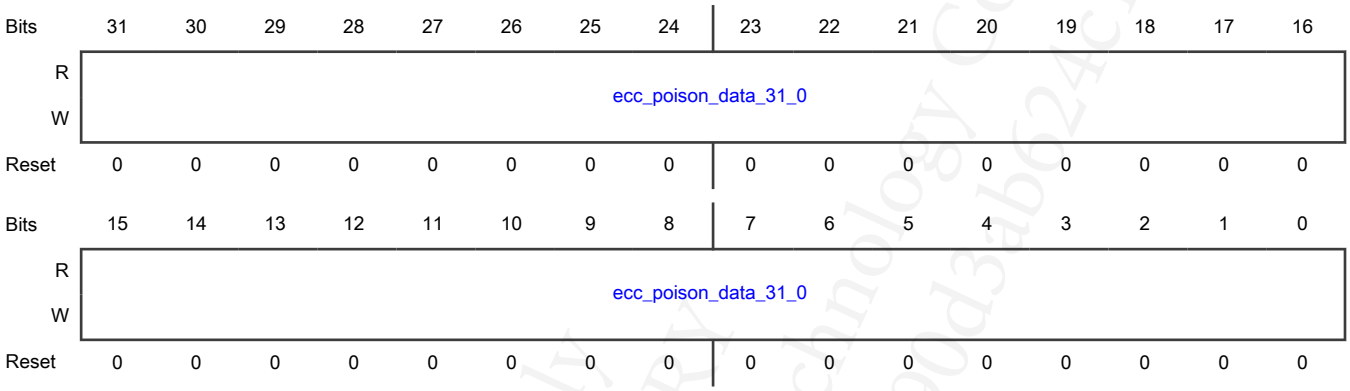
Preliminary NDA only
COMPANY PROPRIETARY
Beijing ESI Automotive Technology Co., Ltd.
c8c462e2-21ff-4b25-390d-b624c1

36.14.120 ECC Poison Pattern 0 (ECCPOISONPAT0)

Offset

Register	Offset
ECCPOISONPAT0	37Ch

Diagram



Fields

Field	Function
31-0 ecc_poison_data_31_0	Poison pattern for DRAM data[31:0]. Only bit is set to 1 indicate poison (invert) corresponding DRAM bit. It is indirect register. Selector is ADVECCINDEX.ecc_poison_beats_sel. Programming Mode: Static

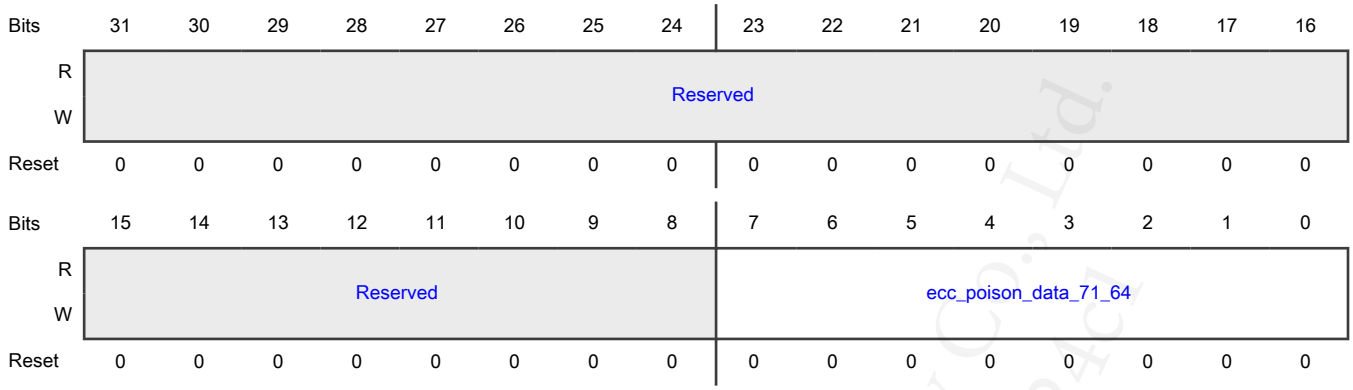
36.14.121 ECC Poison Pattern 2 (ECCPOISONPAT2)

Offset

Register	Offset
ECCPOISONPAT2	384h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b651-390d34b624c1

Diagram



Fields

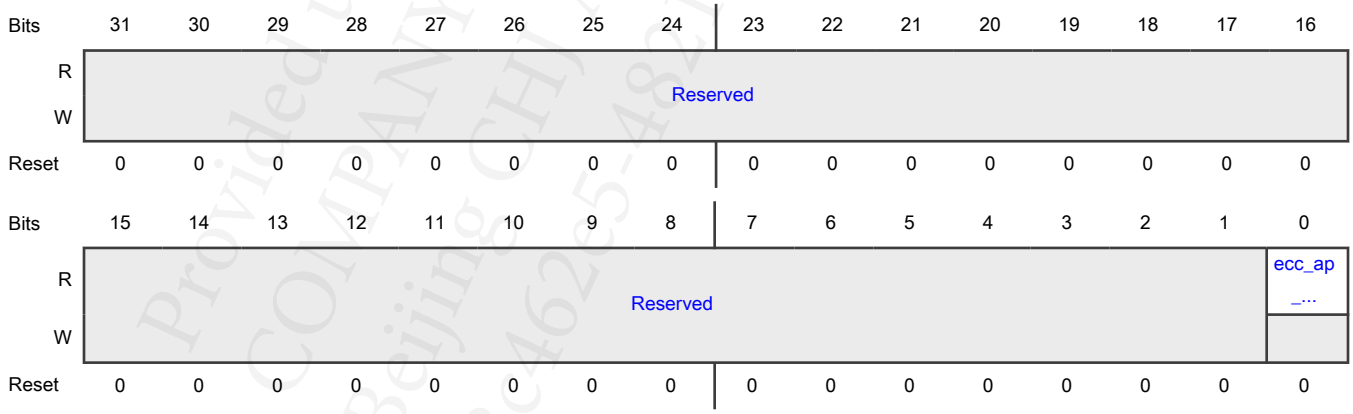
Field	Function
31-8 —	Reserved
7-0 ecc_poison_data_71_64	Poison pattern for DRAM data[71:64]. Only bit is set to 1 indicate poison (invert) corresponding DRAM bit. It is indirect register. Selector is ADVECCINDEX.ecc_poison_beats_sel. Programming Mode: Static

36.14.122 Address protection within ECC Status (ECCAPSTAT)

Offset

Register	Offset
ECCAPSTAT	388h

Diagram



Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c84625-4821-48a8-b251-30d3ab624c1

Fields

Field	Function
31-1 —	Reserved
0 ecc_ap_err	Indicates the number of ECC errors (correctable/uncorrectable) within one burst exceeded the threshold. (ECCCFG0.ecc_ap_err_threshold) Programming Mode: Dynamic

36.14.123 Register parity configuration (REGPARCFG)

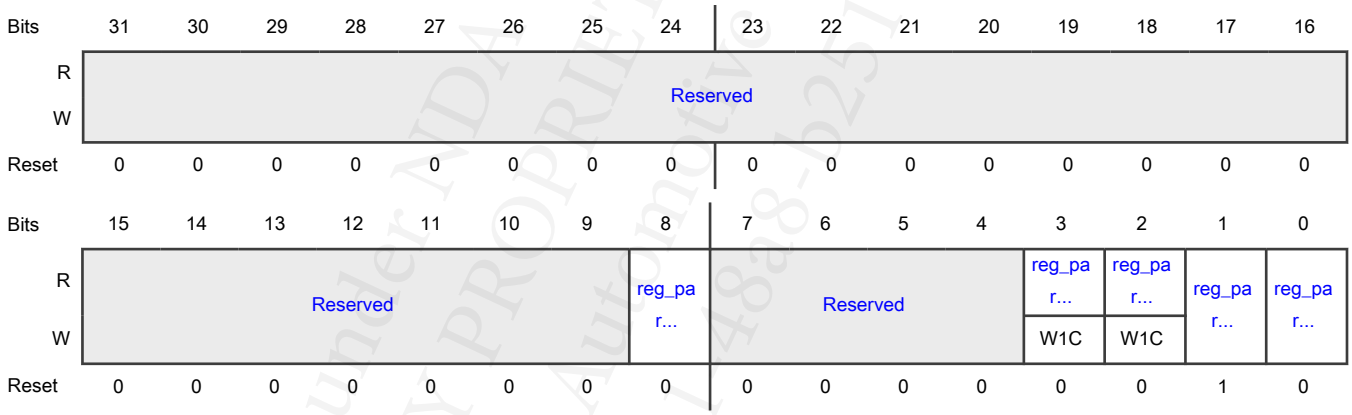
Offset

Register	Offset
REGPARCFG	3C0h

Function

(Note that all fields must be programmed with single write operation)

Diagram



Fields

Field	Function
31-9 —	Reserved
8 reg_par_poison_en	Enable register parity poisoning. Programming Mode: Dynamic, Quasi-Dynamic and static registers.
7-4	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

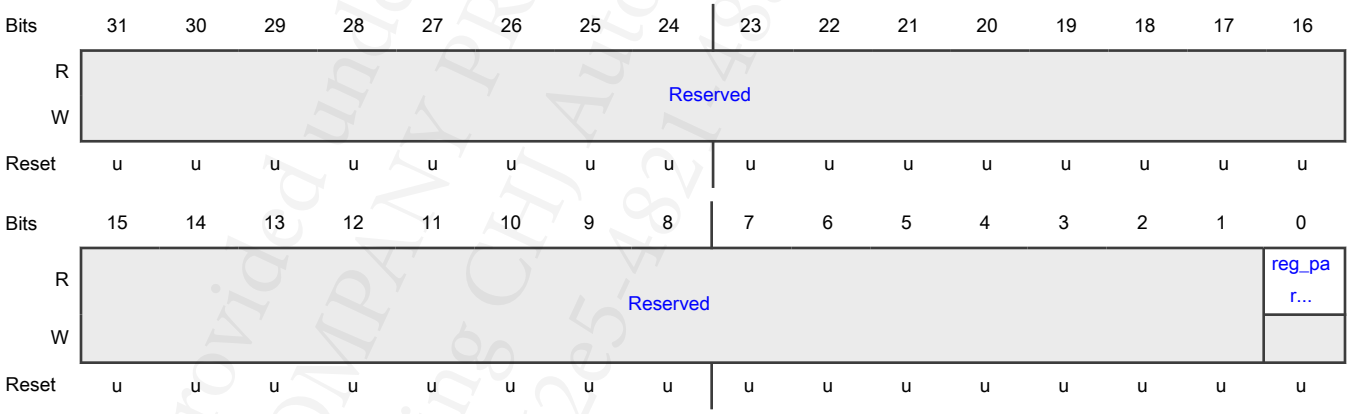
Field	Function
—	
3 reg_par_err_intr_force	Interrupt force bit for reg_par_err_intr, setting this register will cause the output interrupt to be asserted. uMCTL2 automatically clears this bit. There is no interaction between functionally triggering an interrupt and forcing an interrupt (they are mutually exclusive). Programming Mode: Dynamic
2 reg_par_err_intr_clr	Interupt clear bit for reg_par_err_intr. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
1 reg_par_err_intr_en	Enables interrupt generation, if set to 1, on signal reg_par_err_intr upon detection of register parity error. Programming Mode: Dynamic
0 reg_par_en	Register parity enable register. Programming Mode: Dynamic

36.14.124 Register parity status (REGPARSTAT)

Offset

Register	Offset
REGPARSTAT	3C4h

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e54821-48a8-b251-390d8ab6240d

Fields

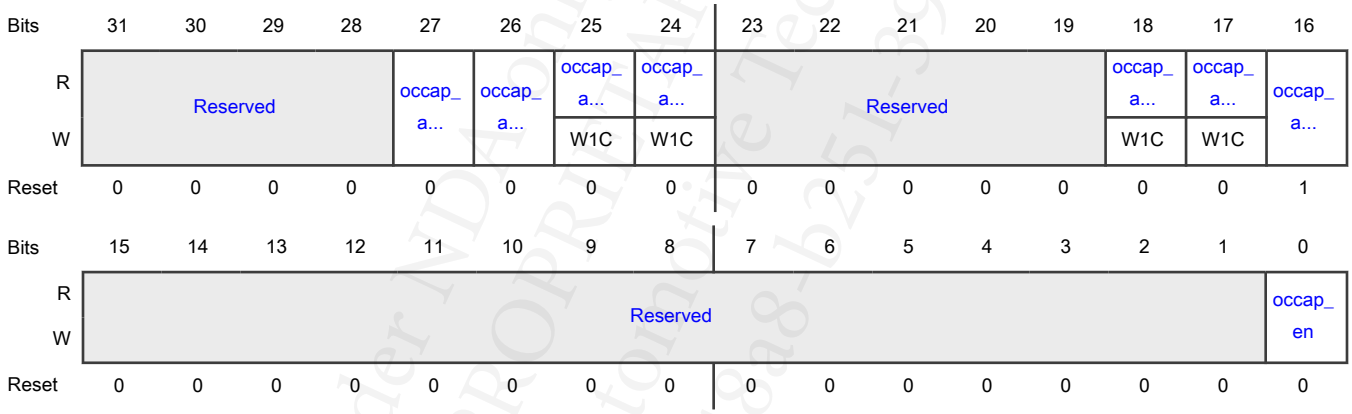
Field	Function
31-1 —	Reserved
0 reg_par_err_intr	Interrupt asserted when Register Parity error is detected. Cleared by setting REGPARCFG.reg_par_err_intr_clr to 1. Programming Mode: Static

36.14.125 On-Chip command/Address Protection Configuration (OCCAPCFG)

Offset

Register	Offset
OCCAPCFG	3E0h

Diagram



Fields

Field	Function
31-28 —	Reserved
27 occap_arb_raq_poison_en	Enables poisoning for the Read Address Queues (RAQ) inside each XPI. Poisoning inverts all parity bits generated by the parity generator. Error will be flagged as soon as the first RAQ is read. This register is not cleared automatically and must be reprogrammed to 0 at the end of the operation. Programming Mode: Dynamic
26 occap_arb_cmp_poison_err_inj	Enable error injection in the poisoning of OCCAP Arbiter logic Injects error into poisoning logic (either parallel or seq) such that XOR logic for one signal is not poisoned when expected. If set, it allows ability to corrupt the following register fields. - 1'b0: OCCAPSTAT.occap_arb_poison_parallel/seq_err=0 - 1'b1:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	OCCAPSTAT.occap_arb_poison_parallel/seq_err=1 Do not change value in same APB write as setting of occap_arb_poison_parallel/_seq Programming Mode: Dynamic
25 occap_arb_cmp_poison_parallel	Enables full poisoning for compare logic inside XPI. Poisoning inverts all bits of all outputs coming from the duplicated modules before the XOR comparators together. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
24 occap_arb_cmp_poison_seq	Enables poisoning for compare logic inside XPI. Poisoning inverts all bits coming from the duplicated modules before the XOR comparators one output at the time per each comparator. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
23-19 —	Reserved
18 occap_arb_intr_force	Interrupt force bit for occap_arb_err_intr, setting this register will cause the output interrupt to be asserted. uMCTL2 automatically clears this bit. There is no interaction between functionally triggering an interrupt and forcing an interrupt (they are mutually exclusive). Programming Mode: Dynamic
17 occap_arb_intr_clr	Interrupt clear bit for occap_arb_err_intr and occap_arb_cmp_poison_complete. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
16 occap_arb_intr_en	Enables interrupt generation upon detection of OCCAP Arbiter errors. Programming Mode: Dynamic
15-1 —	Reserved
0 occap_en	On Chip Command/Address Path Protection (OCCAP) enable register. Programming Mode: Quasi-dynamic Group 3

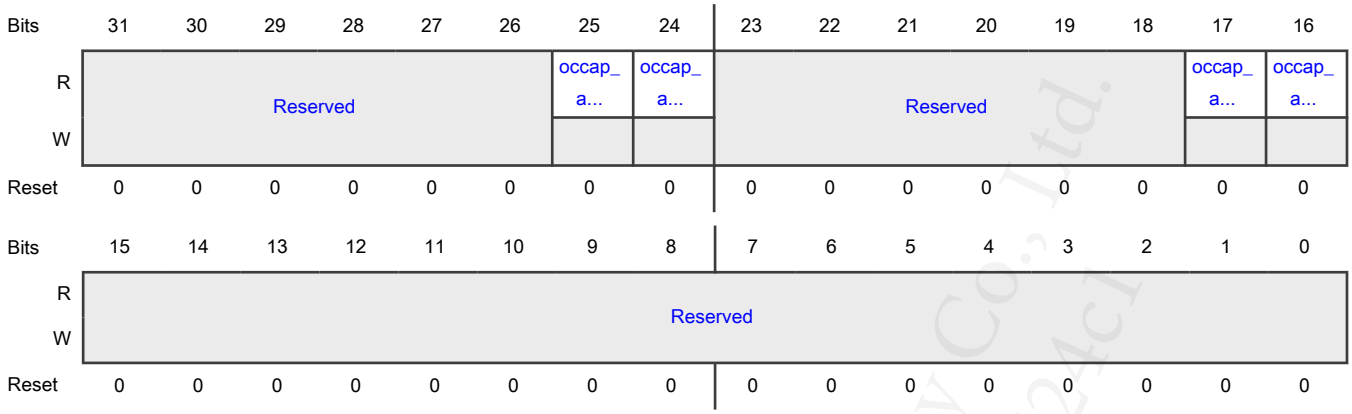
36.14.126 On-Chip command/Address Protection Status (OCCAPSTAT)

Offset

Register	Offset
OCCAPSTAT	3E4h

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-421-48a8-b251-90aab6221

Diagram



Fields

Field	Function
31-26 —	Reserved
25 occap_arb_cmp_poison_parallel_err	Error when occap_arb_cmp_poison_full_en was active due to incorrect no. of errors being occurring. Internal logic checks the number of errors detected while poisoning occurred for occap_arb_cmp_poison_full_en. It checks for error on all of the the corresponding XOR outputs. If multi-bit, checks also that all XOR bits are set. Register is valid when occap_arb_cmp_poison_complete=1. Programming Mode: Static
24 occap_arb_cmp_poison_seq_err	Error when occap_arb_cmp_poison_en was active due to incorrect no. of errors being occurring. Internal logic checks that the correct number of errors detected while poisoning one output at the time occurred for occap_arb_cmp_poison_en. It checks for error on one output at the time. Register is valid when occap_arb_cmp_poison_complete=1. Programming Mode: Static
23-18 —	Reserved
17 occap_arb_cmp_poison_complete	OCCAP ARB comparator poisoning complete interrupt status. Register cleared by OCCAPCFG.occap_arb_intr_clr. Programming Mode: Static
16 occap_arb_err_intr	OCCAP Arbiter error interrupt status. Register cleared by OCCAPCFG.occap_arb_intr_clr. Programming Mode: Static
15-0 —	Reserved

36.14.127 On-Chip command/Address Protection Configuration 1 (OCCAPCFG1)

Offset

Register	Offset
OCCAPCFG1	3E8h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved					occap_d...	occap_d...	occap_d...	Reserved					occap_d...	occap_d...	occap_d...
W							W1C	W1C						W1C	W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					occap_d...	occap_d...	occap_d...	Reserved					occap_d...	occap_d...	occap_d...
W							W1C	W1C						W1C	W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-27 —	Reserved
26 occap_ddrc_ctrl_poison_err_inj	Enable error injection in the poisoning of OCCAP MC CTRL logic Injects error into poisoning logic (either parallel or seq) such that XOR logic for one signal is not poisoned when expected. If set, it allows ability to corrupt the following register fields. - 1'b0: OCCAPSTAT1.occap_ddrc_ctrl_poison_parallel/seq_err=0 - 1'b1: OCCAPSTAT1.occap_ddrc_ctrl_poison_parallel/seq_err=1 Do not change value in same APB write as setting of occap_ddrc_ctrl_poison_parallel/_seq Programming Mode: Dynamic
25 occap_ddrc_ctrl_poison_parallel	Enables poisoning of OCCAP MC CTRL logic for all parts of comparison logic, in parallel. Poisons comparison logic for one core_ddrc_core_clk cycle by inverting a ddrc_ctrl[0]'s signal to XOR logic. ddrc_ctrl[1] related signals are never poisoned. All signals are poisoned in parallel and checks if all signals were poisoned correctly. If this is not the case, this is flagged by OCCAPSTAT1.occap_ddrc_ctrl_poison_parallel_err=1. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
24 occap_ddrc_ctrl_poison_seq	Enables poisoning of OCCAP MC CTRL logic for all parts of comparison logic, in sequence. Poisons comparison logic for one core_ddrc_core_clk cycle by inverting a ddrc_ctrl[0]'s signal to XOR logic. ddrc_ctrl[1] related signals are never poisoned. Each signal from ddrc_ctrl[0] is poisoned in series and checks in turn, that each signal was poisoned correctly. If this is not the case, this is flagged by OCCAPSTAT1.occap_ddrc_ctrl_poison_seq_err=1. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
23-19	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
18 occap_ddrc_ctrl_intr_force	Interrupt force bit for occap_ddrc_ctrl_err_intr, setting this register will cause the output interrupt to be asserted. uMCTL2 automatically clears this bit. There is no interaction between functionally triggering an interrupt and forcing an interrupt (they are mutually exclusive). Programming Mode: Dynamic
17 occap_ddrc_ctrl_intr_clr	Interrupt clear bit for occap_ddrc_ctrl_err_intr. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
16 occap_ddrc_ctrl_intr_en	Enables interrupt generation on signal occap_ddrc_ctrl_err_intr upon detection of OCCAP MC CTRL errors. Programming Mode: Dynamic
15-11 —	Reserved
10 occap_ddrc_data_poison_err_inj	Enable error injection in the poisoning of OCCAP MC DATA logic Injects error into poisoning logic (either parallel or seq) such that XOR logic for one signal is not poisoned when expected. If set, it allows ability to corrupt the following register fields. - 1'b0: OCCAPSTAT1.occap_ddrc_data_poison_parallel/seq_err=0 - 1'b1: OCCAPSTAT1.occap_ddrc_data_poison_parallel/seq_err=1 Do not change value in same APB write as setting of occap_ddrc_data_poison_parallel/_seq Programming Mode: Dynamic
9 occap_ddrc_data_poison_parallel	Enables poisoning of OCCAP MC DATA logic for all parts of comparison logic, in parallel. Poisons comparison logic for one core_ddrc_core_clk cycle by inverting all bits of a signal to XOR logic. All signals of instance[0] of the duplicated modules are poisoned in parallel and checks if all signals were poisoned correctly. If this is not the case, this is flagged by OCCAPSTAT1.occap_ddrc_data_poison_parallel_err=1. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
8 occap_ddrc_data_poison_seq	Enables poisoning of OCCAP MC DATA logic for all parts of comparison logic, in sequence. Poisons comparison logic for one core_ddrc_core_clk cycle by inverting all bits of a signal to XOR logic. All signals of instance[0] of the duplicated modules are poisoned in parallel and checks if all signals were poisoned correctly. If this is not the case, this is flagged by OCCAPSTAT1.occap_ddrc_data_poison_seq_err=1. uMCTL2 automatically clears this bit. Programming Mode: Dynamic
7-3 —	Reserved
2 occap_ddrc_data_intr_force	Interrupt force bit for occap_ddrc_data_err_intr, setting this register will cause the output interrupt to be asserted. uMCTL2 automatically clears this bit. There is no interaction between functionally triggering an interrupt and forcing an interrupt (they are mutually exclusive). Programming Mode: Dynamic
1 occap_ddrc_data_intr_clr	Interrupt clear bit for occap_ddrc_data_err_intr. uMCTL2 automatically clears this bit. Programming Mode: Dynamic

Table continues on the next page...

Table continued from the previous page...

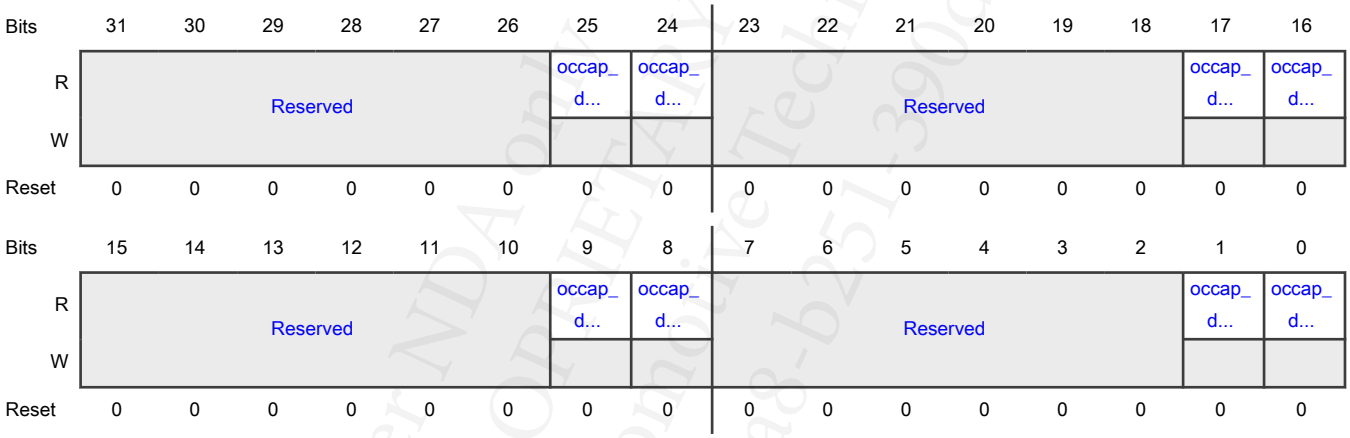
Field	Function
0 occap_ddrc_data_intr_en	Enables interrupt generation on signal occap_ddrc_data_err_intr upon detection of OCCAP MC DATA errors. Programming Mode: Dynamic

36.14.128 On-Chip command/Address Protection Status 1 (OCCAPSTAT1)

Offset

Register	Offset
OCCAPSTAT1	3ECh

Diagram



Fields

Field	Function
31-26 —	Reserved
25 occap_ddrc_ctrl_poison_parallel_err	Error when occap_ddrc_ctrl_poison_parallel was active due to incorrect no. of errors being occurring. Internal logic checks the number of errors detected while poisoning occurred for occap_ddrc_ctrl_poison_parallel. It checks for error on all of the the corresponding XOR outputs. If multi-bit, checks also that all XOR bits are set. It checks all XOR in parallel. Register is valid only when occap_ddrc_ctrl_cmp_poison_complete=1. Programming Mode: Static
24 occap_ddrc_ctrl_poison_seq_err	Error when occap_ddrc_ctrl_poison_seq was active due to incorrect no. of errors being occurring. Internal logic checks the number of errors detected while poisoning occurred for occap_ddrc_ctrl_poison_seq. It checks for error on all of the corresponding XOR outputs. It checks each XOR sequentially. Register is valid only when occap_ddrc_ctrl_cmp_poison_complete=1. Programming Mode: Static

Table continues on the next page...

Table continued from the previous page...

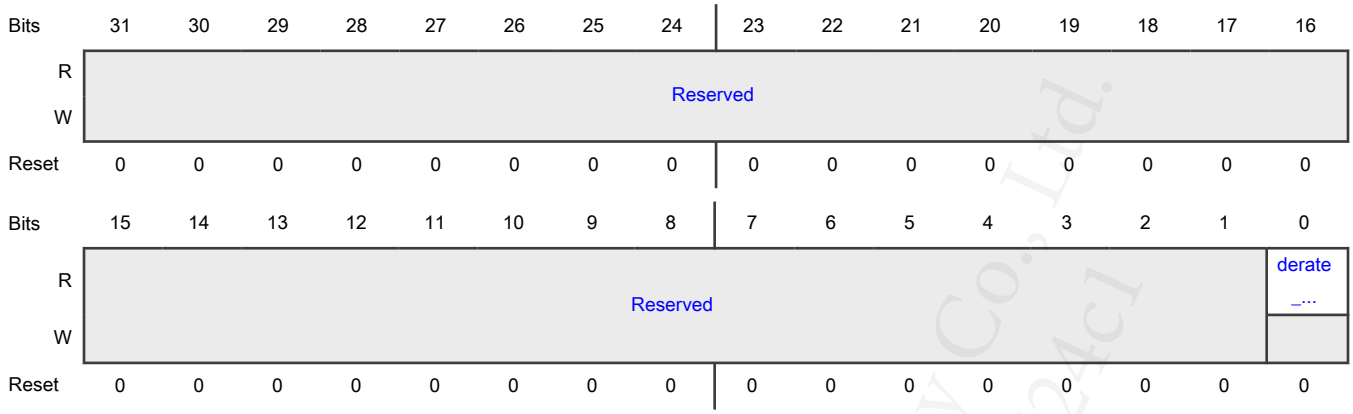
Field	Function
23-18 —	Reserved
17 occap_ddrc_ctrl_poison_complete	OCCAP MC CTRL poisoning complete interrupt status. Register cleared by OCCAPCFG1.occap_ddrc_ctrl_err_intr_clr. Programming Mode: Static
16 occap_ddrc_ctrl_err_intr	OCCAP MC CTRL error interrupt status. Register cleared by OCCAPCFG1.occap_ddrc_ctrl_err_intr_clr. Programming Mode: Static
15-10 —	Reserved
9 occap_ddrc_data_poison_parallel_err	Error when occap_ddrc_data_poison_parallel was active due to incorrect no. of errors being occurring. Internal logic checks the number of errors detected while poisoning occurred for occap_ddrc_data_poison_parallel. It checks for error on all of the the corresponding XOR outputs. If multi-bit, checks also that all XOR bits are set. It checks all XOR in parallel. This is cleared when OCCAPCFG1.occap_ddrc_data_poison_parallel=0 occurs. Programming Mode: Static
8 occap_ddrc_data_poison_seq_err	Error when occap_ddrc_data_poison_seq was active due to incorrect no. of errors being occurring. Internal logic checks the number of errors detected while poisoning occurred for occap_ddrc_data_poison_seq. It checks for error on all of the corresponding XOR outputs. It checks each XOR sequentially. This is cleared when OCCAPCFG1.occap_ddrc_data_poison_seq=0 occurs. Programming Mode: Static
7-2 —	Reserved
1 occap_ddrc_data_poison_complete	OCCAP MC DATA poisoning complete interrupt status. Register cleared by OCCAPCFG1.occap_ddrc_data_err_intr_clr. Programming Mode: Static
0 occap_ddrc_data_err_intr	OCCAP MC DATA error interrupt status. Register cleared by OCCAPCFG1.occap_ddrc_data_err_intr_clr. Programming Mode: Static

36.14.129 Temperature Derate Status (DERATESTAT)

Offset

Register	Offset
DERATESTAT	3F0h

Diagram



Fields

Field	Function
31-1 —	Reserved
0 derate_temp_limit_intr	Derate temperature interrupt indicating LPDDR2/3/4 DRAM temperature operating limit is exceeded. This register field is set to 1 when the value read from MR4[2:0] is 3'b000 or 3'b111. Cleared by register DERATECTL.derate_temp_limit_intr_clr. Programming Mode: Static

36.15 Performance monitor register descriptions

36.15.1 PERF memory map

Perf_Registers base address: 403E_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Counter control 0 (COUNTER_0_CTRL_REG_ADDR)	32	RW	0000_0000h
4h	Counter control 1 (COUNTER_1_CTRL_REG_ADDR)	32	RW	0000_0000h
8h	Counter control 2 (COUNTER_2_CTRL_REG_ADDR)	32	RW	0000_0000h
Ch	Counter control 3 (COUNTER_3_CTRL_REG_ADDR)	32	RW	0000_0000h
20h - 2Ch	Counter Data (COUNTER_0_DATA_REG_ADDR - COUNTER_3_DATA_REG_ADDR)	32	RO	0000_0000h
40h	Mode Register Read 0 address (MRR_0_DATA_REG_ADDR)	32	RW	0000_0000h
44h	Mode Register Read 1 address (MRR_1_DATA_REG_ADDR)	32	RO	0000_0000h

36.15.2 Counter control 0 (COUNTER_0_CTRL_REG_ADDR)

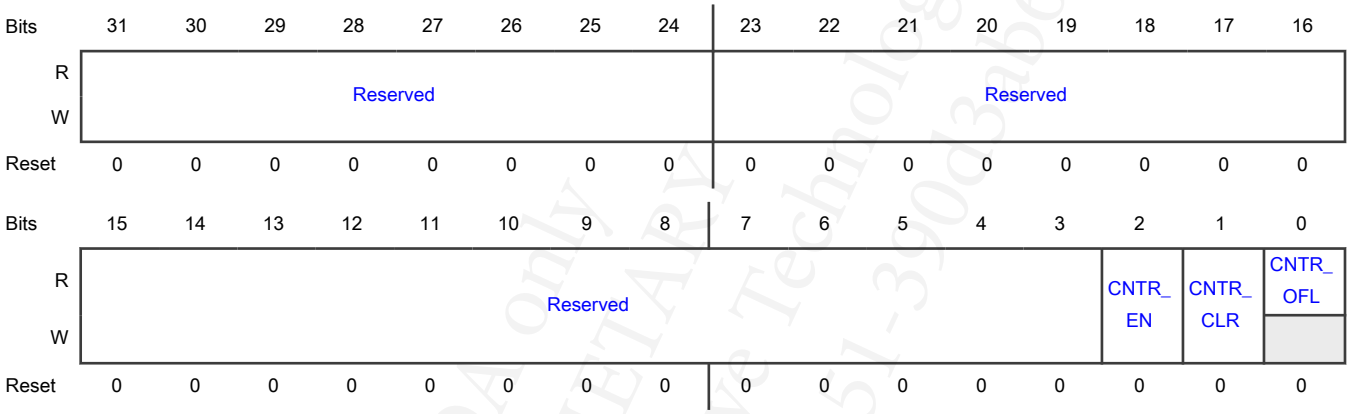
Offset

Register	Offset
COUNTER_0_CTRL_REG_ADDR	0h

Function

Provides control for counter 0 and indicates whether the counter has expired (overflowed).

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-3 —	Reserved
2 CNTR_EN	Counter enable 0b - Disable (clock gated) 1b - Enable. If CNT_CLR is 1, the counter begins counting.
1 CNTR_CLR	Counter clear Sets the counter value to 0 or 1. 0b - Set counter to 0 and clear counter overflow (CNTR_OFL).

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-39003b624c1

Table continued from the previous page...

Field	Function
	1b - Set counter to 1. If the counter is enabled (CNTR_EN), the counter starts counting. This has an effect only if CNTR_CLR is 0; otherwise, it is ignored.
0 CNTR_OFL	Counter overflow Indicates whether the counter has expired (overflowed). 0b - Expired (and no longer counting) 1b - Not expired (and still counting)

36.15.3 Counter control 1 (COUNTER_1_CTRL_REG_ADDR)

Offset

Register	Offset
COUNTER_1_CTRL_REG_ADDR	4h

Function

Provides control for counter 1 and indicates whether the counter has expired (overflowed).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CSV								COUNT_PR							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved													CNTR_EN	CNTR_CLR	CNTR_OFL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 CSV	Count_value_select[7:0] Selects which counter is to be applied.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-390d3ab624c1

Field	Function

CSV[7:0]	Value	comment
8'h00	No increment (zero)	Counter will stop incrementing.
8'h01	stat_ddrc_reg_selfref_type = CP[1:0]	Counts when ram is running vs self refresh
8'h04	NA	NA
8'h05	WAQ_POP_0 = 1	Number of writes
8'h08	NA	NA
8'h09	WAQ_WCOUNT_0[3:0] = CP[3:0]	Write Queue depth is fixed number X (Number of used positions)
8'h10	LPR_CREDIT_CNT[6:0] = CP[6:0]	LP read cam has X entries (Number of available CAM free slots)
8'h11	HPR_CREDIT_CNT[6:0] = CP[6:0]	HP read cam has X entries(Number of available CAM free slots)
8'h12	WR_CREDIT_CNT[6:0] = CP[6:0]	write cam has X entries (Number of available CAM free slots)
8'h2X	Group 0 counts from perf signals	X stands for as specified below
0000	perf_hif_rd	asserts for every read command to the DDRC
0001	perf_hif_wr	assert for every write command to the DDRC
0010	perf_hif_rmw	assert for every read modify write to the DDRC
0011	perf_hif_hi_pri_rd	assert for every high priority read
0100	perf_hpr_req_with_nocredit	high performance request when full
0101	perf_hpr_xact_when_critical	high performance request when critical
0110	perf_lpr_req_with_nocredit	low performance request when full
0111	perf_lpr_xact_when_critical	low performance request when critical
1001	perf_wr_xact_when_critical	write request when critical
1010	perf_dfi_rd_data_cycles	assert for every read data cycle
1011	perf_dfi_wr_data_cycles	assert for every write data cycle
8'h3X	Group 1 counts from perf signals	X stands for values as specified below

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHI Automotive Technology Co. Ltd.
c8c45e42-1da8-b251-e990-0a2a1

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>CSV[7:0]</th><th>Value</th><th>comment</th></tr>
<tr><td>0000</td><td>perf_rdwr_transitions</td><td>assert for every read->write or write->read transition</td></tr>
<tr><td>0001</td><td>perf_op_is_precharge</td><td>assert for every precharge issued</td></tr>
<tr><td>0010</td><td>perf_op_is_activate</td><td>assert for every activate issued</td></tr>
<tr><td>0011</td><td>perf_op_is_load_mode</td><td>assert for every MRW or MRR</td></tr>
<tr><td>0100</td><td>perf_op_is_mwr</td><td></td></tr>
<tr><td>0101</td><td>perf_op_is_rd</td><td></td></tr>
<tr><td>0110</td><td>perf_op_is_rd_activate</td><td></td></tr>
<tr><td>0111</td><td>perf_op_is_refresh</td><td></td></tr>
<tr><td>1000</td><td>perf_op_is_wr</td><td></td></tr>
<tr><td>1001</td><td>perf_raw_hazard</td><td>RAW or WAW hazard</td></tr>
</table>
</td></tr>
<tr><td>23-16
COUNT_PR</td><td>Count_parameter[7:0]
00000000b - Counter does not count (clock gated).</td></tr>
<tr><td>15-3
—</td><td>Reserved</td></tr>
<tr><td>2
CNTR_EN</td><td>Counter Enable
0b - Disable (clock gated)
1b - Enable. If CNT_CLR is 1, the counter begins counting.</td></tr>
<tr><td>1
CNTR_CLR</td><td>Counter clear
Sets the counter value to 0.
NOTE
On reset, the MC clears the counter and writes 1 to this field. To start counting, you must enable the counter (CNTR_EN).
0b - Set counter value to 0 and clear counter overflow (CNTR_OFL). In response, the MC automatically writes 1 to this field to start the counter. If the counter is enabled (CNTR_EN), the counter increments if the selected event is available and begins counting.
1b - No clearing effect. Normal counter operation.</td></tr>
<tr><td>0
CNTR_OFL</td><td>Counter overflow
Indicates whether the counter has expired (overflowed).
0b - Expired (and no longer counting)
1b - Not expired (and still counting)</td></tr>
</table>

36.15.4 Counter control 2 (COUNTER_2_CTRL_REG_ADDR)

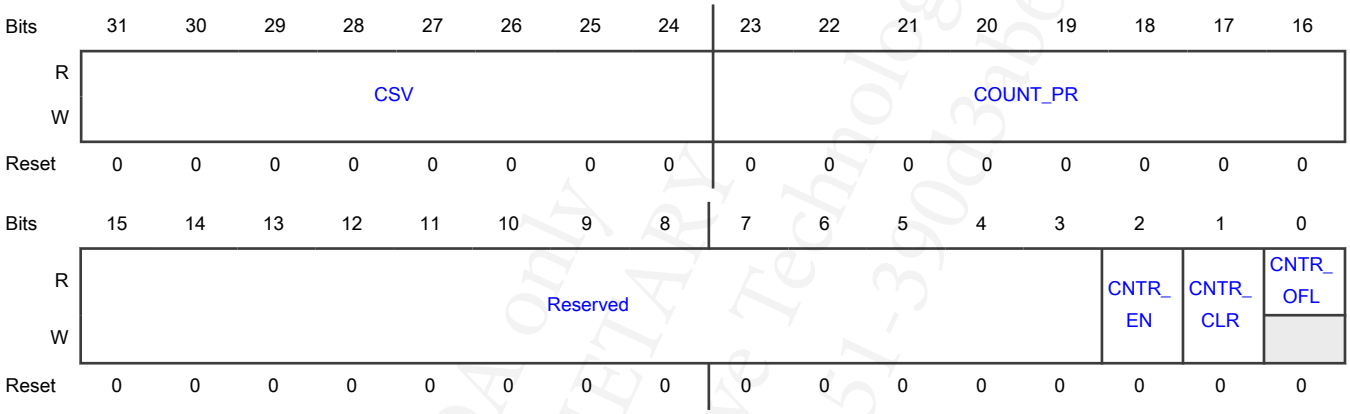
Offset

Register	Offset
COUNTER_2_CTRL_REG_ADDR	8h

Function

Provides control for counter 2 and indicates whether the counter has expired (overflowed).

Diagram



Fields

Field	Function
31-24 CSV	CSV[7:0] Selects which counter is to be applied.

CSV[7:0]	Value	comment
8'h00	No increment (zero)	Counter will stop incrementing.
8'h01	stat_ddrc_reg_selfref_type = CP[1:0]	Counts when ram is running vs self refresh
8'h04	NA	NA
8'h05	WAQ_POP_0 = 1	Number of writes
8'h08	NA	NA
8'h09	WAQ_WCOUNT_0[3:0] = CP[3:0]	Write Queue depth is fixed number X (Number of used positions)
8'h10	LPR_CREDIT_CNT[6:0] = CP[6:0]	LP read cam has X entries (Number of available CAM free slots)

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHIPSEA Automotive Technology Co., Ltd.
c8c46e3f-1148a8-b251-39003b624c1

Field	Function

CSV[7:0]	Value	comment
8'h11	HPR_CREDIT_CNT[6:0] = CP[6:0]	HP read cam has X entries(Number of available CAM free slots)
8'h12	WR_CREDIT_CNT[6:0] = CP[6:0]	write cam has X entries (Number of available CAM free slots)
8'h2X	Group 0 counts from perf signals	X stands for as specified below
0000	perf_hif_rd	asserts for every read command to the DDRC
0001	perf_hif_wr	assert for every write command to the DDRC
0010	perf_hif_rmw	assert for every read modify write to the DDRC
0011	perf_hif_hi_pri_rd	assert for every high priority read
0100	perf_hpr_req_with_nocredit	high performance request when full
0101	perf_hpr_xact_when_critical	high performance request when critical
0110	perf_lpr_req_with_nocredit	low performance request when full
0111	perf_lpr_xact_when_critical	low performance request when critical
1001	perf_wr_xact_when_critical	write request when critical
1010	perf_dfi_rd_data_cycles	assert for every read data cycle
1011	perf_dfi_wr_data_cycles	assert for every write data cycle
8'h3X	Group 1 counts from perf signals	X stands for values as specified below
0000	perf_rdwr_transitions	assert for every read->write or write->read transition
0001	perf_op_is_precharge	assert for every precharge issued
0010	perf_op_is_activate	assert for every activate issued
0011	perf_op_is_load_mode	assert for every MRW or MRR
0100	perf_op_is_mwr	
0101	perf_op_is_rd	
0110	perf_op_is_rd_activate	
0111	perf_op_is_refresh	
1000	perf_op_is_wr	

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHIPANALYSE Technology Co.,Ltd.
8846 8821 1851-3900

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>CSV[7:0]</th><th>Value</th><th>comment</th></tr>
<tr><td>1001</td><td>perf_raw_hazard</td><td>RAW or WAW hazard</td></tr>
</table>
</td></tr>
<tr><td>23-16
COUNT_PR</td><td>Count_parameter[7:0]
00000000b - Counter does not count (clock gated).</td></tr>
<tr><td>15-3
—</td><td>Reserved</td></tr>
<tr><td>2
CNTR_EN</td><td>Counter Enable
0b - Disable (clock gated)
1b - Enable. If CNT_CLR is 1, the counter begins counting.</td></tr>
<tr><td>1
CNTR_CLR</td><td>Counter clear
Sets the counter value to 0.
NOTE
On reset, the MC clears the counter and writes 1 to this field. To start counting, you must enable the counter (CNTR_EN).
0b - Set counter value to 0 and clear counter overflow (CNTR_OFL). In response, the MC automatically writes 1 to this field to start the counter. If the counter is enabled (CNTR_EN), the counter increments if the selected event is available and begins counting.
1b - No clearing effect. Normal counter operation.</td></tr>
<tr><td>0
CNTR_OFL</td><td>Counter overflow
Indicates whether the counter has expired (overflowed).
0b - Expired (and no longer counting)
1b - Not expired (and still counting)</td></tr>
</table>

36.15.5 Counter control 3 (COUNTER_3_CTRL_REG_ADDR)

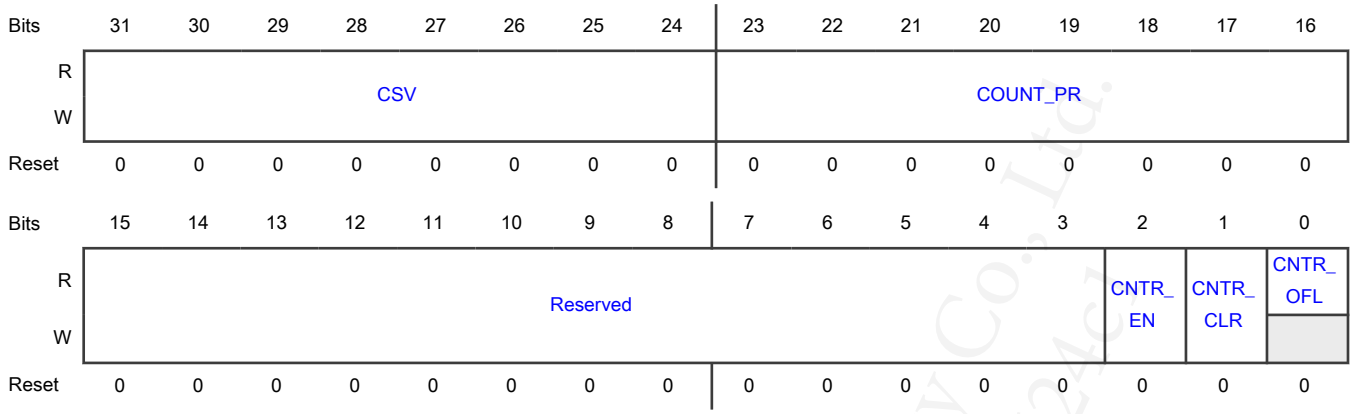
Offset

Register	Offset
COUNTER_3_CTRL_REG_ADDR	Ch

Function

Provides control for counter 3 and indicates whether the counter has expired (overflowed).

Diagram



Fields

Field	Function
31-24 CSV	CSV[7:0] Selects which counter is to be applied.

CSV[7:0]	Value	comment
8'h00	No increment (zero)	Counter will stop incrementing.
8'h01	stat_ddrc_reg_selfref_type = CP[1:0]	Counts when ram is running vs self refresh
8'h04	NA	NA
8'h05	WAQ_POP_0 = 1	Number of writes
8'h08	NA	NA
8'h09	WAQ_WCOUNT_0[3:0] = CP[3:0]	Write Queue depth is fixed number X (Number of used positions)
8'h10	LPR_CREDIT_CNT[6:0] = CP[6:0]	LP read cam has X entries (Number of available CAM free slots)
8'h11	HPR_CREDIT_CNT[6:0] = CP[6:0]	HP read cam has X entries(Number of available CAM free slots)
8'h12	WR_CREDIT_CNT[6:0] = CP[6:0]	write cam has X entries (Number of available CAM free slots)
8'h2X	Group 0 counts from perf signals	X stands for as specified below
0000	perf_hif_rd	asserts for every read command to the DDRC
0001	perf_hif_wr	assert for every write command to the DDRC

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHI Automotive Technology Co., Ltd.
c8c46e...421...4ca8-b251390d3ab624

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>CSV[7:0]</th><th>Value</th><th>comment</th></tr>
<tr><td>0010</td><td>perf_hif_rmw</td><td>assert for every read modify write to the DDRC</td></tr>
<tr><td>0011</td><td>perf_hif_hi_pri_rd</td><td>assert for every high priority read</td></tr>
<tr><td>0100</td><td>perf_hpr_req_with_nocredit</td><td>high performance request when full</td></tr>
<tr><td>0101</td><td>perf_hpr_xact_when_critical</td><td>high performance request when critical</td></tr>
<tr><td>0110</td><td>perf_lpr_req_with_nocredit</td><td>low performance request when full</td></tr>
<tr><td>0111</td><td>perf_lpr_xact_when_critical</td><td>low performance request when critical</td></tr>
<tr><td>1001</td><td>perf_wr_xact_when_critical</td><td>write request when critical</td></tr>
<tr><td>1010</td><td>perf_dfi_rd_data_cycles</td><td>assert for every read data cycle</td></tr>
<tr><td>1011</td><td>perf_dfi_wr_data_cycles</td><td>assert for every write data cycle</td></tr>
<tr><td>8'h3X</td><td>Group 1 counts from perf signals</td><td>X stands for values as specified below</td></tr>
<tr><td>0000</td><td>perf_rdwr_transitions</td><td>assert for every read->write or write->read transition</td></tr>
<tr><td>0001</td><td>perf_op_is_precharge</td><td>assert for every precharge issued</td></tr>
<tr><td>0010</td><td>perf_op_is_activate</td><td>assert for every activate issued</td></tr>
<tr><td>0011</td><td>perf_op_is_load_mode</td><td>assert for every MRW or MRR</td></tr>
<tr><td>0100</td><td>perf_op_is_mwr</td><td></td></tr>
<tr><td>0101</td><td>perf_op_is_rd</td><td></td></tr>
<tr><td>0110</td><td>perf_op_is_rd_activate</td><td></td></tr>
<tr><td>0111</td><td>perf_op_is_refresh</td><td></td></tr>
<tr><td>1000</td><td>perf_op_is_wr</td><td></td></tr>
<tr><td>1001</td><td>perf_raw_hazard</td><td>RAW or WAW hazard</td></tr>
</table>
</td></tr>
<tr><td>23-16
COUNT_PR</td><td>Count_parameter[7:0]
00000000b - Counter does not count (clock gated).</td></tr>
<tr><td>15-3
—</td><td>Reserved</td></tr>
<tr><td>2
CNTR_EN</td><td>Counter Enable</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable (clock gated) 1b - Enable
1 CNTR_CLR	Counter clear Sets the counter value to 0. **NOTE** On reset, the MC clears the counter and writes 1 to this field. To start counting, you must enable the counter (CNTR_EN). 0b - Set counter value to 0 and clear counter overflow (CNTR_OFL). In response, the MC automatically writes 1 to this field to start the counter. If the counter is enabled (CNTR_EN), the counter increments if the selected event is available and begins counting. 1b - No clearing effect. Normal counter operation.
0 CNTR_OFL	Counter overflow Indicates whether the counter has expired (overflowed). 0b - Expired (and no longer counting) 1b - Not expired (and still counting)

36.15.6 Counter Data (COUNTER_0_DATA_REG_ADDR - COUNTER_3_DATA_REG_ADDR)

Offset

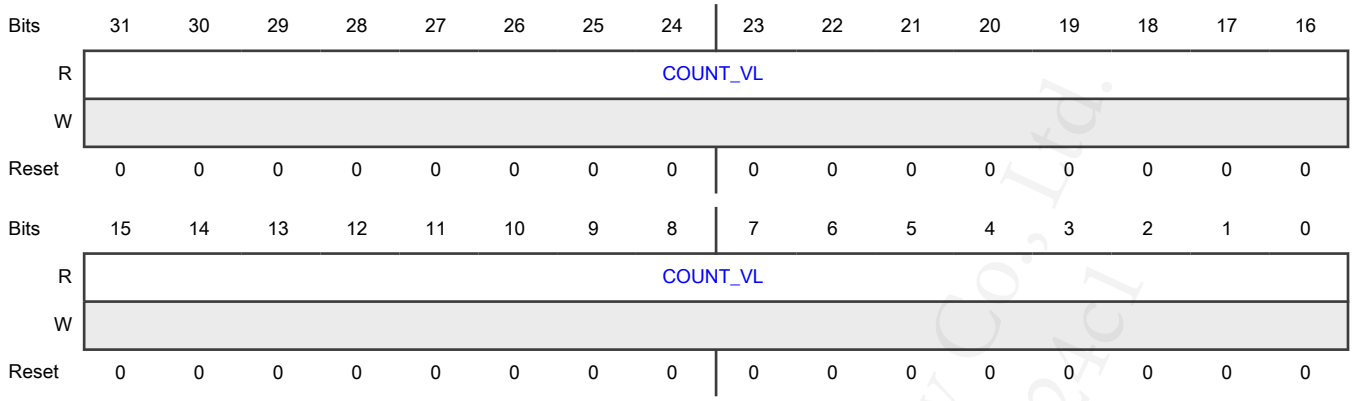
Register	Offset
COUNTER_0_DATA_REG_ADDR	20h
COUNTER_1_DATA_REG_ADDR	24h
COUNTER_2_DATA_REG_ADDR	28h
COUNTER_3_DATA_REG_ADDR	2Ch

Function

Counter 0 counts real-time clock cycles. Counters 1-3 count DDR events.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3b62dc1

Diagram



Fields

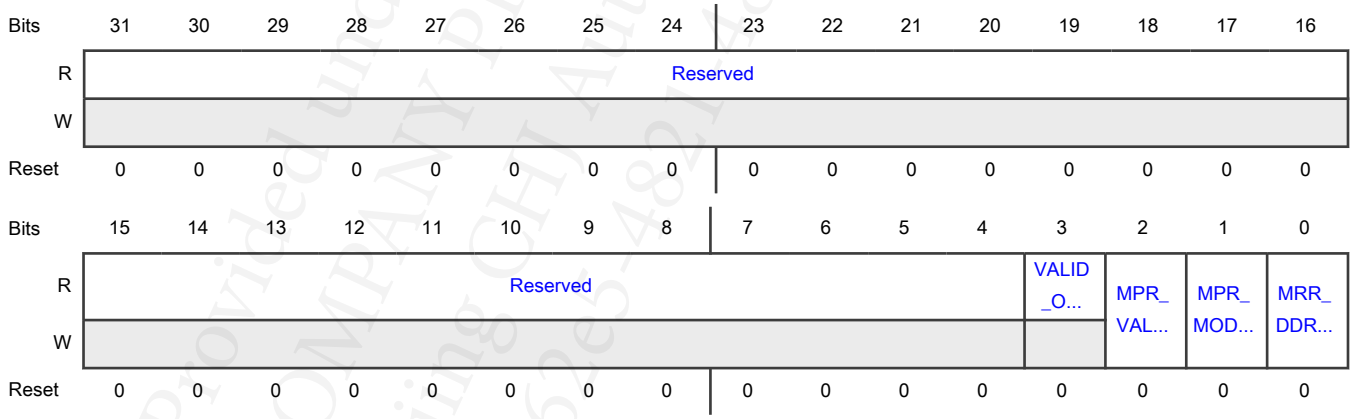
Field	Function
31-0 COUNT_VL	Count Value The current counter value.

36.15.7 Mode Register Read 0 address (MRR_0_DATA_REG_ADDR)

Offset

Register	Offset
MRR_0_DATA_REG_ADDR	40h

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e65-4821-48a8-b251-590d3ab624c1

Fields

Field	Function
31-4 —	Always read zero
3 VALID_OUT	Valid out
2 MPR_VALID_CLR_REG	MPR_VALID_CLR_REG = 1 clears the VALID_OUT of 'Mode Register Read 0 address' and also clears the 'Mode Register Read 1 address'.
1 MPR_MODE_SEL_REG	Set MPR_MODE_SEL_REG = 1 to enable serial mode and Set MPR_MODE_SEL_REG = 0 to enable parallel mode.
0 MRR_DDR_SEL_REG	Set MPR_MODE_SEL_REG = 1 to enable LPDDR4 mode and Set MPR_MODE_SEL_REG = 0 to enable DDR4 mode.

36.15.8 Mode Register Read 1 address (MRR_1_DATA_REG_ADDR)

Offset

Register	Offset
MRR_1_DATA_REG_ADDR	44h

Function

Table 208. Details of MRR_VL meaning

If MRR_0_DATA_REG_ADDR DDR[MRR_DDR_SEL_REG] is:	and MRR_0_DATA_REG_ADDR DDR[MRR_MODE_SEL_REG] is:	then MRR_1_DATA_REG_ADDR DDR[MRR_VL] is:
0	0	A concatenation data in this order: 1. data_in_d2[31:24] 2. data_in_d2[23:16] 3. data_in_d2[15:8] 4. data_in_d2[7:0]
0	1	A concatenation data in this order: 1. data_in_d[120] 2. data_in_d[88]

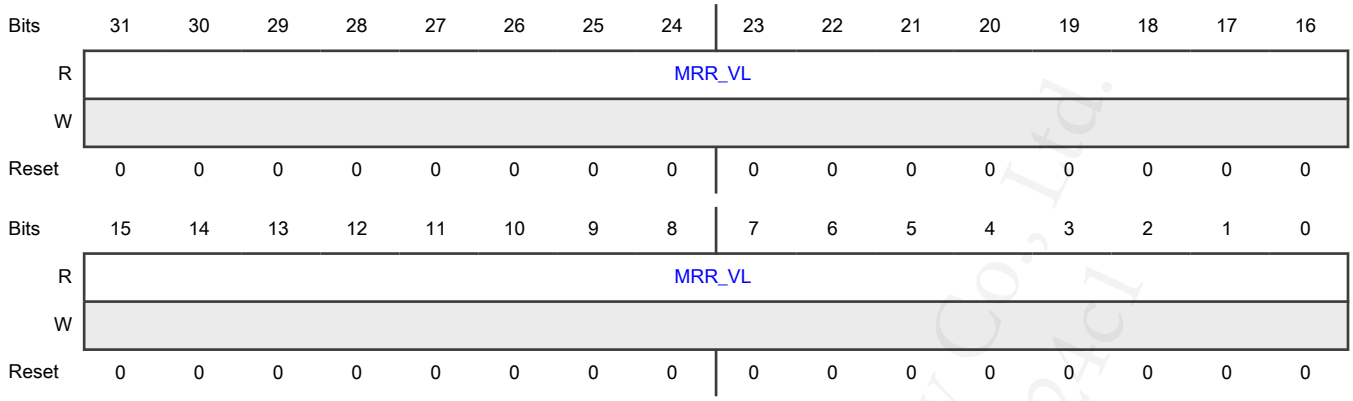
Table continues on the next page...

Table 208. Details of MRR_VL meaning (continued)

If MRR_0_DATA_REG_ADDR DDR[MRR_DDR_SEL_REG] is:	and MRR_0_DATA_REG_ADDR DDR[MRR_MODE_SEL_REG] is:	then MRR_1_DATA_REG_ADDR DDR[MRR_VL] is:
		3. data_in_d[56] 4. data_in_d[24] 5. data_in_d2[120] 6. data_in_d2[88] 7. data_in_d2[56] 8. data_in_d2[24] 9. data_in_d[112] 10. data_in_d[80] 11. data_in_d[48] 12. data_in_d[16] 13. data_in_d2[112] 14. data_in_d2[80] 15. data_in_d2[48] 16. data_in_d2[16] 17. data_in_d[104] 18. data_in_d[72] 19. data_in_d[40] 20. data_in_d[8] 21. data_in_d2[104] 22. data_in_d2[72] 23. data_in_d2[40] 24. data_in_d2[8] 25. data_in_d[96] 26. data_in_d[96] 27. data_in_d[32] 28. data_in_d[0] 29. data_in_d2[96] 30. data_in_d2[64] 31. data_in_d2[32] 32. data_in_d2[0]
1	0 or 1	mrr_data_in[31:0]

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39003b0b5240

Diagram



Fields

Field	Function
31-0 MRR_VL	MRR value See Table 208for an explanation.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 37
DDR General Purpose Registers (DDR_GPR)

37.1 Introduction

DDR_GPR:

- Drives the specific input ports of the DDR module, giving you further configurable options to use the module in different ways
- Helps in capturing some status outputs of the DDR module
- Supports the debugging of the DDR module

37.2 DDR_GPR register descriptions

37.2.1 DDR_GPR memory map

DDR_GPR base address: 4007_C600h

Offset	Register	Width (In bits)	Access	Reset value
0h	Configuration 0 (DDR_Config_0)	32	RW	0000_0000h
4h	Configuration 1 (DDR_Config_1)	32	RW	0000_0000h
8h	Configuration 2 (DDR_Config_2)	32	RW	0000_0000h
Ch	Configuration 3 (DDR_Config_3)	32	RO	0000_0000h
10h	Configuration 4 (DDR_Config_4)	32	RO	0000_0000h
14h	Configuration 5 (DDR_Config_5)	32	RW	0000_0000h
1Ch	LP3 IO Retention Control (DDR_RET_CONTROL)	32	RW	0000_0001h

37.2.2 Configuration 0 (DDR_Config_0)

Offset

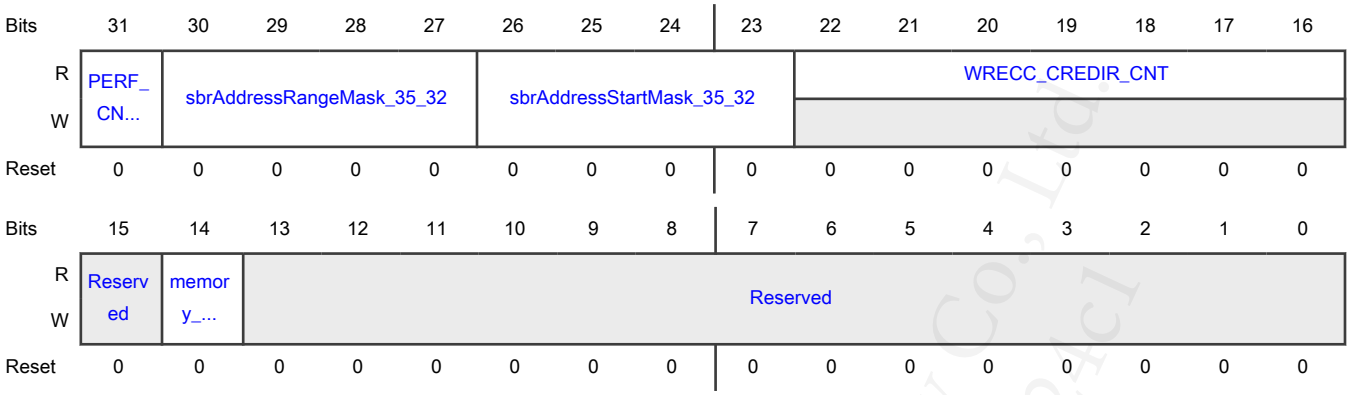
Register	Offset
DDR_Config_0	0h

Function

Reset type = Functional

Provided under non-disclosure agreement only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-3e0d3ab624c1

Diagram



Fields

Field	Function
31 PERF_CNT_FULL_MASK	Monitor Counter Full Interrupt Enables and disables the "monitor counter full" interrupt. 0b - Disable 1b - Enable
30-27 sbrAddressRangeMask_35_32	Scrubber Address Range Mask Bits 35:32 Limits the address range that the ECC scrubber can generate. The full address range mask is 36 bits wide. sbrAddressRangeMask_35_32 contains bits 35:32 of the full mask. DDR_Config_2[sbrAddressRangeMask] contains the remaining bits. For normal scrubber operation, write a 0 to this field. Use other values only for debug purposes. Valid values are 2^n-1, where n=0–33. You must also ensure that the value does not exceed the memory capacity.
26-23 sbrAddressStartMask_35_32	Scrubber Address Start Mask Bits 35:32 Specifies part of the starting address which the ECC scrubber generates. The full scrubber address start mask is 36 bits wide. sbrAddressStartMask_35_32 contains bits 35:32 of the full mask. DDR_Config_1[sbrAddressStartMask] contains the remaining bits. For normal scrubber operation, write a 0 to this field. Use other values only for debug purposes. Valid values are 2^n-1, where n=0–33. You must also ensure that the value: • Does not exceed the memory capacity • Is less than the value defined by the full address range mask
22-16 WRECC_CREDIR_CNT	WRECC Credit Provides the number of available slots (free positions) of write ECC WR content addressable memory (CAM). Each slot holds a DRAM burst synchronous to the DDR core clock (800 MHz). The value is decremented/incremented as commands flow in/out of the write ECC CAM.
15	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
14 memory_ret	Memory Data Retention Controls the retention of memory data in Standby mode. In Standby mode, address and data inputs are disabled. Data stored in the memory is retained, but the memory cannot be accessed for reads or writes. 0b - Do not retain 1b - Retain
13-0 —	Reserved

37.2.3 Configuration 1 (DDR_Config_1)

Offset

Register	Offset
DDR_Config_1	4h

Function

Reset type = Functional

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	sbrAddressStartMask															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	sbrAddressStartMask															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0	Scrubber Address Start Mask Bits 31:0 Specifies part of the starting address which the ECC scrubber generates.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
e8462e5-4821-48a8-b251-590d3ab624c1

Field	Function
sbrAddressStartMask	The full scrubber address start mask is 36 bits wide. sbrAddressStartMask contains bits 31:0 of the full mask. DDR_Config_0[sbrAddressStartMask_35_32] contains the remaining bits. For normal scrubber operation, write a 0 to this field. Use other values only for debug purposes. Valid values are 2^n-1, where n=0–33. You must also ensure that the value: • Does not exceed the memory capacity • Is less than the value defined by the full address range mask

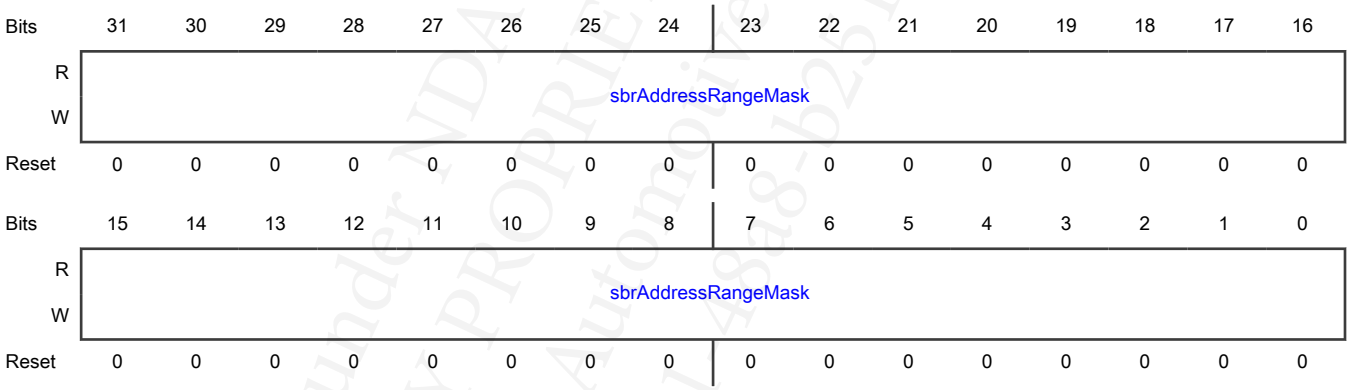
37.2.4 Configuration 2 (DDR_Config_2)

Offset

Register	Offset
DDR_Config_2	8h

Function
Reset type = Functional

Diagram



Fields

Field	Function
31-0 sbrAddressRangeMask	Scrubber Address Range Mask Bits 31:0 Limits the address range that the ECC scrubber can generate. The full address range mask is 36 bits wide. sbrAddressRangeMask contains bits 31:0 of the full mask. DDR_Config_0[sbrAddressRangeMask_35_32] contains the remaining bits. For normal scrubber operation, write a 0 to this field. Use other values only for debug purposes. Valid values are 2^n-1, where n=0–33. You must also ensure that the value does not exceed the memory capacity.

Printed under NDA only
COMPANY PROPRIETARY
BYD Auto Co., Ltd.
CHJ Automotive Technology Co., Ltd.
c6d6e3-4821-4848-b251-390d8ab624c1

37.2.5 Configuration 3 (DDR_Config_3)

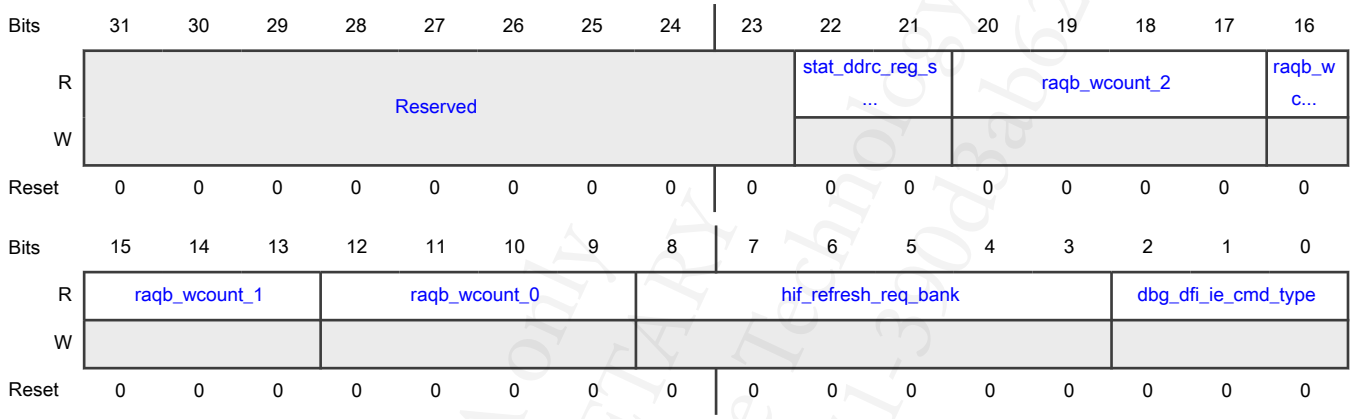
Offset

Register	Offset
DDR_Config_3	Ch

Function

Reset type = Functional

Diagram



Fields

Field	Function
31-23 —	Reserved
22-21 stat_ddrc_reg_selfref_type	DDR Controller 0 Self Refresh Status And Type Mirrors the value of STAT[selfref_type] in the DDR module.
20-17 raqb_wcount_2	Port 2 Read Address Channel Shows the number of used positions in the blue read address FIFO.
16-13 raqb_wcount_1	AXI Port 1 Read Address Channel Shows the number of used positions in the blue read address FIFO.
12-9 raqb_wcount_0	AXI Port 0 Read Address Channel Shows the number of used positions in the blue read address FIFO.
8-3	Refresh Bank Indicates the next bank to refresh.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>hif_refresh_req_bank</td><td>For multi-rank configurations, the bank number is reported independently for each rank and the information for all ranks is concatenated to form this field.</td></tr>
<tr><td>2-0
dbg_dfi_ie_cmd_type</td><td>Debug DFI Command Type
Corresponds to the "Inline ECC Debug" signal of DDR Controller 0.
The value of this field is valid when the DFI command/address is RD/RDA or WR/WRA. Otherwise, the value is irrelevant.
<table>
<tr><th>Command</th><th>dbg_dfi_ie_cmd_type value</th><th>Meaning</th></tr>
<tr><td>RD/RDA</td><td>000b</td><td>RD_N (RD data for non-protected region)</td></tr>
<tr><td></td><td>001b</td><td>RD_E (RD data for protected region)</td></tr>
<tr><td></td><td>010b</td><td>RE_B (RD ECC in block read/write)</td></tr>
<tr><td></td><td>011b</td><td>RE_RMW (RD portion of RMW ECC in block write access)</td></tr>
<tr><td></td><td>111b</td><td>MPR read (DDR4 only) with WR/WRA commands</td></tr>
<tr><td>WR/WRA</td><td>000b</td><td>WD_N (WR data for non-protected region)</td></tr>
<tr><td></td><td>001b</td><td>WD_E (WR data for protected region)</td></tr>
<tr><td></td><td>010b</td><td>WE_BW (WR ECC in block write)</td></tr>
<tr><td></td><td>111b</td><td>MPR write (DDR4 only)</td></tr>
</table>
</td></tr>
</table>

37.2.6 Configuration 4 (DDR_Config_4)

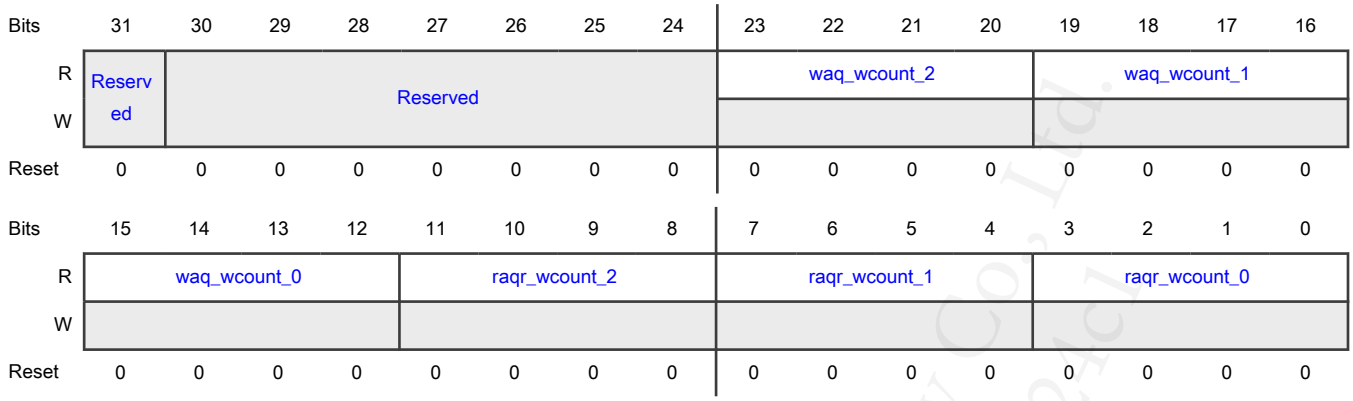
Offset

Register	Offset
DDR_Config_4	10h

Function
Reset type = Functional

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b2f5-3f0b6240

Diagram



Fields

Field	Function
31 —	Reserved
30-24 —	Reserved
23-20 waq_wcount_2	AXI Port 2 Write Address Channel Shows the number of used positions in the write address FIFO.
19-16 waq_wcount_1	AXI Port 1 Write Address Channel Shows the number of used positions in the write address FIFO.
15-12 waq_wcount_0	AXI Port 0 Write Address Channel Shows the number of used positions in the write address FIFO.
11-8 raqr_wcount_2	AXI Port 2 Read Address Channel Shows the number of used positions in the red read address FIFO.
7-4 raqr_wcount_1	AXI Port 1 Read Address Channel Shows the number of used positions in the red read address FIFO.
3-0 raqr_wcount_0	AXI Port 0 Read Address Channel Shows the number of used positions in the red read address FIFO.

Preliminary Draft Under NDA Only
COMPANY PROPRIETARY
Beijing C&J Automotive Technology Co., Ltd.
c8c4625-482a-4a8-b251-390d8ab624c1

37.2.7 Configuration 5 (DDR_Config_5)

Offset

Register	Offset
DDR_Config_5	14h

Function

Contains the mapped safety logic (the RCCU set/clear logic) for AXI inside the DDR subsystem.

Each AXI instance has two RCCUs—main and dummy. Each RCCU compares two data sources (SRC_A and SRC_B) that enter and exit the AXI interface, and sends an alarm under the circumstances described in Figure 188.

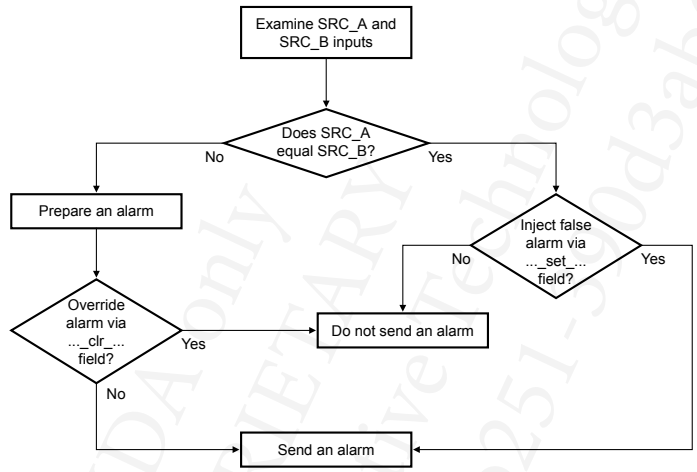


Figure 188. RCCU alarm flow

Reset type = Functional

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved											main_rc...	dummy_r...	main_rc...	dummy_r...	main_rc...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	dummy_r...	main_rc...	dummy_r...	main_rc...	dummy_r...	main_rc...	dummy_r...	Reserved								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHD Automotive Technology Co., Ltd.
c8c46264821748a8-b251-5d0d3ab62dc1

Fields

Field	Function
31-21 —	Reserved
20 main_rccu_clr_axi2	Main RCCU Clear Alarm For AXI 2 Allows you to clear (override) an alarm that the main RCCU produced for AXI 2. See Figure 188. 0b - Do not clear 1b - Clear
19 dummy_rccu_clr_axi2	Dummy RCCU Clear Alarm For AXI 2 Allows you to clear (override) an alarm that the dummy RCCU produced for AXI 2. See Figure 188. 0b - Do not clear 1b - Clear
18 main_rccu_set_axi2	Main RCCU Set Alarm For AXI 2 Allows you to set (inject) a false alarm into the main RCCU for AXI 2. See Figure 188. 0b - Do not set 1b - Set
17 dummy_rccu_set_axi2	Dummy RCCU Set Alarm For AXI 2 Allows you to set (inject) a false alarm into the dummy RCCU for AXI 2. See Figure 188. 0b - Do not set 1b - Set
16 main_rccu_clr_axi1	Main RCCU Clear Alarm For AXI 1 Allows you to clear (override) an alarm that the main RCCU produced for AXI 1. See Figure 188. 0b - Do not clear 1b - Clear
15 dummy_rccu_clr_axi1	Dummy RCCU Clear Alarm For AXI 1 Allows you to clear (override) an alarm that the dummy RCCU produced for AXI 1. See Figure 188. 0b - Do not clear 1b - Clear
14 main_rccu_set_axi1	Main RCCU Set Alarm For AXI 1 Allows you to set (inject) a false alarm into the main RCCU for AXI 1. See Figure 188. 0b - Do not set 1b - Set
13	Dummy RCCU Set Alarm For AXI 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
dummy_rccu_set_axi1	Allows you to set (inject) a false alarm into the dummy RCCU for AXI 1. See Figure 188. 0b - Do not set 1b - Set
12 main_rccu_clr_axi0	Main RCCU Clear Alarm For AXI 0 Allows you to clear (override) an alarm that the main RCCU produced for AXI 0. See Figure 188. 0b - Do not clear 1b - Clear
11 dummy_rccu_clr_axi0	Dummy RCCU Clear Alarm For AXI 0 Allows you to clear (override) an alarm that the dummy RCCU produced for AXI 0. See Figure 188. 0b - Do not clear 1b - Clear
10 main_rccu_set_axi0	Main RCCU Set Alarm For AXI 0 Allows you to set (inject) a false alarm into the main RCCU for AXI 0. See Figure 188. 0b - Do not set 1b - Set
9 dummy_rccu_set_axi0	Dummy RCCU Set Alarm For AXI 0 Allows you to set (inject) a false alarm into the dummy RCCU for AXI 0. See Figure 188. 0b - Do not set 1b - Set
8-0 —	Reserved

37.2.8 LP3 IO Retention Control (DDR_RET_CONTROL)

Offset

Register	Offset
DDR_RET_CONTROL	1Ch

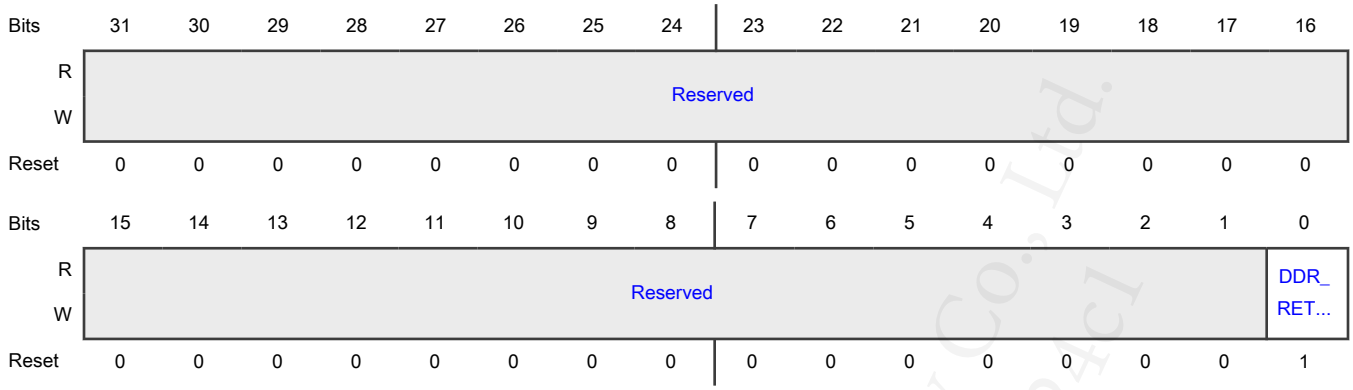
Function

Controls the DDR LP3 IO retention before asserting the software functional reset through MC_ME.

Reset type = Functional

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-8251-590d3ab6240

Diagram



Fields

Field	Function
31-1 —	Reserved
0 DDR_RET_CONTROL	DDR LP3 IO Retention Control Enables and disables DDR LP3 IO retention. Write a 0 to this field immediately before asserting software functional reset through MC_ME to enable the retention. 0b - Enabled 1b - Disabled

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 38
Quad Serial Peripheral Interface (QuadSPI)

38.1 Chip-specific QuadSPI configuration

38.1.1 Overview

The QuadSPI module and associated PHY provide a common memory interface to single, dual, quad, and octal NOR flash memory, including Hyperflash memory. The primary use case of the QuadSPI is for interfacing with external flash memory and octal PSRAMs. The flash memory interface supports the following external NOR flash memories:

- 8-bit wide bus (x8) DDR NOR, OctaFlash™, XTRM, and HyperFlash™ flash memories
- 4-bit wide bus (x4) SDR and DDR quad NOR flash memories
- Cypress QuadFlash memory
- Octal PSRAM memory

For QuadSPI pin-mapping, see the device 'Input Muxing' tab in the IOMUX spreadsheet attached to this document.

For the target frequencies, see the chip data sheet.

38.1.2 Supported modes

The table below provides an overview of the QuadSPI modes supported.

Table 209. QuadSPI modes supported

Frequency (MHz)	DDR / SDR	Slave delay chain Setting (DLL)	Slave delay chain Setting (Tap selection)	DQS Setting	Flash type	Flash interface
66	DDR	ON	AUTO	Pad Loopback (dummy)	Quad Flash (1.8 V)	QuadSPI flash A interface QuadSPI flash B interface
66	DDR	ON	AUTO	Pad Loopback (dummy)	Quad Flash (3.3 V)	QuadSPI flash B interface
104	SDR	OFF	BYPASS	Pad Loopback (dummy)	Quad Flash (3.3 V)	QuadSPI flash B interface
133, 100, 40	SDR	OFF	BYPASS	Pad Loopback (dummy)	Octal Flash (1.8 V)	QuadSPI flash A interface QuadSPI flash B interface
133	SDR	OFF	BYPASS	Pad Loopback (dummy)	Quad Flash (1.8 V)	QuadSPI flash A interface QuadSPI flash B interface

Table continues on the next page...

Table 209. QuadSPI modes supported (continued)

Frequency (MHz)	DDR / SDR	Slave delay chain Setting (DLL)	Slave delay chain Setting (Tap selection)	DQS Setting	Flash type	Flash interface
200, 166, 133, 100	DDR	ON	MANUAL	External DQS (Edge-aligned)	Octal Flash (1.8 V)	100/133 are supported on both A and B interface. 166/200 are supported on flash A interface only.

38.1.3 QuadSPI clocking

The ratio of bus clock to flash clock must not be less than 1:5. The ratio of bus clock to internal reference clock must not be less than that derived from the formula below. For more details, see the chip data sheet.

The flash read timing must meet the following condition:

The total time period of the number of clock cycles required to fetch the next *n* bits of data (where *n* can be 64, 32, or 16) from flash memory must be greater than two bus clock periods.

Derive DDR octal writes using the following equation:

(3 x flash clock period) + (3 x bus clock period) < (8 x flash clock period)

38.1.4 QuadSPI limitations

The implementation of QuadSPI in this chip has the following limitations:

- This chip does not support WRAP* transactions for AHB read.
- Any speculative access of the QuadSPI memory mapped region 00000000h-20000000h before you configure QuadSPI might result in unexpected chip behavior. Therefore, in this situation, you must use core programming to disable speculative access of that region.
- QuadSPI AHB region 00000000h-1FFFFFFF is non-cacheable.
- It is required that the application must rely on the FCCU fault for the ECC error handling by writing '0' to the QuadSPI RSER[CRCBIE] and RSER[CRCAIE]. In cases where FCCU is not configured (for example, in the customer bootloader), these fields must be enabled.

38.1.5 Chip-specific feature configuration

QuadSPI instance on this chip does not support Peripheral chip select (PCS). Disregard its references in this chapter.

QuadSPI instance on this chip supports Auto-learning feature at 66MHz only. For more details, see the chip Data Sheet.

NOTE

Reset (flush) the QSPI AHB buffer and the IPS domain before the first AHB read transaction, or if you reconfigure them.

38.2 Introduction

The QuadSPI module acts as an interface to a single or two external SFM devices, each with up to eight bidirectional data lines.

38.2.1 Features

QuadSPI supports the following features:

- Flexible sequence engine to support various flash memory vendor devices. As there is no specific standard, the module supports various kinds of flash memories from different vendors. See Serial flash memory devices for example sequences.
- Single, dual, and quad modes of operation supported for Quad flash memories
- Octal and single IO modes of operation supported for Octal flash memories
- Double Data Rate (DDR)/Double Transfer Rate (DTR) mode in which the data is generated on every edge of the serial flash memory clock
- Flash memory data strobe signal to support data sampling in DDR and Single Data Rate (SDR) mode
- Support for HyperFlash memory
- Support for Macronix Octal Data integrity features such as ECC and R/W parity (CRC1)
- Support for parallel writes through register-mapped interface in single I/O mode
- Ability to connect two identical serial flash memory devices and access them simultaneously for data read operations, forming one (virtual) flash memory with doubled readout bandwidth
- AHB master to read RX buffer data through AMBA AHB (64-bit width interface) or IPS registers space (32-bit access) and fill TX buffer via IPS registers space (32-bit access)
  - AHB master can be a DMA with a configurable inner loop size
- Multi-master accesses are allowed, but only a single master access is supported with On-the-fly AES decryption (OTFAD)
  - Flexible and configurable buffer for each master—total available buffer size is 1024 bytes.
- All AHB accesses to flash memory devices are directly memory mapped to the chip system memory
- Programmable sequence engine to cater to future command/protocol changes and ability to support all existing vendor commands and operations. The software needs to select the corresponding sequence according to the connected flash memory device.
  - Support for all types of addressing
- Support for interaction with OTFAD

38.2.2 RX buffer push event

To add the valid entries into the RX buffer

By default, each buffer push event adds two entries to the RX buffer because the interface to the serial clock domain is 64 bits in width. Depending on the number of bytes read from the serial flash memory device, it is possible for the very last buffer push event that only one entry is added.

RBSR[RDBFL] is incremented by the number of entries added to the RX buffer.

38.2.3 RX buffer POP event

To remove valid entries from the RX buffer

Each buffer POP event removes (RBCT[WMRK] + 1) valid entries from the buffer. BSR[RDBFL] is decremented by the same number and RBSR[RDCTR] is incremented accordingly.

38.2.4 Block diagram

The following figure shows a block diagram of the QuadSPI module.

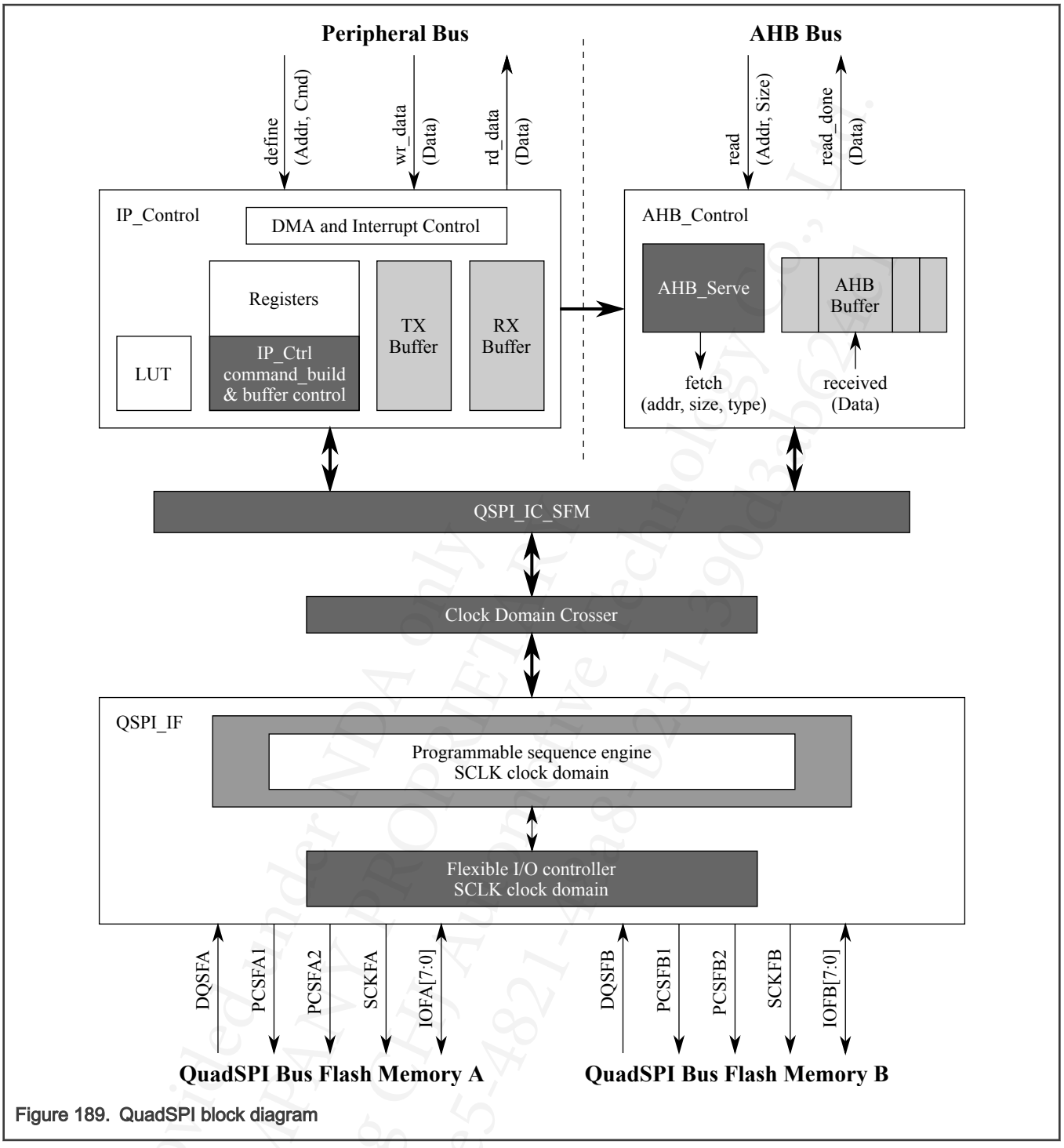


Figure 189. QuadSPI block diagram

38.2.5 QuadSPI modes of operation

QuadSPI supports the following modes of operation:

- Normal mode: You can use this mode for write or read accesses to an external serial flash memory device. See Normal mode for details.
  - Serial flash memory write: You can program data into the flash memory through the IP interface only. See Flash memory programming for details.

— Serial flash memory read: Read the contents of the serial flash memory device. Two separate read channels are available through the RX buffer and AHB buffer. See Flash memory read for details.

- Module Disable mode: You can use this mode for disabling serial flash memory clock and AHB command. The clock to the non-memory mapped logic in QuadSPI can be stopped in the Module Disable mode. The module enters the mode by setting MCR[MDIS].

38.3 External signal description

This section provides the external signal information for the QuadSPI module.

The following table lists the external signals belonging to the module in conjunction with the different modes of operation.

Table 210. Signal properties

Signal name	Function	Direction	Description
PCSFA1	Peripheral Chip Select Flash Memory A1	O	This signal is the chip select for the serial flash memory device A1 that represents the first device in a dual-die package flash memory A or the first of the two flash memory devices that share IOFA. See Dual-die flash memories for details.
PCSFA2	Peripheral Chip Select Flash Memory A2	O	This signal is the chip select for the serial flash memory device A2 that represents the second device in a dual-die package flash memory A or the second of the two flash memory devices that share IOFA. See Dual-die flash memories for details.
PCSFB1	Peripheral Chip Select Flash memory B1	O	This signal is the chip select for the serial flash memory device B1 that represents the first device in a dual-die package flash memory B or the first of the two flash memory devices that share IOFB. See Dual-die flash memories for details.
PCSFB2	Peripheral Chip Select Flash Memory B2	O	This signal is the chip select for the serial flash memory device B2 that represents the second device in a dual-die package flash memory B or the second of the two flash memory devices that share IOFB. See Dual-die flash memories for details.
SCKFA	Serial Clock Flash Memory A	O	This signal is the serial clock output to the serial flash memory device A.
SCKFB	Serial Clock Flash B	O	This signal is the serial clock output to the serial flash memory device B.
IOFA[7:0]	Serial I/O Flash memory A	I/O	These signals are the data I/O lines to/from the serial flash memory device A. See Driving external signals for details about the signal drive and timing behavior. Note that the signal pins of the serial flash memory device may change their function according to the SFM Command executed, leaving them as control inputs when single and dual instructions are executed. The module supports driving these inputs to dedicated values. In single I/O mode, QuadSPI drives data on IOFA[0] and expects data on IOFA[1].
IOFB[7:0]	Serial I/O Flash Memory B	I/O	These signals are the data I/O lines to/from the serial flash memory device B. See Driving external signals for details about the signal drive and timing behavior. Note that the signal pins of the serial flash memory device may change their function according to the SFM command executed, leaving

Table continues on the next page...

Table 210. Signal properties (continued)

Signal name	Function	Direction	Description
			them as control inputs when single and dual instructions are executed. The module supports driving these inputs to dedicated values. In single I/O mode, QuadSPI drives data on IOFB[0] and expects data on IOFB[1].
DQSFA	Data Strobe signal Flash Memory A	I	This is the data strobe signal for port A. Some flash memory vendors provide the Data Read Strobe (DQS) signal to which the read data is aligned in DDR mode.
DQSFB	Data Strobe signal Flash Memory B	I	This is the data strobe signal for port B. Some flash memory vendors provide the DQS signal to which the read data is aligned in DDR mode.
INTA	ECC error signal for Flash Memory A	I	Flash Memory A drives this signal to active low value in case of an ECC error.
INTB	ECC error signal for Flash Memory B	I	Flash Memory B drives this signal to active low value in case of an ECC error.

38.3.1 Driving external signals

Single/dual/quad/octal instructions

Depending on the serial flash memory device connected to the QuadSPI module, there are instructions using a different number of data lines:

- Single pad: Single line I/O with one data out and one data in line to/from the serial flash memory device
- Dual pad: Dual line I/O with two bidirectional I/O lines, driven alternatively by the serial flash memory device or the QuadSPI module
- Quad pad: Quad line I/O with four bidirectional I/O lines, driven alternatively by the serial flash memory device or the QuadSPImodule
- Octal pad: Octal line I/O with eight bidirectional I/O lines, driven alternatively by the serial flash memory device or the QuadSPI module

The different phases of the serial flash memory access scheme are shown in the following figure.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48b2-b213-90d3a6b4c

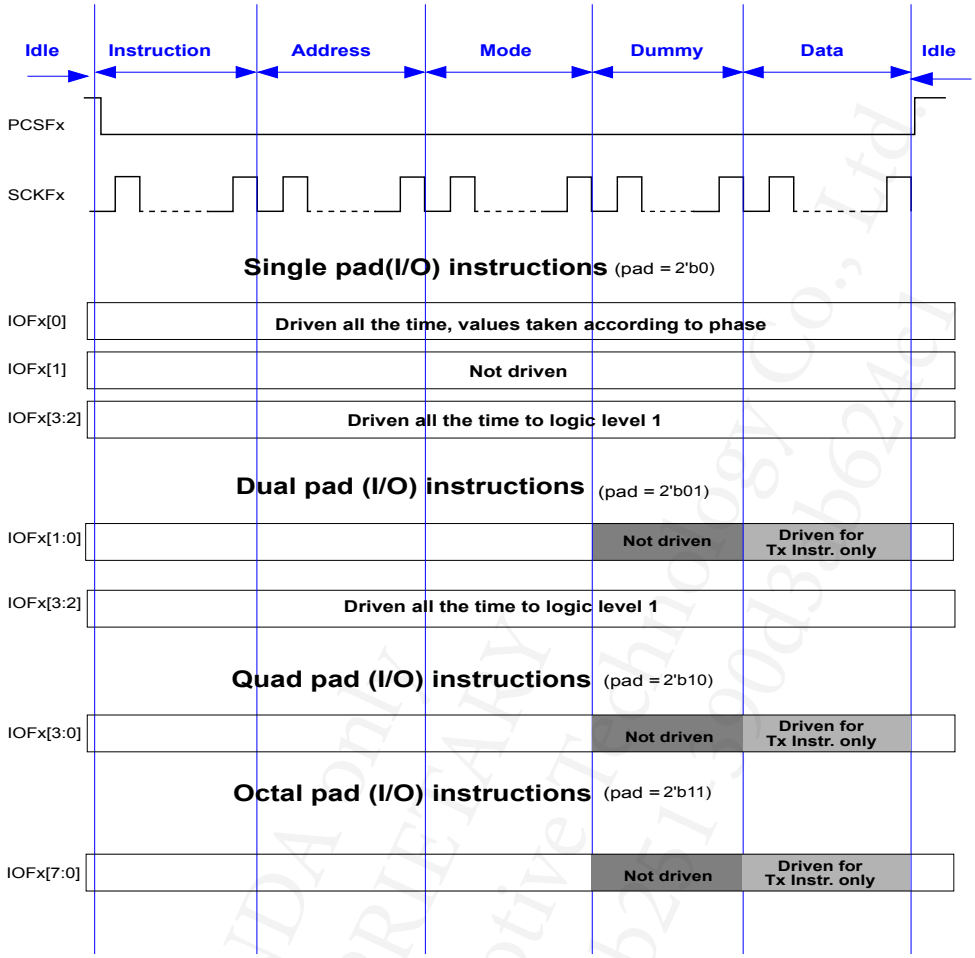


Figure 190. Serial flash memory access scheme

Following are the different phases and the I/O driving characteristics of the QuadSPI module:

- Idle: Serial flash memory device not selected – no interaction with the serial flash memory device and the IOFx signals are driven.
- Instruction: Serial flash memory device selected – the instruction is sent to the serial device and all the IOFx signals are driven.
- Address: Serial flash memory address is sent to the device – all the IOFx signals are driven and this phase is not applicable for all SFM commands.
- Mode: Mode bytes are sent to the serial flash memory device – all the IOFx signals are driven and this phase is not applicable for all SFM commands.
- Dummy: Dummy clocks are provided to the serial flash memory device. See Figure 190 for the IOFx signals driven. The actual data lines required for the SFM command executed are not driven for data read commands.

NOTE

— This phase is not applicable for all the SFM commands.

— All read commands in Dual pad, Quad pad or Octal pad modes must use dummy phase before read phase. Note that this restriction is not applicable to Single-pad mode.

- Data: Serial flash memory data are sent to or received from the serial flash memory device. See the preceding figure for the IOFx signals driven. The actual data lines required for the SFM command executed are not driven for data read commands.

NOTE

This phase is not applicable for all the SFM commands.

The PCSFx and SCKFx signals are driven permanently throughout all the phases. In the individual flash memory mode, this applies to the selected flash memory device. In the parallel flash memory mode, this applies to both the serial flash memory devices simultaneously.

Access to a single, individual serial flash memory device

See Serial flash memory access schemes for details.

Read access to two serial flash memory devices attached to the QuadSPI module in parallel. See Serial flash memory access schemes for details.

38.4 Memory map and register definition

This section provides the memory map and register definitions for the QuadSPI module.

38.4.1 Register write access

Following are the write access restriction terms that apply to all the registers:

- Register write access restriction

  For each register field, the write access conditions are specified in the detailed register description.

  The following table provides a description of the write access conditions. If, for a specific register bit or field, none of the given write access conditions is fulfilled, any write attempt to this register bit or field is ignored without any notification. The values of the bits or fields are not changed.

  The condition term [A or B] indicates that the register or field can be written to if at least one of the conditions is fulfilled.

Table 211. Register write access restrictions

Condition	Description
Anytime	No write access restriction
Disabled mode	Write access only if MCR[MDIS] = 1
Normal mode	Write access only if the module is in the normal mode

- Register write access requirements

  You can access all registers using 8-bit, 16-bit, and 32-bit wide operations. For some of the registers, at least a 16-bit or 32-bit wide write access is required to ensure correct operation. This write access requirement is stated in the detailed register description for each affected register.

38.4.2 QuadSPI register descriptions

This section provides the memory map and register definitions for the QuadSPI module.

Access to the following addresses does not result in a transfer error:

- 4h
- 24h
- 168h

38.4.2.1 QuadSPI memory map

QuadSPI base address: 4013_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration Register (MCR)	32	RW	000F_404Ch
8h	IP Configuration Register (IPCR)	32	RW	0000_0000h
Ch	Flash Memory Configuration Register (FLSHCR)	32	RW	0000_0303h
10h	Buffer 0 Configuration Register (BUF0CR)	32	RW	0000_000Bh
14h	Buffer 1 Configuration Register (BUF1CR)	32	RW	0000_0001h
18h	Buffer 2 Configuration Register (BUF2CR)	32	RW	0000_0002h
1Ch	Buffer 3 Configuration Register (BUF3CR)	32	RW	8000_0003h
20h	Buffer Generic Configuration Register (BFGENCR)	32	RW	0000_0000h
30h	Buffer 0 Top Index Register (BUF0IND)	32	RW	0000_0000h
34h	Buffer 1 Top Index Register (BUF1IND)	32	RW	0000_0000h
38h	Buffer 2 Top Index Register (BUF2IND)	32	RW	0000_0000h
50h	AHB Write Configuration Register (AWRCR)	32	RW	0000_0000h
60h	DLL Flash Memory A Configuration Register (DLLCRA)	32	RW	0120_0000h
64h	DLL Flash Memory B Configuration Register (DLLCRB)	32	RW	0120_0000h
6Ch	Parity Configuration Register (PARITYCR)	32	RW	0000_0000h
100h	Serial Flash Memory Address Register (SFAR)	32	RW	0000_0000h
104h	Serial Flash Memory Address Configuration Register (SFACR)	32	RW	0000_0800h
108h	Sampling Register (SMPR)	32	RW	0000_0000h
10Ch	RX Buffer Status Register (RBSR)	32	RO	0000_0000h
110h	RX Buffer Control Register (RBCT)	32	RW	0000_0000h
120h	AHB Write Status Register (AWRSR)	32	RO	0000_0000h
12Ch	DLL Status Register (DLLSR)	32	RO	8000_8000h
130h	Data Learning Configuration Register (DLCR)	32	RW	40FF_40FFh
134h	Data Learning Status Flash Memory A Register (DLSR_FA)	32	RO	0000_0000h
138h	Data Learning Status Flash Memory B Register (DLSR_FB)	32	RO	0000_0000h
150h	TX Buffer Status Register (TBSR)	32	RO	0000_0000h
154h	TX Buffer Data Register (TBDR)	32	RW	0000_0000h
158h	TX Buffer Control Register (TBCT)	32	RW	0000_0000h
15Ch	Status Register (SR)	32	RO	0200_3800h
160h	Flag Register (FR)	32	W1C	0800_0000h
164h	Interrupt and DMA Request Select and Enable Register (RSER)	32	RW	0000_0000h
16Ch	Sequence Pointer Clear Register (SPTRCLR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
180h	Serial Flash Memory A1 Top Address Register (SFA1AD)	32	RW	0000_0000h
184h	Serial Flash Memory A2 Top Address Register (SFA2AD)	32	RW	0000_0000h
188h	Serial Flash Memory B1 Top Address Register (SFB1AD)	32	RW	0000_0000h
18Ch	Serial Flash Memory B2 Top Address Register (SFB2AD)	32	RW	0000_0000h
190h	Data Learn Pattern Register (DLPR)	32	RW	AA55_3443h
194h	Flash Memory A Failing Address Status Register (FAILA_ADDR)	32	RO	FFFF_FFFFh
198h	flash Memory B Failing Address Status Register (FAILB_ADDR)	32	RO	FFFF_FFFFh
200h - 2FCh	RX Buffer Data Register (RBDR0 - RBDR63)	32	RO	0000_0000h
300h	LUT Key Register (LUTKEY)	32	RW	5AF0_5AF0h
304h	LUT Lock Configuration Register (LCKCR)	32	RW	0000_0002h
310h	LUT Register (LUT0)	32	RW	0818_0403h
314h	LUT Register (LUT1)	32	RW	2400_1C08h
318h - 44Ch	LUT Register (LUT2 - LUT79)	32	RW	0000_0000h

38.4.2.2 Module Configuration Register (MCR)

Offset

Register	Offset
MCR	0h

Function

This register holds configuration data associated with the QuadSPI operation.

NOTE

When out of reset, after first initial MCR programming and exiting module disable mode (that is, when the value of MCR[MDIS] is 0), you must write 1 to MCR[SWRSTSD] and release it after six (three system and three flash memory) clock cycles.

Special write-access is permitted in different modes:

- CK2_DCARS_FB, CKN_FB_EN, DQS_FB_SEL, CK2_DCARS_FA, CKN_FA_EN, DQS_FA_SEL: Disabled mode
- ISD3FB, ISD2FB, ISD3FA, ISD2FA: Disabled mode
- All other fields: Anytime

Printed under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co.,Ltd.
c8c4e2e5-e9_48a8-b251-39

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CK2_DCA...	CKN_FB_...	DQS_FB_SEL		CK2_DCA...	CKN_FA_...	DQS_FA_SEL		Reserved	0	0		ISD3FB	ISD2FB	ISD3FA	ISD2FA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	MDIS	Reserved	DLPEN	0	0	0	Reserved	DDR_EN	DQS_EN	DQS_LAT...	Reserved	Reserved		SWRSTHD	SWRSTSD
W					CLR_TXF	CLR_RXF										
Reset	0	1	0	0	0	0	0	0	0	1	0	0	1	1	0	0

Fields

Field	Function
31 CK2_DCARS_FB	CK2 DDR center-aligned read strobe for flash memory B Write 1 to this field to enable CK2 output 90-degree phase shifted clock. **NOTE** Program this field as 0 in the following cases: • Write transactions for DCARS flash memory • Always for Non-DCARS flash memory
30 CKN_FB_EN	Enable clock on differential CKN pad of flash memory B Set this field to '1' to enable clock on differential CKN pad for flash memory B.
29-28 DQS_FB_SEL	DQS clock for sampling read data at flash memory B Selects DQS clock for sampling read data at flash memory B QuadSPI port 00b - Reserved 01b - Pad loopback 10b - Reserved 11b - External DQS **NOTE** In case of an internal DQS/padloopback selection to access port B, port A cannot be programmed for external DQS and vice-versa.
27 CK2_DCARS_FA	CK2 DDR center-aligned read strobe for flash memory A Write 1 to this field to enable CK2 output 90-degree phase shifted clock.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE Program this field as 0 in the following cases: • Write transactions for DCARS flash memory • Non-DCARS flash memory
26 CKN_FA_EN	CKN pad for flash memory A Set 1 to enable clock on differential CKN pad for flash memory A.
25-24 DQS_FA_SEL	DQS clock for sampling read data at flash memory A Selects DQS clock for sampling read data at flash memory A QuadSPI port 00b - Reserved 01b - Pad loopback 10b - Reserved 11b - External DQS **NOTE** In case of an internal DQS/padloopback selection to access port A, port B cannot be programmed for external DQS and vice-versa.
23 —	Reserved
22 —	Reserved
21-20 —	Reserved
19 ISD3FB	Idle signal drive IOFB[3] flash memory B This field determines the logic level that the IOFB[3] output of the QuadSPI module is driven to in the inactive state. See Driving flash memory control signals in single and dual modes. 0b - IOFB[3] is driven to logic L. 1b - IOFB[3] is driven to logic H.
18 ISD2FB	Idle signal drive IOFB[2] flash memory B This field determines the logic level that the IOFB[2] output of the QuadSPI module is driven to in the inactive state. See Driving flash memory control signals in single and dual modes. 0b - IOFB[2] is driven to logic L. 1b - IOFB[2] is driven to logic H.
17	Idle signal drive IOFA[3] flash memory A

Table continues on the next page...

Table continued from the previous page...

Field	Function
ISD3FA	Determines the logic level that the IOFA[3] output of the QuadSPI module is driven to in the inactive state. See Driving flash memory control signals in single and dual modes for details. 0b - IOFA[3] is driven to logic L 1b - IOFA[3] is driven to logic H
16 ISD2FA	Idle signal drive IOFA[2] flash memory A Determines the logic level that the IOFA[2] output of the QuadSPI module is driven to in the inactive state. See Driving flash memory control signals in single and dual modes for details. 0b - IOFA[2] is driven to logic L. 1b - IOFA[2] is driven to logic H.
15 —	Reserved
14 MDIS	Module disable Allows the clock to the non-memory mapped logic in the QuadSPI to be stopped. 0b - Enable QuadSPI clocks 1b - Allow external logic to disable QuadSPI clocks
13 —	Reserved
12 DLPEN	Data learning pattern enable Write 1 to this field to enable data learning mechanism.
11 CLR_TXF	Clear TX FIFO/buffer This is a self-clearing field that invalidates the TX buffer content. 0b - No action 1b - Read and write pointers of the TX buffer are reset to 0 and TBSR[TRCTR] is reset to 0.
10 CLR_RXF	Clear RX FIFO This is a self-clearing field that invalidates the RX buffer content. 0b - No action 1b - Read and write pointers of the RX buffer are reset to 0 and RBSR[RDBFL] is reset to 0.
9 —	Reserved
8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 DDR_EN	DDR mode enable Enables the DDR mode 0b - 2x clock disabled for SDR instructions only 1b - 2x clock enabled for both the SDR and DDR instructions
6 DQS_EN	DQS enable Is valid for both the SDR and DDR modes. For details, see DQS sampling method. 0b - Reserved 1b - The incoming data is sampled on both the edges of the DQS input when the value of MCR[DDR_EN] is 1; else, on only one edge when MCR[DDR_EN] is 0.
5 DQS_LAT_EN	DQS latency enable Is valid when latency is included between read accesses from flash memory in DQS sampling. For details, see DQS sampling method. 0b - DQS latency is disabled. 1b - DQS feature with latency included is enabled.
4 —	Reserved
3-2 —	Reserved
1 SWRSTHD	Software reset for AHB domain 0b - No action 1b - AHB domain flops are reset. This field does not reset configuration registers. It is advisable to reset both the serial flash memory domain and AHB domain at the same time. Resetting only one domain might lead to side effects. **NOTE** The software resets need the clock to be running to propagate to the design. The value of MCR[MDIS] should be 0 when the software reset bits are asserted. Also, before they can be deasserted again (by setting MCR[SWRSTHD] to 0), it is recommended to set the value of MCR[MDIS] to 1. After the software resets have been deasserted, the normal operation can be started by setting MCR[MDIS] to 0.
0 SWRSTSD	Software reset for serial flash memory domain 0b - No action

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Serial flash memory domain flops are reset. This field does not reset configuration registers. It is advisable to reset both the serial flash memory domain and AHB domain at the same time. Resetting only one domain might lead to side effects. **NOTE** The software resets need the clock to be running to propagate to the design. The value of MCR[MDIS] should therefore be 0 when the software reset bits are asserted. Also, before they can be deasserted again (by specifying 0 as the value for MCR[SWRSTSD]), it is recommended to specify 1 as the value for MCR[MDIS]. After the software resets are deasserted, the normal operation can be started by specifying 0 as the value for MCR[MDIS].

38.4.2.3 IP Configuration Register (IPCR)

Offset

Register	Offset
IPCR	8h

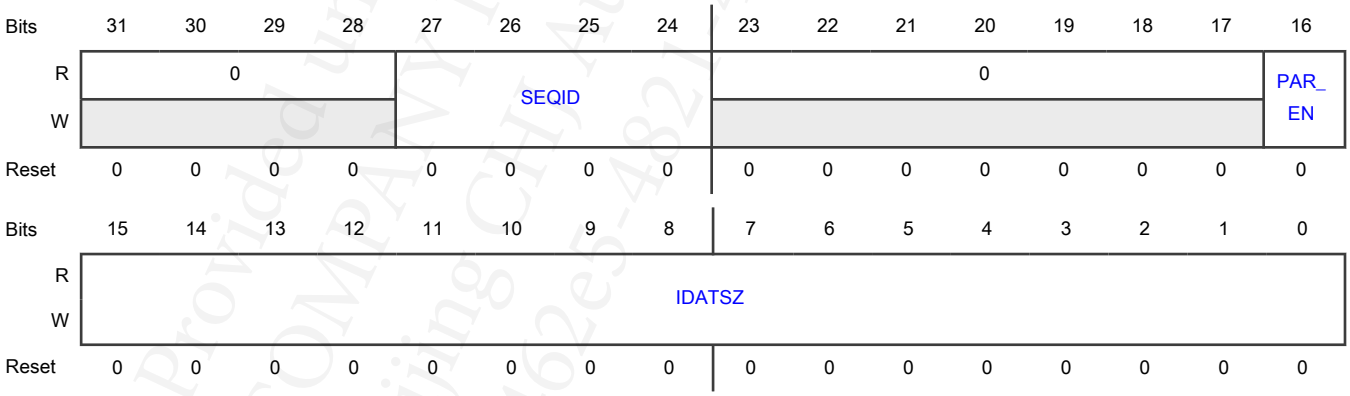
Function

This register provides all the configuration required for an IP-initiated command, which can be triggered by writing in the SEQID field of this register. If the SEQID field is written successfully, a new command to the external serial flash memory is initiated per the sequence pointed to by this field. See Normal mode for details on command triggering and command execution.

Special write-access is permitted if:

- SR[IP_ACC]=0

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab62

Fields

Field	Function
31-28 —	Reserved
27-24 SEQID	Points to a sequence in the LUT This field defines bits [6:2] of the LUT index. The bits [1:0] are always assumed to be 0. See LUT for details. A write to this field triggers a transaction on the serial flash memory interface.
23-17 —	Reserved
16 PAR_EN	If the value of this field is 1, a transaction to two serial flash memory devices is triggered in the parallel mode. See Parallel flash memory mode for details.
15-0 IDATSZ	IP data transfer size This field defines the data transfer size, in bytes, of the IP command.

38.4.2.4 Flash Memory Configuration Register (FLSHCR)

Offset

Register	Offset
FLSHCR	Ch

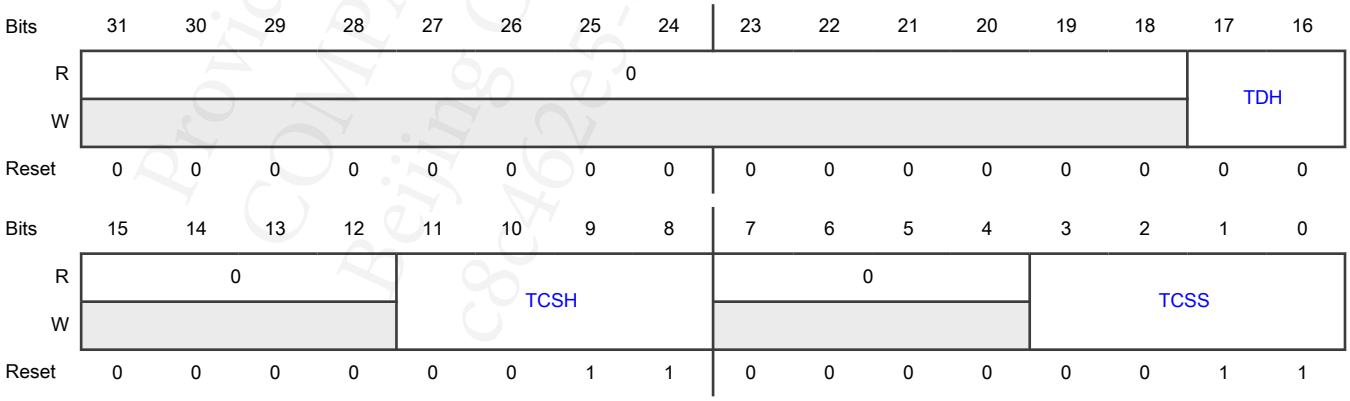
Function

This register contains the timings that are specific to the flash memory device. The QuadSPI controller must meet these timings for the device to function correctly.

Special write-access is permitted if:

- SR[AHB_ACC] = 0
- SR[IP_ACC] = 0

Diagram



Fields

Field	Function
31-18 —	Reserved
17-16 TDH	Serial flash memory data in hold time This field helps in meeting the data in hold time requirement of a flash memory. It is valid only in the DDR mode. **NOTE** This field should be set to 0x00 in the SDR mode (MCR[DDR_EN]=0). See Data input hold requirement of flash memory for details. The valid TDH programming options are shown below and the other combinations are invalid. 00b - Data aligned with the posedge of internal reference clock of QuadSPI 01b - Data aligned with 2x serial flash memory half clock
15-12 —	Reserved
11-8 TCSH	Serial flash memory CS hold time This hold time is in terms of serial flash memory clock cycles, and it must be greater than or equal to three flash memory clock cycles .
7-4 —	Reserved
3-0 TCSS	Serial flash memory CS setup time This setup time is in terms of serial flash memory clock cycles, and it must be greater than or equal to two flash memory clock cycles.

38.4.2.5 Buffer 0 Configuration Register (BUF0CR)

Offset

Register	Offset
BUF0CR	10h

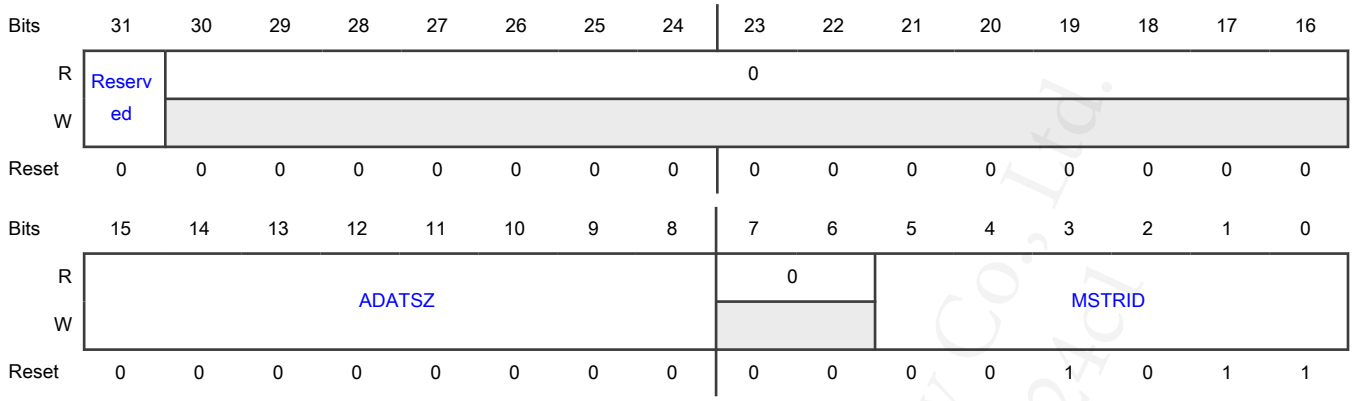
Function

This register provides the configuration for any read access routed to buffer0, which happens when the master port number of the incoming AHB request matches BUF0CR[MSTRID]. Any buffer "miss" leads to a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31 —	Reserved
30-16 —	Reserved
15-8 ADATSZ	AHB data transfer size Defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. For example, a value of 0x2 sets transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. The software should ensure that this transfer size is not greater than the size of the buffer. If OTFAD is enabled, the QuadSPI module prefetches only that much data from flash memory such that the prefetch never crosses the 1 KB boundary.
7-6 —	Reserved
5-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 0 Any AHB read access with this master port number is routed to this buffer. You must ensure that the master IDs associated with all buffers are different.

38.4.2.6 Buffer 1 Configuration Register (BUF1CR)

Offset

Register	Offset
BUF1CR	14h

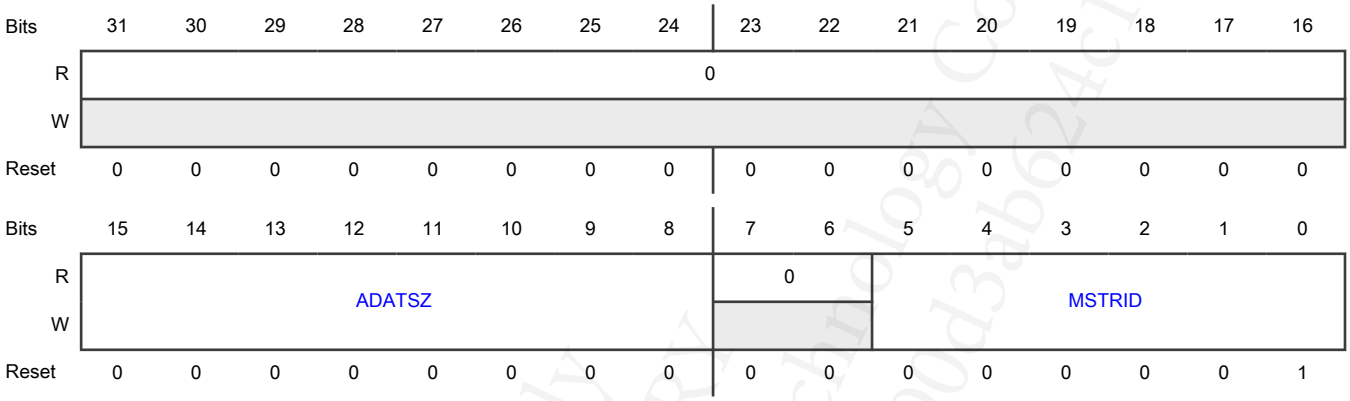
Function

This register provides the configuration for any access routed to buffer 1, which happens when the master port number of the incoming AHB request matches the MSTRID field of this register. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 ADATSZ	AHB data transfer size This field defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. For example, a value of 0x2 sets the transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. Software should ensure that this transfer size is not greater than the size of this buffer. If OTFAD is enabled, the QuadSPI module prefetches only the required data from the flash memory so that the prefetch never crosses the 1 KB boundary.
7-6 —	Reserved
5-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 1 Any AHB read access with this master port number is routed to this buffer. You must ensure that the master IDs associated with all buffers are different.

Provided by NXP under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e65-4821-4e8-b251-390d3db624c1

38.4.2.7 Buffer 2 Configuration Register (BUF2CR)

Offset

Register	Offset
BUF2CR	18h

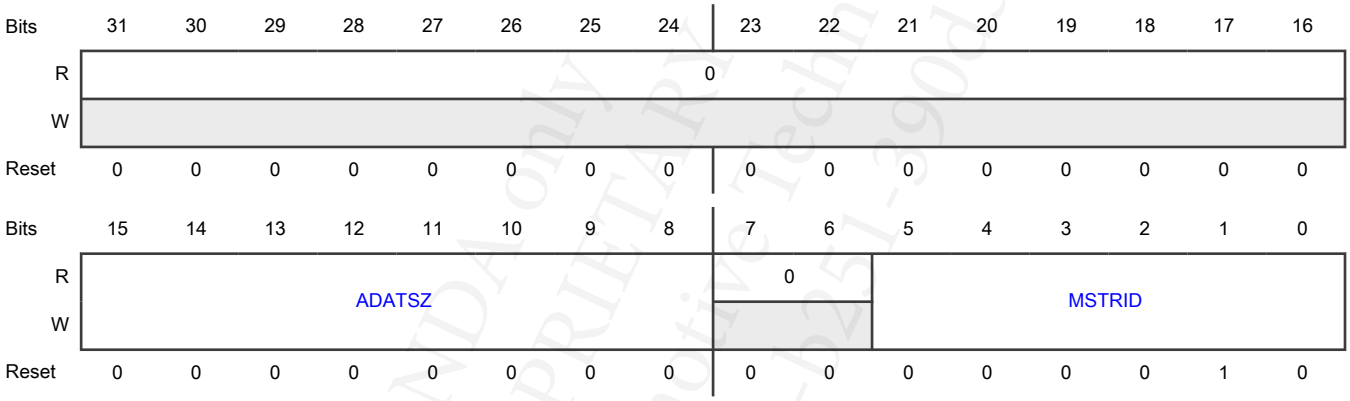
Function

This register provides the configuration for any access routed to buffer 2, which happens when the master port number of the incoming AHB request matches the MSTRID field of this register. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 ADATSZ	AHB data transfer size This field defines the read data transfer size in 8 bytes of an AHB triggered read access to the serial flash memory. For example, a value of 0x2 sets transfer size to 16 bytes. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. The software should ensure that this transfer size is not greater than the size of this buffer. If OTFAD is enabled, the QuadSPI module prefetches only the required data from the flash memory such that the prefetch never crosses the 1 KB boundary.
7-6 —	Reserved
5-0	Master ID

Table continues on the next page...

Table continued from the previous page...

Field	Function
MSTRID	The ID of the AHB master associated with BUFFER2. Any AHB read access with this master port number is routed to this buffer. It must be ensured that the master IDs associated with all buffers are different.

38.4.2.8 Buffer 3 Configuration Register (BUF3CR)

Offset

Register	Offset
BUF3CR	1Ch

Function

This register provides the configuration for any access to buffer 3.

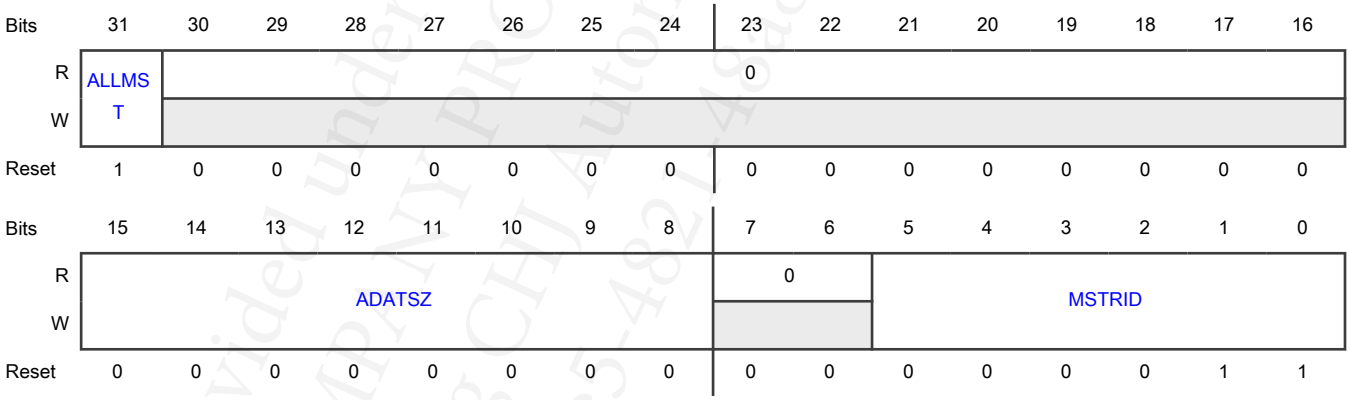
An access is routed to buffer 3 when the master port number of the incoming AHB request matches the MSTRID field of BUF3CR. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by BFGENCR[SEQID].

In case the value of the ALLMST field is not 1, any such transaction (where master port number does not match any of the MSTRID fields) is returned with an ERROR response.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31 ALLMST	All master enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When set, buffer3 acts as an all-master buffer. Any AHB access with a master port number not matching with the master ID of buffer0, buffer1, or buffer2 is routed to buffer3. When set, the MSTRID field of this register is ignored.
30-16 —	Reserved
15-8 ADATSZ	AHB data transfer size Defines the read data transfer size in 8 bytes of an AHB triggered read access to serial flash memory. When ADATSZ = 0, the data size mentioned in the sequence pointed to by the SEQID field overrides this value. The software should ensure that this transfer size is not greater than the size of this buffer. If OTFAD is enabled, the QuadSPI module prefetches only the required data from the flash memory so that the prefetch never crosses the 1 KB boundary.
7-6 —	Reserved
5-0 MSTRID	Master ID ID of the AHB master associated with BUFFER 3. Any AHB read access with this master port number is routed to this buffer. You must ensure that the master IDs associated with all buffers are different.

38.4.2.9 Buffer Generic Configuration Register (BFGENCR)

Offset

Register	Offset
BFGENCR	20h

Function

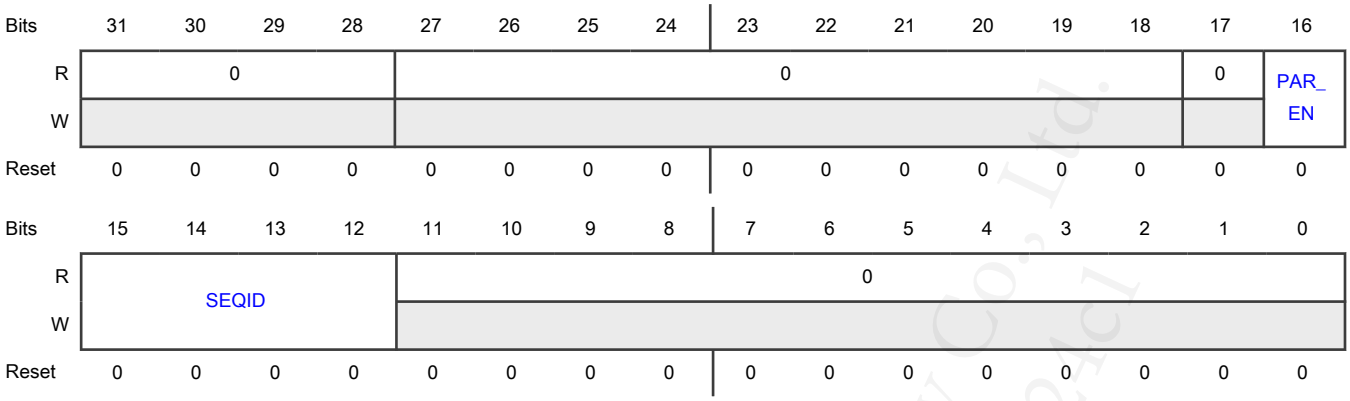
This register provides generic configuration to any of the buffer accesses. Any buffer "miss" leads to the buffer being flushed and a serial flash memory transaction being triggered per the sequence pointed to by the SEQID field. If the value of the PAR_EN field is 1, all the buffer accesses result in parallel accesses to the flash memories.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CR... Automotive Technology Co., Ltd.
c8c462e5-4e01-48a8-b251-390d3a...4c1

Diagram



Fields

Field	Function
31-28 —	Reserved
27-18 —	Reserved
17 —	Reserved
16 PAR_EN	Parallel mode enable If the value of this field is 1, a transaction to two serial flash memory devices is triggered in parallel mode. See Parallel flash memory mode for details.
15-12 SEQID	Points to a sequence in the LUT This field defines the bits [6:2] of the LUT index. The bits [1:0] are always assumed to be 0. See LUT. **NOTE** If the sequence pointer differs in the new and the previous sequences, you should reset it. See sequence pointer clear register for more information.
11-0 —	Reserved

38.4.2.10 Buffer 0 Top Index Register (BUF0IND)

Offset

Register	Offset
BUF0IND	30h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4814-8a8-b251-390d3ab624c1

Function

This register specifies the top index for buffer 0, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

The register value should be set to the desired number of bytes. For example, setting BUF0IND[31:3] to 0 gives 0 bytes, setting the value to 1 gives 8 bytes, and so on.

The size of buffer 0 is the difference between BUF0IND and 0.

The software must ensure that the value of TPINDX0 is not greater than the size of buffer 0.

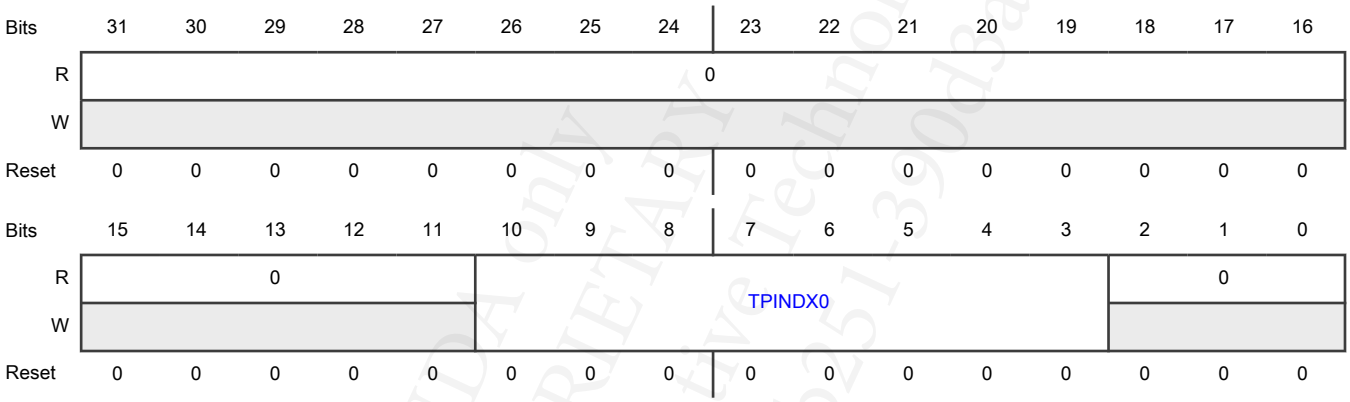
NOTE

The hardware does not provide any protection against illegal programming.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-11 —	Reserved
10-3 TPINDX0	Top index of buffer 0
2-0 —	Reserved

38.4.2.11 Buffer 1 Top Index Register (BUF1IND)

Offset

Register	Offset
BUF1IND	34h

Function

This register specifies the top index of buffer 1, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

The size of buffer 1 is the difference between BUF1IND and BUF0IND. The register value should be entered in bytes. For example, if BUF0IND = 0x100, then setting BUF1IND = 0x130 sets the size of buffer 1 to 0x30 bytes.

The software must ensure that the value of TPINDX1 is not greater than the size of buffer 1.

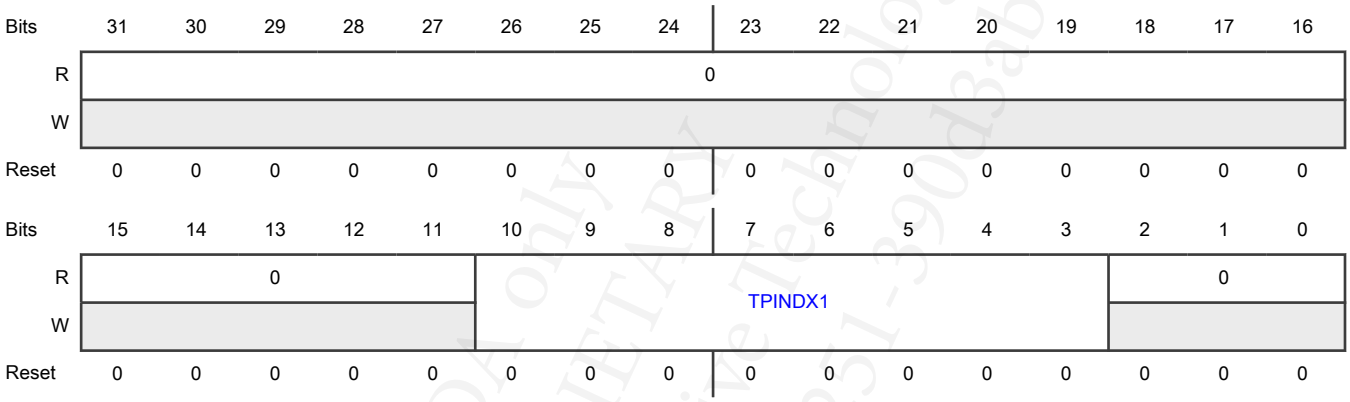
NOTE

The hardware does not provide any protection against illegal programming.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-11 —	Reserved
10-3 TPINDX1	Top index of buffer 1
2-0 —	Reserved

38.4.2.12 Buffer 2 Top Index Register (BUF2IND)

Offset

Register	Offset
BUF2IND	38h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-590d3ab624c1

Function

This register specifies the top index of buffer 2, which defines its size. Note that the three LSBs of this register are set to 0. This ensures that the buffer is 64-bit aligned because each buffer entry is 64-bits long.

The size of buffer 2 is the difference between BUF2IND and BUF1IND. The register value should be entered in bytes. For example, if BUF1IND = 0x130 then setting BUF2IND = 0x180 sets the size of buffer 2 to 0x50 bytes.

The software must ensure that the value of TPINDX2 is not greater than the size of buffer 2.

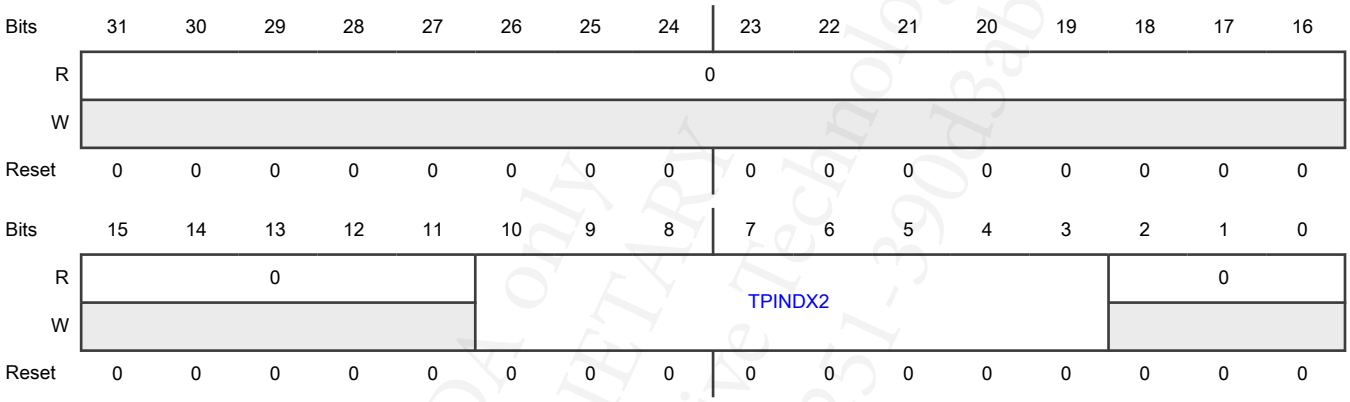
NOTE

The hardware does not provide any protection against illegal programming.

Special write-access is permitted if:

- SR[AHB_ACC] = 0

Diagram



Fields

Field	Function
31-11 —	Reserved
10-3 TPINDX2	Top index of buffer 2
2-0 —	Reserved

38.4.2.13 AHB Write Configuration Register (AWRCR)

Offset

Register	Offset
AWRCR	50h

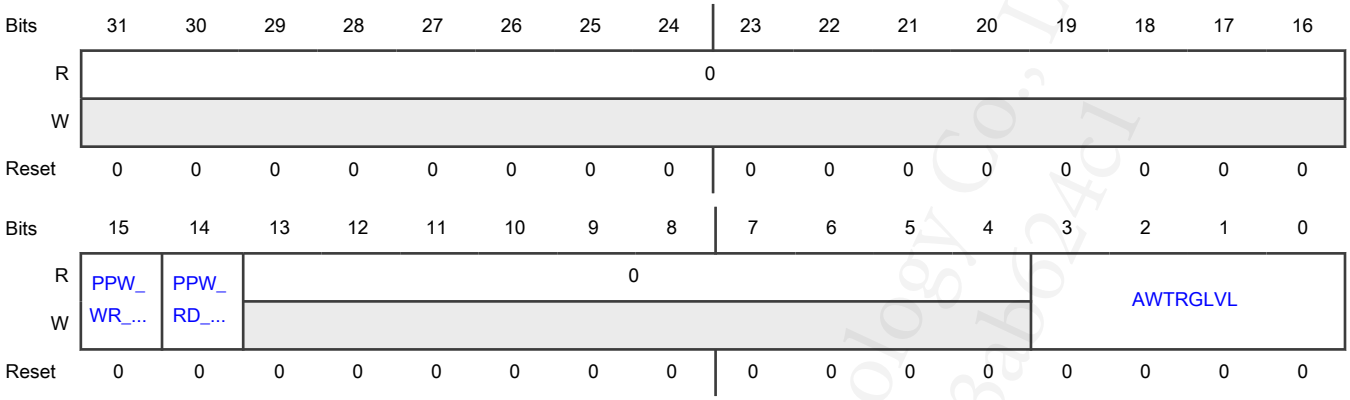
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46255-4821-48a8-b251-590d3ab624c1

Function

Special write-access is permitted if:

- SR[AWRACC] = 0

Diagram



Fields

Field	Function
31-16 —	Reserved
15 PPW_WR_DIS	Page program wait write disabled 0b - Enables subsequent writes 1b - Disables subsequent writes to the flash memory. After the first write transaction, AHB write is returned with an error response.
14 PPW_RD_DIS	Page program wait read disabled 0b - Enables subsequent reads 1b - Disables subsequent reads to the flash memory. After the first write transaction, AHB read is returned with an error response.
13-4 —	Reserved
3-0 AWTRGLVL	AHB write trigger level Defines a trigger level of TX FIFO in terms of a 4-byte entry. AHB write to the flash memory is triggered when either of the following happens: • TX FIFO crosses the level defined by this field • AHB transaction completes This is done to prevent an underrun in the flash memory.

Preliminary Information
COMPANY CONFIDENTIAL
Brilliant CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-390d3ab624c1

38.4.2.14 DLL Flash Memory A Configuration Register (DLLCRA)

Offset

Register	Offset
DLLCRA	60h

Function

This register configures DLL and slave delay chain for flash memory A.

The value of the DLLEN field must be 1 after all reference (FREQEN, DLL_REFCNTR, DLLRES, SLAVE_AUTO_UPDT) delay chain configurations are programmed.

See DLL and delay chain usage for the programming sequence.

NOTE

See the chip data sheet for information on programming register fields.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLLEN	FREQEN	0		DLL_REFCNTR				DLLRES				SLV_FINE_OFFSET			
W																
Reset	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	SLV_DLY_OFFSET			SLV_DLY_COARSE				Reserved			0	SLAVE_A...	SLV_EN	SLV_DLL...	SLV_UPD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DLLEN	DLL enable 0b - DLL reference logic remains in reset and should be 0 for at least three flash memory clock cycles for reset. 1b - Enables DLL logic. Set it to 1 after all the configuration for DLLCR reference settings is complete.
30 FREQEN	Frequency enable 0b - Selects delay chain for low frequency of operation 1b - Selects delay chain for high frequency of operation
29-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-24 DLL_REFCNTR	DLL reference counter Select the "n+1" interval of DLL phase detection and reference delay updating interval (minimum recommended value = 1).
23-20 DLLRES	DLL resolution Minimum resolution for DLL phase detector to remain locked/unlocked based on flash memory clock jitter. The minimum value is 2, and should be programmed to a more suitable value, such as 6.
19-16 SLV_FINE_OFFSET	Fine offset delay elements in incoming DQS This field sets the number of fine offset delay elements up to 16 in incoming DQS, and the default must be 1 element.
15 —	Reserved
14-12 SLV_DLY_OFFSET	T/16 offset delay elements in incoming DQS This field sets the number of T/16 offset delay elements in incoming DQS; default is 0.
11-8 SLV_DLY_COARSE	Delay elements in each delay tap This field sets the number of delay elements in each delay tap. The field is used to overwrite DLL-generated delay values and works when the value of SLV_DLL_BYPASS is 1.
7-5 —	Reserved
4 —	Reserved
3 SLAVE_AUTO_UPDT	Slave chain update This field automatically updates the slave chain as soon as DLL is locked. 0b - Auto-update feature is disabled. 1b - Auto-update feature is enabled.
2 SLV_EN	Slave enable 0b - DLL slave logic remains in reset, and its value should be 0 for at least three flash memory clock cycles for reset. 1b - Enables DQS slave delay chain, and should be 1 before any slave configuration settings take place.
1 SLV_DLL_BYPASS	Slave DLL bypass This field enables selection of the number of delays in each slave delay tap.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disables manual selection of coarse delays in the slave delay chain. 1b - Enables selection of number of delays in each slave delay tap, based on DLLCRA[SLV_DLY_COARSE].
0 SLV_UPD	Slave update You must program this field only after slave delay chain configuration takes place. 0b - Disables any further update on DQS slave delay chain. 1b - Updates the DQS slave delay chain with either ref-delay or bypass slave delay value, and should be set in the absence of the DQS clock.

38.4.2.15 DLL Flash Memory B Configuration Register (DLLCRB)

Offset

Register	Offset
DLLCRB	64h

Function

This register configures DLL and slave delay chain for flash memory B.

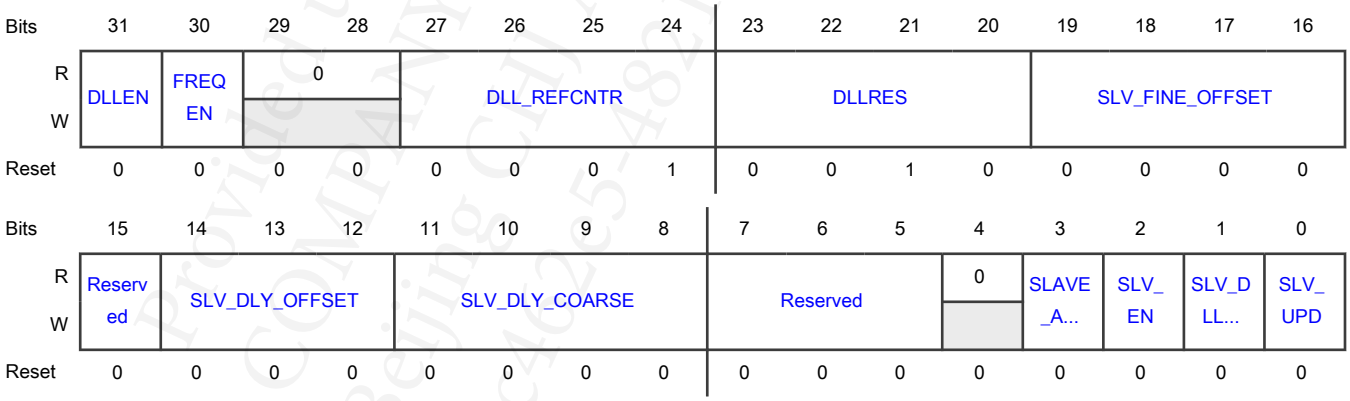
DLLEN must be set after all reference (FREQEN, DLL_REFCNTR, DLLRES, SLAVE_AUTO_UPDT) delay chain configurations are programmed.

See DLL and delay chain usage for the programming sequence.

NOTE

See the chip data sheet for programming the register fields.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4025-4821-4a8-b251-390d3ab624c1

Fields

Field	Function
31 DLLEN	DLL enable 0b - DLL reference logic remains in reset, keep in "0" for at least 3 flash memory clock cycles for reset. 1b - Enables DLL logic. Set it to "1" after all configuration of DLLCR reference settings.
30 FREQEN	Frequency enable 0b - Selects delay chain for low frequency of operation 1b - Selects delay chain for high frequency of operation
29-28 —	Reserved
27-24 DLL_REFCNTR	DLL reference counter Select the "n+1" interval of DLL phase detection and reference delay updating interval (minimum recommended value is 1).
23-20 DLLRES	DLL resolution Minimum resolution for DLL phase detector to remain locked or unlocked based on flash memory clock jitter. Minimum value is 2, and you should program it to a more some suitable value, such as 6.
19-16 SLV_FINE_OFFSET	Number of delay elements in each delay tap This field sets the number of fine offset delay elements up to 16 in incoming DQS; default should be '1' element.
15 —	Reserved
14-12 SLV_DLY_OFFSET	Number of delay elements in each delay tap This field sets the number of T/16 offset delay elements in incoming DQS; the default is 0.
11-8 SLV_DLY_COARSE	Number of delay elements in each delay tap This field sets the number of delay elements in each delay tap. The field is used to overwrite the DLL generated delay values and works when SLV_DLL_BYPASS = 1.
7-5 —	Reserved
4 —	Reserved
3	Update slave chain This field automatically updates the slave chain as soon as DLL is locked.

Table continues on the next page...

Table continued from the previous page...

Field	Function
SLAVE_AUTO_UPDT	0b - Auto-update feature is disabled. 1b - Auto-update feature is enabled.
2 SLV_EN	Slave enable 0b - DLL slave logic remains in reset. Keep it "0" for at least three flash memory clock cycles for reset. 1b - Enables DQS slave delay chain. Set to "1" before any slave configuration settings
1 SLV_DLL_BYPASS	Slave DLL Bypass This field enables selection of the number of delay in each slave delay tap. 0b - Disables manual selection of coarse delays in slave delay chain. 1b - Enables selection of number of delay in each slave delay tap based on DLLCRB[SLV_DLY_COARSE]
0 SLV_UPD	Slave update Program this field only after slave delay chain configuration. 0b - Disables any further update on DQS slave delay chain. 1b - Updates DQS slave delay chain with either the ref-delay or bypass slave delay value. The field should be set when there is no DQS clk.

38.4.2.16 Parity Configuration Register (PARITYCR)

Offset

Register	Offset
PARITYCR	6Ch

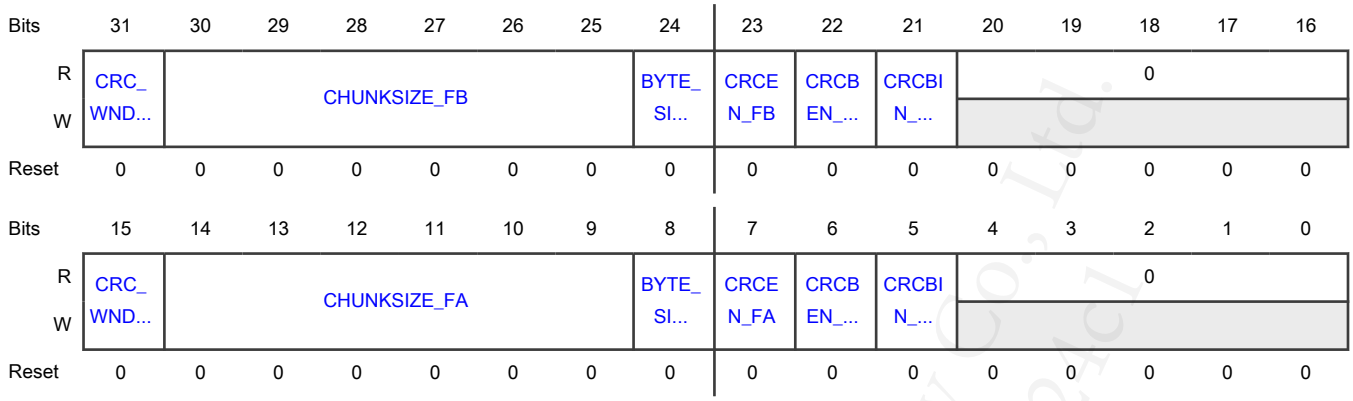
Function

This register replicates the parity related configuration programming performed on the flash memory configuration registers.

You must write 1 to PARITYCR[CRCEN_FA] and PARITYCR[CRCEN_FB] after all parity configuration bits are programmed. However, you can program PARITYCR[BYTE_SIZE_FA] and PARITYCR[BYTE_SIZE_FB] at any time.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-8921-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 CRC_WNDW_FB	CRC address window configuration 0b - Calculates parity (CRC) over fixed address window within the denoted flash memory B chunk size boundary. 1b - Calculates parity (CRC) over incremental window of flash memory B chunk size irrespective of address.
30-25 CHUNKSIZE_FB	Chunk size for flash memory B Defines the chunk size (in terms of 4 Bytes) after parity should be inserted and compared by the QuadSPI controller for flash memory B. Values 0x1, 0x2, ... implies 4 bytes, 8 bytes ... so on.
24 BYTE_SIZE_FB	Byte size for flash memory B It can be programmed anytime. Enables single byte read/write parity with FlashB configuration register. This field overrides chink-size used for Flash data array. You must write 1 to this field before any read/write operation on the flash memory internal registers. And you must write 0 to this field before reading flash memory data array.
23 CRCEN_FB	CRC parity checker logic 0b - Disables parity mechanism. In case of parity error, set to "0" to clear parity error. Only supported for DDR octal commands. 1b - Enables CRC parity checker logic for flash memory B read paths. Set this field after programming CHUNK_SIZEB, CRCBEN_FB and CRCBIN_FB.
22 CRCBEN_FB	Adds CRC bar parity from flash memory B output to QuadSPI controller.
21 CRCBIN_FB	Adds CRC bar parity to flash memory B input from QuadSPI controller
20-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 CRC_WNDW_FA	CRC address window configuration 0b - Calculates parity (CRC) over fixed address window within the denoted flash memory A chunk size boundary. 1b - Calculates parity (CRC) over incremental window of flash memory A chunk size, irrespective of address.
14-9 CHUNKSIZE_FA	Chunk size for flash memory A Defines the chunk size (in terms of 4 bytes) after parity is inserted and compared by the QuadSPI controller for flash memory A. Values 0x1, 0x2, and so on indicate 4 bytes, 8 bytes, and so on.
8 BYTE_SIZE_FA	Byte size for flash memory A This field can be programmed any time. It enables single-byte, read/write parity with flash memory A configuration register. The field overrides the chunk size used for flash memory data array. You must write 1 to this field before any read/write operation on the flash memory internal registers. And you must write 0 to this field before reading flash memory data array.
7 CRCEN_FA	CRC parity checker logic 0b - Disables parity mechanism. In case of parity error, set it to 0 to clear parity error. Only supported for DDR Octal commands. 1b - CRC parity checker logic for flash memory A read paths. Configure this field after programming CHUNK_SIZEA, CRCBEN_FA and CRCBIN_FA.
6 CRCBEN_FA	Adds CRC bar parity from flash memory A output to QuadSPI controller
5 CRCBIN_FA	Adds CRC bar parity to flash memory A input from QuadSPI controller
4-0 —	Reserved

38.4.2.17 Serial Flash Memory Address Register (SFAR)

Offset

Register	Offset
SFAR	100h

Function

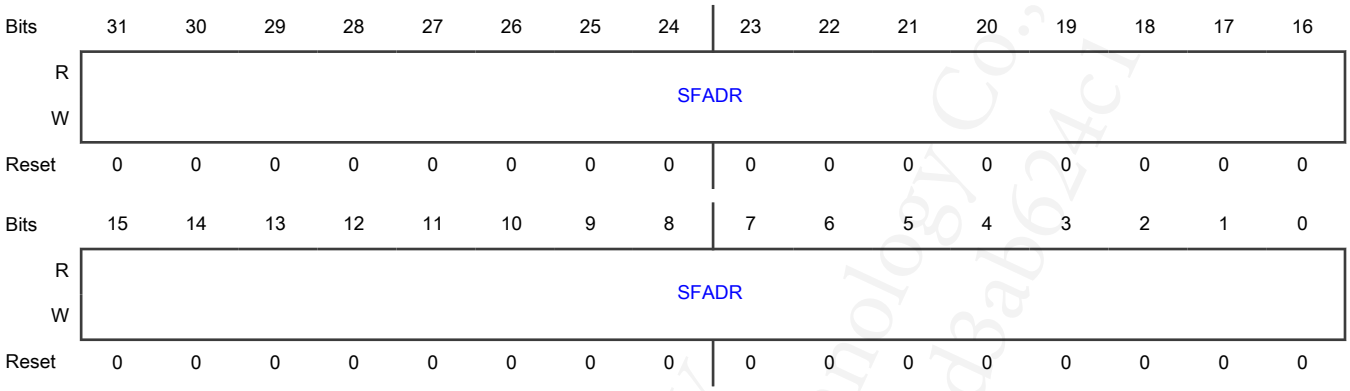
The module automatically translates this address on the memory map to the address on the flash memory. When operating in a 24-bit mode, only bits 23-0 are sent to the flash memory. In the 32-bit mode, bits 27-0 are used with bits 31-28 driven to 0 when the value of SFACR[CAS] is 0. For example, if the value of SFACR[CAS] is 3, then bits 26-3 are sent to the flash memory as its page address in case flash memory is operating in a 24-bit mode. The total number of address bits requested by the flash memory, as

its page and column address, must not be more than 32 bits. See Table 212 for the mapping between the access mode and the SFAR content and Normal mode for details on command triggering and command execution. The software must ensure that the serial flash memory address provided in the SFAR register lies in the valid flash memory address range, as defined in Table 212.

Special write-access is permitted if:

- SR[IP_ACC] = 0

Diagram



Fields

Field	Function
31-0 SFADR	Serial flash memory address

38.4.2.18 Serial Flash Memory Address Configuration Register (SFACR)

Offset

Register	Offset
SFACR	104h

Function

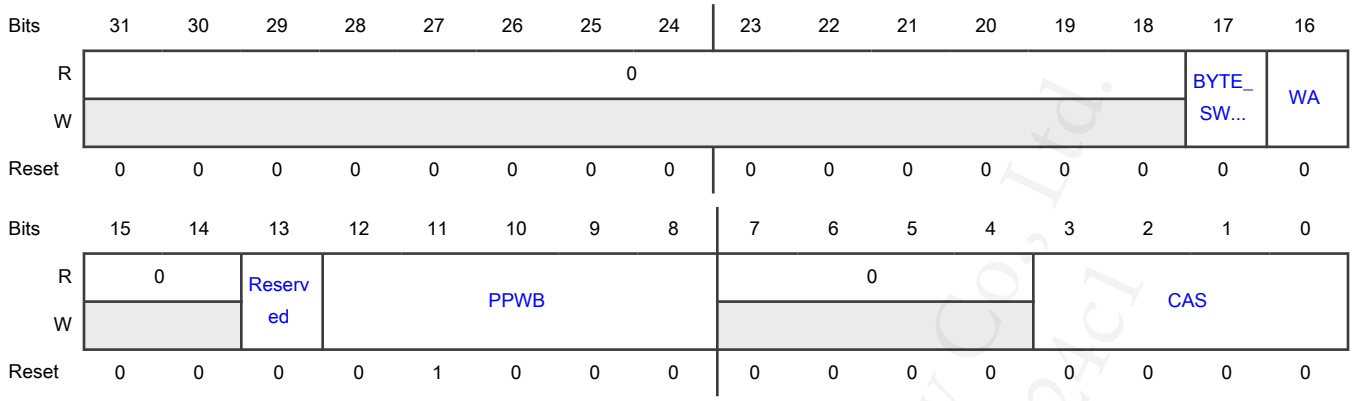
This register contains the address requirements that are specific to serial flash memory. These requirements must be configured according to the connected flash memory, for the controller to function properly. The module automatically translates the address of SFAR on the memory map or the incoming address on the AHB bus to the column address on the flash memory. For example, if a flash memory needs 3 bits as its column address, then only the lower three bits of the SFAR/AHB address are sent to the flash memory as its column address. The software should ensure that the serial flash memory address provided in SFAR or the incoming AHB address lies in the valid flash memory address range.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Printed under IDA only
COMPANY PROPRIETARY
Beijing ECARX Automotive Technology Co., Ltd.
c8c46265-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-18 —	Reserved
17 BYTE_SWAP	Byte swapping In case of Octal DDR mode, this field controls whether a word unit composed of 2 bytes from posedge and negedge of a single DQS cycle needs to be swapped. 0b - One word of two bytes at [nth, n+1th] address 1b - One word of two bytes at [n+1th, nth] address
16 WA	Word addressable Defines whether the serial flash memory is a byte addressable flash memory or a word addressable flash memory. According to the configuration of this field, the address is remapped to the flash memory interface. See Address scheme for details. 0b - Byte addressable serial flash memory mode 1b - Word (2-byte) addressable serial flash memory mode
15-14 —	Reserved
13 —	Reserved
12-8 PPWB	Page program boundary Flash memory-specific page program boundary size should be programmed in Log2 (size in bytes) format. The default is 8 = log2(256) for a 256-byte page program size.
7-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 CAS	Column address space Defines the width of the column address. If the column address is, for example, [2:0] of SFAR/AHB address, then CAS must be 3. If there is no column address separation in any serial flash memory, the value of this field must be specified as 0.

38.4.2.19 Sampling Register (SMPR)

Offset

Register	Offset
SMPR	108h

Function

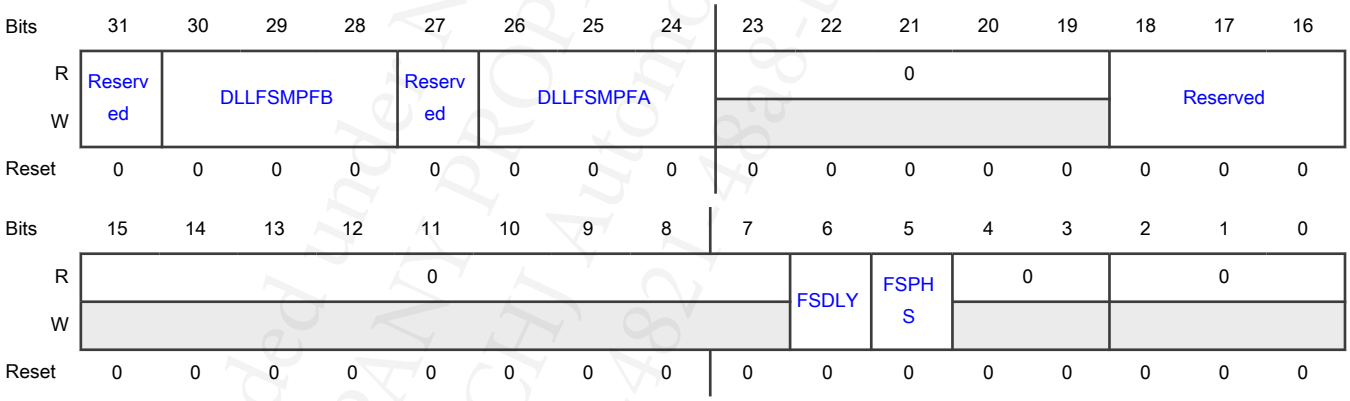
This register allows configuration of how the incoming data from the external serial flash memory devices is sampled in the QuadSPI module.

NOTE

See the chip data sheet for programming the register fields.

Special write-access is permitted in the disabled mode.

Diagram



Fields

Field	Function
31 —	Reserved
30-28	Selects the nth tap provided by slave delay chain for flash memory B

Table continues on the next page...

Table continued from the previous page...

Field	Function
DLLFSMPFB	The value of *n* can vary from 0 to 7, with each tap delay based on the DLLCRB register.
27 —	Reserved
26-24 DLLFSMPFA	Selects the nth tap provided by slave delay chain for flash memory A The value of *n* can vary from 0 to 7, with each tap delay based on the DLLCRA register.
23-19 —	Reserved
18-16 —	Reserved
15-7 —	Reserved
6 FSDLY	Full-speed delay selection for internal/pad loop back DQS sampling This field selects the delay in accordance with the reference edge for the valid sample point. 0b - Same DQS 1b - Half-cycle early DQS
5 FSPHS	Full-speed phase selection for SDR instructions This field selects the edge of the sampling clock valid for full-speed commands. 0b - Select sampling at non-inverted clock 1b - Select sampling at inverted clock
4-3 —	Reserved
2-0 —	Reserved

38.4.2.20 RX Buffer Status Register (RBSR)

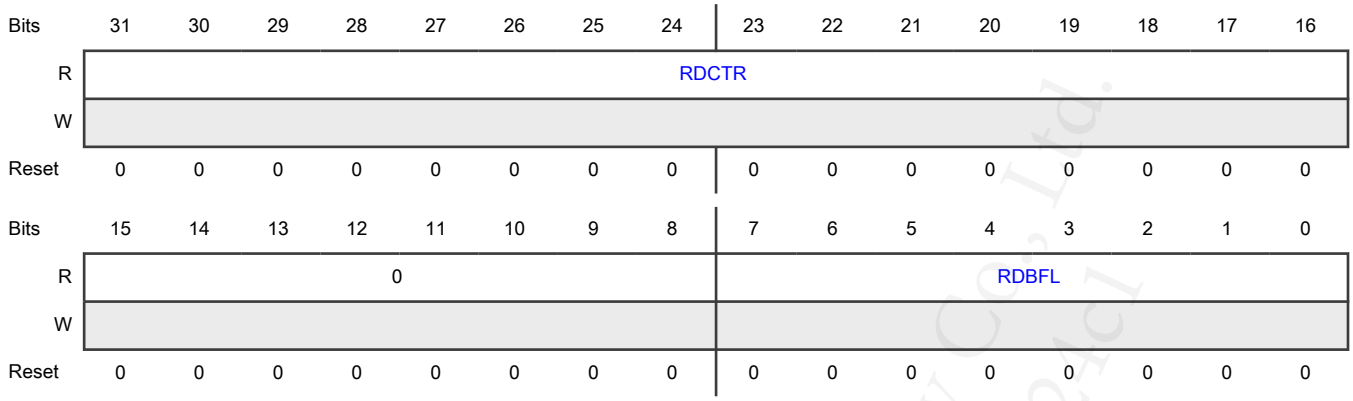
Offset

Register	Offset
RBSR	10Ch

Function

This register contains information related to the receive data buffer.

Diagram



Fields

Field	Function
31-16 RDCTR	Read counter Indicates the number of 4-byte entries removed from the RX buffer. For example, a value of 0x2 indicates that 8 bytes have been removed. It is incremented by the number (RBCT[WMRK] + 1) on RX buffer POP event. The RX buffer can be popped using DMA or FR[RBDF]. The RSER[RBDDE] defines which pop should be pursued. For details, see AHB RX Data Buffer Register (ARDB0 - ARDB127) and Data Transfer from the QuadSPI Module Internal Buffers.
15-8 —	Reserved
7-0 RDBFL	RX buffer fill level Indicates the number of 4-byte entries available in the RX buffer. For example, a value of 0x2 indicates 8 bytes are available.

38.4.2.21 RX Buffer Control Register (RBCT)

Offset

Register	Offset
RBCT	110h

Function

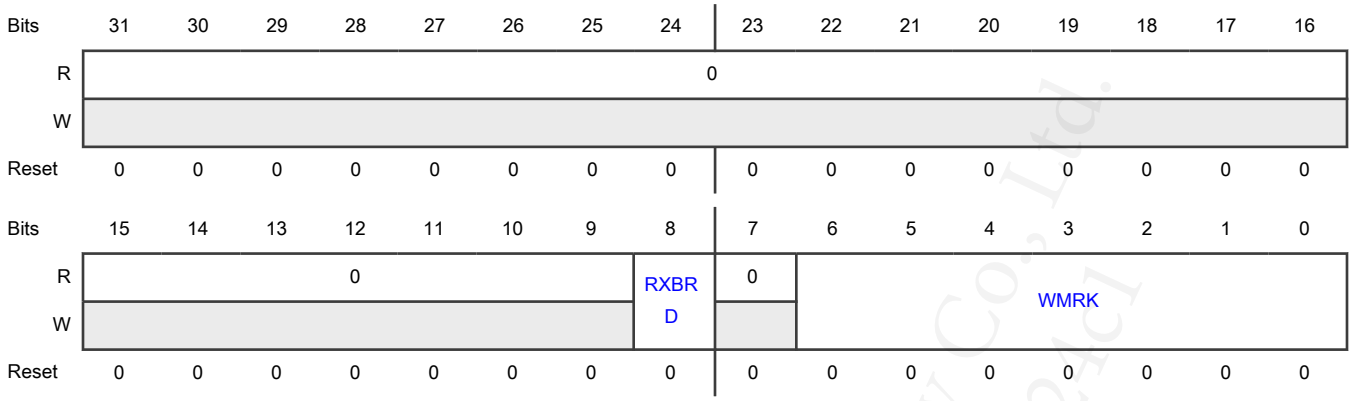
This register contains control data related to the receive data buffer.

Special write-access is permitted if:

- SR[IP_ACC] = 0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a2-b251-90d3ab624c1

Diagram



Fields

Field	Function
31-9 —	Reserved
8 RXBRD	RX buffer readout This field specifies the access scheme for the RX buffer readout. 0b — RX buffer content is read using the AHB bus registers. See AHB RX Data Buffer Register (ARDB0 - ARDB127). 1b — RX buffer content is read using the IP bus registers. See RX Buffer Data Register (RBDR0 - RBDR63).
7 —	Reserved
6-0 WMRK	RX buffer watermark This field determines when the readout action of the RX buffer is triggered. When the number of valid entries in the RX buffer is equal to or greater than the number given by (WMRK+1), the SR[RXWE] flag is asserted. The value should be entered as the number of 4-byte entries minus 1. For example, a value of 0x0 sets the watermark to 4 bytes, 1 to 8 bytes, 2 to 12 bytes, and so on. For details, see DMA usage.

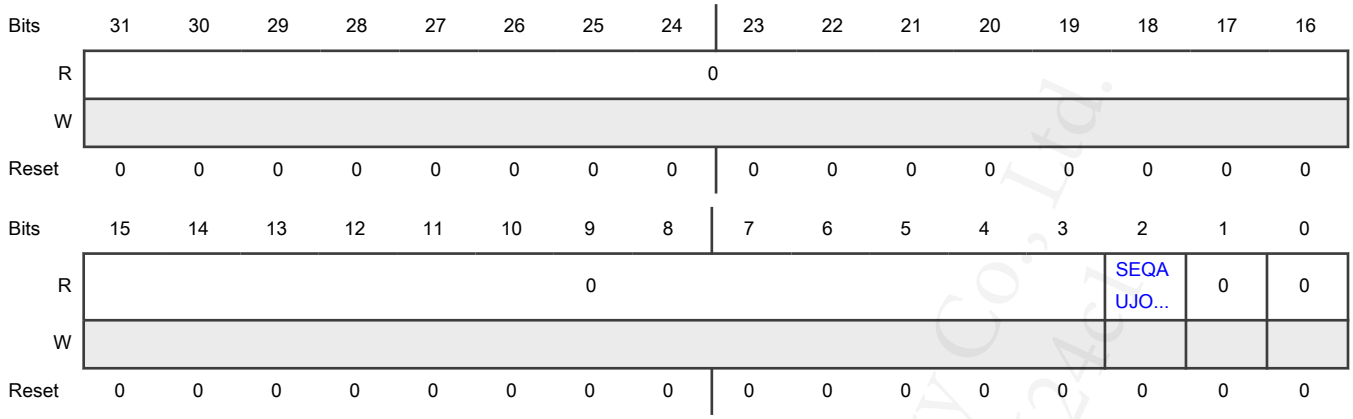
38.4.2.22 AHB Write Status Register (AWRSR)

Offset

Register	Offset
AWRSR	120h

Function

Diagram



Fields

Field	Function
31-3 —	Reserved
2 SEQAUJOIN	Sequence auto join Asserted when the LUT sequences are automatically joined using the JMP_TO_SEQ command. This remains asserted as long as you do not encounter the STOP/JMP_ON_CS command.
1 —	Reserved
0 —	Reserved

38.4.2.23 DLL Status Register (DLLSR)

Offset

Register	Offset
DLLSR	12Ch

Function

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b25f-390d3ab624d1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLLB_LO...	SLVB_LO...	DLLB_RA...	DLLB_FI...	0				DLLB_SLV_FINE_VAL				DLLB_SLV_COARSE_VAL			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLLA_LO...	SLVA_LO...	DLLA_RA...	DLLA_FI...	0				DLLA_SLV_FINE_VAL				DLLA_SLV_COARSE_VAL			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DLLB_LOCK	DLL B lock status
30 SLVB_LOCK	High frequency slave delay chain locked Sets when slave decoder is updated with DLLCRB[SLV_UPD] It gets reset after user programs SLV_UPD=0 after observing This field along with Lock status (in case of non-bypass) mode.
29 DLLB_RANGE_ERR	DLL master delay chain "1" indicates that DLL master delay chain is working out of delay range due to incorrect DLL configuration.
28 DLLB_FINE_UNDERFLOW	Fine delay chain underflow "1" indicates fine delay chain underflow has occurred.
27-24 —	Reserved
23-20 DLLB_SLV_FINE_VAL	Fine delay cells in slave delay chain This value indicates total no of fine delay cells (1 delay unit) selected in slave delay chain.
19-16 DLLB_SLV_COARSE_VAL	Coarse delay cells This value indicates total no of coarse delay cells (16 delay unit) selected in slave delay chain.
15 DLLA_LOCK	DLL A lock status
14	Slave high lock status

Table continues on the next page...

Table continued from the previous page...

Field	Function
SLVA_LOCK	High-frequency slave delay chain locked. The field is configured when the slave decoder is updated with DLLCRA[SLV_UPD] and is reset after you configure SLV_UPD to 0.
13 DLLA_RANGE_ERR	DLL master delay chain The value 1 indicates that DLL master delay chain is working out of delay range because of incorrect DLL configuration.
12 DLLA_FINE_UNDERFLOW	Fine delay chain underflow The value 1 indicates that fine delay chain underflow has occurred.
11-8 —	Reserved
7-4 DLLA_SLV_FINE_VAL	Fine delay cells in slave delay chain This value indicates the total number of fine delay cells (1 delay unit) selected in the slave delay chain.
3-0 DLLA_SLV_COARSE_VAL	Coarse delay cells in slave delay chain This value indicates the total number of coarse delay cells (16 delay units) selected in the slave delay chain.

38.4.2.24 Data Learning Configuration Register (DLCR)

Offset

Register	Offset
DLCR	130h

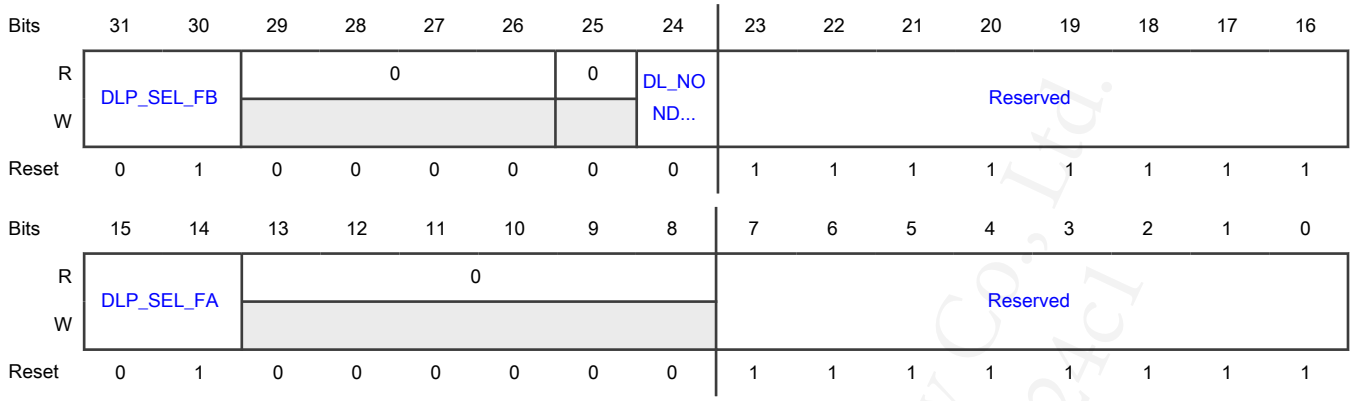
Function

This register is used for programming the data learning settings.

- A flash memory device that supports data learning but cannot provide more than 10 bits of data learn pattern must be considered as non-DLP Flash. For Non-DLP flash devices, you must read at least 16 bits data learn pattern from a known location.
- You must program different learning pattern on the two data pins–IO1 and IO3. This is required to accommodate pessimistic data skew between the different IO lines.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-0821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-30 DLP_SEL_FB	Selects a pattern matching IO pads Special write-access is permitted in the disabled mode. 00b - Pattern matching is ignored. 01b - IO1 is used for matching. 10b - IO3 is used for matching. 11b - Both IO1 and IO3 are used for pattern matching.
29-26 —	Reserved
25 —	Reserved
24 DL_NONDLP_FLSH	Data learning enabled for non-DLP flash memory Configure the value of this field as 1 to enable data learning for flash memories that do not provide a data learning pattern. For a non-DLP flash memory, execute a one-time data learning through the IPS to select a tap before an AHB read operation.
23-16 —	Reserved
15-14 DLP_SEL_FA	Selects pattern matching IO pads 00b - Pattern matching is ignored 01b - IO1 is used for matching 10b - IO3 is used for matching 11b - Both IO1 and IO3 are used for pattern matching

Table continues on the next page...

Table continued from the previous page...

Field	Function
13-8 —	Reserved
7-0 —	Reserved

38.4.2.25 Data Learning Status Flash Memory A Register (DLSR_FA)

Offset

Register	Offset
DLSR_FA	134h

Function

This register shows sampling point selected by data learning algorithm when the value of DLSR_FA[DLPFFA] is 0. Otherwise, it shows the pattern matching outline.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLPFFA	0														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POS_EDGE								NEG_EDGE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DLPFFA	Data learning pattern fail This field asserts when data learning fails at flash memory A.
30-16 —	Reserved
15-8	DLP positive edge match signature for flash memory A

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4821-4a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
POS_EDGE	
7-0 NEG_EDGE	DLP negative edge match signature for flash memory A

38.4.2.26 Data Learning Status Flash Memory B Register (DLSR_FB)

Offset

Register	Offset
DLSR_FB	138h

Function

This register shows sampling point selected by data learning algorithm when DLSR_FB[DLPFFB] is '0'. Otherwise, it shows the pattern matching outline.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLPFFB								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POS_EDGE								NEG_EDGE							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DLPFFB	Data learning pattern fail This field asserts when data learning fails at flash memory B
30-16 —	Reserved
15-8 POS_EDGE	DLP pos edge match signature for flash memory B

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 NEG_EDGE	DLP neg edge match signature for flash memory B

38.4.2.27 TX Buffer Status Register (TBSR)

Offset

Register	Offset
TBSR	150h

Function

This register contains information related to the transmit data buffer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TRCTR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TRBFL							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 TRCTR	Transmit counter This field indicates how many entries of 4 bytes have been written into the TX buffer by host accesses. It is reset to 0 when a 1 is written to MCR[CLR_TXF]. It is incremented on each write access to the TBDR register when another word has been pushed onto the TX buffer. When it is not cleared, the TRCTR field wraps around to 0. See TX Buffer Data Register (TBDR) for details.
15-8 —	Reserved
7-0 TRBFL	TX buffer fill level

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field contains the number of entries of 4 bytes each available in the TX buffer for the QuadSPI module to transmit to the serial flash memory device. The value of this field can reach maximum up to the total TX buffer size.

38.4.2.28 TX Buffer Data Register (TBDR)

Offset

Register	Offset
TBDR	154h

Function

This register provides access to the circular TX buffer of depth 256, so the total size is 256 * 4 bytes. This buffer provides the data written into it as write data for the page programming commands to the serial flash memory device. See Table 220 for the byte ordering scheme. A write transaction on the flash memory with data size of less than 32 bits leads to the removal of one data entry from the TX buffer. The valid bits are used and the rest of the bits are discarded.

Special write-access is permitted if:

- SR[TXFULL] = 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TXDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TXDATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TXDATA	TX data On write access, the data is written to the next available entry of the TX buffer and TBSR[TRBFL] is updated accordingly. On a read access, the last data written to the register is returned.

Provided under NDA of COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. cc462e5-482-1988-b251-393a3ab624c1

38.4.2.29 TX Buffer Control Register (TBCT)

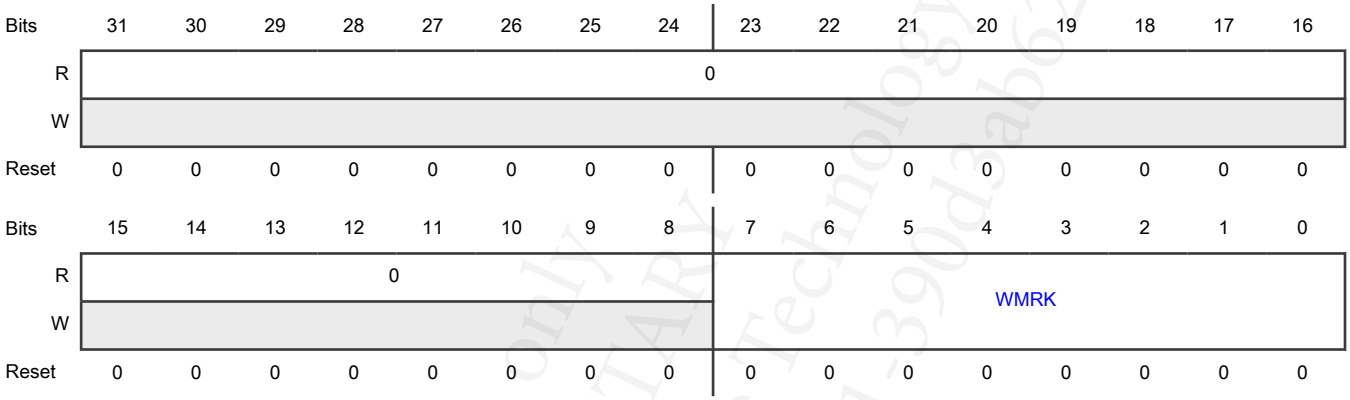
Offset

Register	Offset
TBCT	158h

Function

This register contains control information for transmit data buffer.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 WMRK	Watermark for TX buffer Determines the watermark for the TX buffer When the number of available space in the TX buffer is greater than or equal to the number provided by WMRK (number of 4-byte entries), SR[TXWA] is asserted. For example, a value of 0x1 sets the watermark to 4 bytes, 0x2 sets it to 8 bytes, 0x3 sets it to 12 bytes, and so on. For details, see DMA usage. WMRK = 0 is invalid.

38.4.2.30 Status Register (SR)

Offset

Register	Offset
SR	15Ch

Function

This register provides all the available status information about SFM command execution and arbitration, the RX buffer, TX buffer, and the AHB buffer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			0	TXFULL	TXDMA	TXWA	TXNE	RXDMA	0			RXFULL	0		RXWE
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	AHB3FUL	AHB2FUL	AHB1FUL	AHB0FUL	AHB3NE	AHB2NE	AHB1NE	AHB0NE	AHBTRN	0	AWRACC	0	AHB_ACC	IP_ACC	BUSY
W																
Reset	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 —	Reserved
27 TXFULL	TX buffer full Asserted when the FIFO level reaches 263 (that is, TX buffer size of 256 + async FIFO size of 7)
26 TXDMA	TX DMA Asserted when the TXFIFO fill via DMA is active and DMA is requested or running
25 TXWA	TX buffer watermark available Asserted when the number of available spaces in the TX buffer is greater than or equal to the value provided by TBCT[WMRK]
24 TXNE	TX buffer not empty Asserted when TX buffer contains data
23 RXDMA	RX buffer DMA Asserted when RX buffer read out via DMA is active; that is, when DMA is requested or running
22-20 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 RXFULL	RX buffer full Asserted when the RX buffer is full; that is, when RBSR[RDBFL] is equal to 32
18-17 —	Reserved
16 RXWE	RX buffer watermark exceeded Asserted when the number of valid entries in the RX buffer exceeds the number provided in RBCT[WMRK]
15 —	Reserved
14 AHB3FUL	AHB 3 buffer full Asserted when AHB 3 buffer is full
13 AHB2FUL	AHB 2 buffer full Asserted when AHB 2 buffer is full
12 AHB1FUL	AHB 1 buffer full Asserted when the AHB 1 buffer is full
11 AHB0FUL	AHB 0 buffer full Asserted when the AHB 0 buffer is full
10 AHB3NE	AHB 3 buffer not empty Asserted when the AHB 3 buffer contains data
9 AHB2NE	AHB 2 buffer not empty Asserted when the AHB 2 buffer contains data
8 AHB1NE	AHB 1 buffer not empty Asserted when the AHB 1 buffer contains data
7 AHB0NE	AHB 0 buffer not empty Asserted when the AHB 0 buffer contains data
6 AHBTRN	AHB access transaction pending Asserted when there is a pending request on the AHB interface. See Flash memory mapped AMBA bus.
5 —	Reserved
4	AHB write access

Table continues on the next page...

Table continued from the previous page...

Field	Function
AWRACC	Asserted when AHB write access is enabled
3 —	Reserved
2 AHB_ACC	AHB read access Asserted when the currently executed transaction is initiated by the AHB bus
1 IP_ACC	IP access Asserted when transaction currently executed is initiated by the IP bus
0 BUSY	Module busy Asserted when module is currently busy handling a transaction to an external flash memory device

38.4.2.31 Flag Register (FR)

Offset

Register	Offset
FR	160h

Function

This register provides all available flags about SFM command execution and arbitration, which may serve as the source for the generation of interrupt service requests. Note that the error flags in this register do not relate directly to the execution of the transaction in the serial flash memory device itself but only to the behavior and conditions visible in the QuadSPI module.

Special write-access is permitted in the enabled mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLPFF	0		DLLABRT	TBFF	TBUF	0	DLLUNLCK	ILLINE	0		0	0		RBOF	RBDF
W	W1C			W1C	W1C	W1C		W1C	W1C						W1C	W1C
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AAEF	AITEF	0	ABOF	IUEF	CRCAEF	CRCBEF	PPWF	IPAEF	IPIEF	0	0	0			TFF
W	W1C	W1C		W1C	W1C	W1C	W1C	W1C	W1C	W1C						W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing Tongyi Automotive Technology Co., Ltd.
c8c2e5f1-4821-b251-390a2ab624c1

Fields

Field	Function
31 DLPFF	Data learning pattern failure flag This field is set when the DATA_LEARN instruction is encountered in a sequence, but no sampling point is found for the data learning pattern.
30-29 —	Reserved
28 DLLABRT	DLL abort 1b - This field is set whenever DLL is unlocked while reading data from the flash memory.
27 TBFF	TX buffer fill flag Before writing to the TX buffer, this field should be cleared. Then, it should be read back. If it is set, the TX buffer can include more data. If the field remains cleared, the TX buffer can be considered as full. See TX buffer operation for details.
26 TBUF	TX buffer underrun flag This field is set if the module tries to pull data when the TX buffer is empty. The IP command leading to the TX buffer underrun is continued (data sent to the serial flash memory device is undefined). Here, a valid underrun means that it should have occurred during the transaction so that few bytes (that is, less than 4 bytes) are left in FIFO and the remaining are filled with "FFFFh". The software should initiate a TX transaction only when 128 bits are written in the TX buffer. This field does not set if transfer is less than 128 bits. The application must clear the TX buffer in response to this event by writing a 1 to MCR[CLR_TXF]. The application must clear the TX buffer in response to this event by writing a 1 to MCR[CLR_TXF].
25 —	Reserved
24 DLLUNLCK	DLL unlock 1b - This field is set whenever DLL unlock event occurs, irrespective of flash memory access.
23 ILLINE	Illegal instruction error flag This field is set when an illegal instruction is encountered by the controller in any of the sequences. As soon as the field is set, you must assert MCR[SWRSTSD] and MCR[SWRSTHD]. That is, reset the flash memory and AHB domain after reconfiguring the correct sequence instruction. See Table 218 for a list of legal instructions.
22-21 —	Reserved
20 —	Reserved
19-18 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 RBOF	RX buffer overflow flag This field is set when no more data can be pushed into the RX buffer from the serial flash memory device. The IP command leading to this condition is continued until the number of bytes in IPCR[IDATSZ] are read from the serial flash memory device. The content of the RX buffer remains unchanged.
16 RBDF	RX buffer drain flag This field is set if SR[RXWE] is asserted. Writing 1 to this field triggers one of the following actions: • If the RX buffer has up to RBCT[WMRK] valid entries, then the flag is cleared. • If the RX buffer has more than RBCT[WMRK] valid entries and the RSER[RBDDE] field is not set (flag driven mode), an RX buffer POP event is triggered. The flag remains set if the RX buffer contains more than RBCT[WMRK] valid entries after the RX buffer POP event is complete. The flag is cleared if the RX buffer contains less than or equal to RBCT[WMRK] valid entries after the RX buffer POP event is complete. See the "Receive Buffer Drain Interrupt or DMA Request" section in Normal mode interrupt and DMA requests for details.
15 AAEF	AHB abort error flag This field is set in case software abort is asserted in an ongoing AHB transaction. The field should be cleared before starting any new AHB transaction.
14 AITEF	AHB illegal transaction error flag This is set whenever there is no response generated from QuadSPI to AHB bus in case of an illegal transaction and the watchdog timer expires. The timer value is considered as a parameter.
13 —	Reserved
12 ABOF	AHB buffer overflow flag This is set when the size of the AHB access exceeds the size of the AHB buffer. This condition can occur only if BUFxCR[ADATSZ] is programmed incorrectly. The AHB command leading to this condition is continued until the number of entries according to BUFxCR[ADATSZ] have been read from the serial flash memory device. The content of the AHB buffer is not changed.
11 IUEF	IP command usage error flag This is set when in parallel flash memory mode, the execution of an IP command is started with more than one pad enabled and the sequence pointed to by the sequence ID contains a WRITE or a WRITE_v2 command. See Table 218 for the related commands.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Communication with the serial flash memory device is terminated before the execution of WRITE/ WRITE_v2 command by the QuadSPI module.
10 CRCAEF	Sets when there is CRC or ECC error for flash memory A 0b - CRCEF interrupt is not generated. 1b - CRCEF interrupt is generated.
9 CRCBEF	Sets when there is CRC or ECC error for flash memory B 0b - CRCEF interrupt is not generated. 1b - CRCEF interrupt is generated.
8 PPWF	Page-program wait flag after flash memory write flag This field indicates page-program wait flag after flash memory write.
7 IPAEF	IP command trigger during AHB access error flag This is set when the following condition occurs: • A write access occurs to IPCR[SEQID] and the SR[AHB_ACC] field is set. Any command leading to the assertion of the IPAEF field is ignored.
6 IPIEF	IP command trigger could not be executed error flag This is set when the SR[IP_ACC] and SR[AWRACC] fields are set (that is, an IP triggered command is currently executing) and any of the following conditions occurs: • Write access to the IPCR. Any command leading to the assertion of the IPIEF flag is ignored. • Write access to the SFAR • Write access to the RBCT
5 —	Reserved
4 —	Reserved
3-1 —	Reserved
0 TFF	IP command transaction finished flag This field is set after the QuadSPI module completes a running IP command. If an error occurs, and the related error flags are valid in the same clock cycle, the TFF flag is asserted.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b51-990d3ab624c1

38.4.2.32 Interrupt and DMA Request Select and Enable Register (RSER)

Offset

Register	Offset
RSER	164h

Function

This register provides enables and selectors for the interrupts in the QuadSPI module.

NOTE

Each field of the FR enabled as source for an interrupt prevents the QuadSPI module from entering the Stop mode or Module Disable mode when this flag is set.

Special write-access is permitted in the "Anytime" mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLPFIE	Reserved		0	TBFIE	TBUIE	TBFDE	DLLULIE	ILLINIE	Reserved	RBDDE	Reserved	0		RBOIE	RBDIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AAIE	AITIE	AIBSIE	ABOIE	IUEIE	CRCAIE	CRCBIE	PPWIE	IPAEIE	IPIEIE	0	Reserved	0			TFIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 DLPFIE	Data learning pattern failure interrupt enable Triggered by DLPFF flag in FR. 0b - No DLPFF interrupt is generated. 1b - DLPFF interrupt is generated.
30-29 —	Reserved
28 —	Reserved
27	TX buffer fill interrupt enable flag This field indicates the TX buffer fill interrupt enable flag.

Table continues on the next page...

Table continued from the previous page...

Field	Function
TBFIE	0b - No TBFF interrupt is generated. 1b - TBFF interrupt is generated.
26 TBUIE	TX buffer underrun interrupt enable flag This field indicates the TX buffer underrun interrupt enable flag. 0b - No TBUF interrupt is generated 1b - TBUF interrupt is generated
25 TBFDE	TX buffer fill DMA enable Enables generation of DMA requests for TX buffer fill. When the value of this field is 1, DMA requests are generated as long as SR[TXWA] is set. **NOTE** After you write 1 to this field (to enable DMA transfers), writing 0 does not disable DMA transfers. You must perform a software reset for the AHB domain by using MCR[SWRSTHD] to disable DMA transfers. 0b - No DMA request is generated 1b - DMA request is generated
24 DLLULIE	DLL unlock interrupt enable 1b - Write 1 to this to enable generation of interrupt on DLL unlock event.
23 ILLINIE	Illegal instruction error interrupt enable Triggered by the ILLINE flag in FR 0b - No ILLINE interrupt is generated. 1b - ILLINE interrupt is generated.
22 —	Reserved
21 RBDDE	RX buffer drain DMA enable This field enables generation of DMA requests for RX buffer drain. When the value of this field is 1, the DMA requests are generated as long as SR[RXWE] is set. **NOTE** After you write 1 to this field (to enable DMA transfers), writing 0 does not disable DMA transfers. You must perform a software reset for the AHB domain by using MCR[SWRSTHD] to disable DMA transfers. 0b - No DMA request is generated. 1b - DMA request is generated.
20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-18 —	Reserved
17 RBOIE	RX buffer overflow interrupt enable This field indicates the RX buffer overflow interrupt enable flag. 0b - No RBOF interrupt is generated. 1b - RBOF interrupt is generated.
16 RBDIE	RX buffer drain interrupt enable This field enables generation of IRQ requests for RX buffer drain. When the value of this field is 1, the interrupt is asserted as long as SR[RBDF] is set. 0b - No RBDF interrupt is generated. 1b - RBDF Interrupt is generated.
15 AAIE	AHB abort error interrupt enable Triggered by AAEF flags in FR. 0b - No AAEF interrupt is generated 1b - AAEF interrupt is generated
14 AITIE	AHB illegal transaction interrupt enable flag This field indicates the AHB illegal transaction interrupt enable flag. 0b - No AITEF interrupt is generated. 1b - AITEF interrupt is generated.
13 AIBSIE	AHB illegal burst size interrupt enable flag This field indicates the AHB illegal burst size interrupt enable flag. 0b - No AIBSEF interrupt is generated. 1b - AIBSEF interrupt is generated.
12 ABOIE	AHB buffer overflow interrupt enable flag This field indicates the AHB buffer overflow interrupt enable flag. 0b - No ABOF interrupt is generated. 1b - ABOF interrupt is generated.
11 IUEIE	IP command usage error interrupt enable flag This field indicates the IP command usage error interrupt enable flag.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No IUEF interrupt is generated 1b - IUEF interrupt is generated
10 CRCAIE	CRC and ECC interrupt enable for flash memory A 0b - CRCAEF interrupt is not generated. 1b - CRCAEF interrupt is generated.
9 CRCBIE	Sets when there is CRC or ECC error for flash memory B 0b - CRCBEF interrupt is not generated. 1b - CRCBEF interrupt is generated.
8 PPWIE	Page-program wait interrupt flag This field indicates page-program wait interrupt. 0b - No PPWIE interrupt is generated 1b - PPWIE interrupt is generated
7 IPAEIE	IP command trigger during AHB read access error interrupt enable flag This field indicates IP command trigger during AHB read access error interrupt enable flag. 0b - No IPAEF interrupt is generated 1b - IPAEF interrupt is generated
6 IPIEIE	IP command trigger during IP access error interrupt enable flag This field indicates IP command trigger during IP access error interrupt enable flag. 0b - No IPIEF interrupt is generated 1b - IPIEF interrupt is generated
5 —	Reserved
4 —	Reserved
3-1 —	Reserved
0 TFIE	Transaction finished interrupt enable flag This field indicates the transaction finished interrupt enable flag. 0b - No TFF interrupt is generated. 1b - TFF interrupt is generated.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c0c0e5-4821-48a8-9251-90d3ab624c1

38.4.2.33 Sequence Pointer Clear Register (SPTRCLR)

Offset

Register	Offset
SPTRCLR	16Ch

Function

This register provides fields to reset the IP and buffer sequence pointers. The sequence pointer contains the index of instructions within the LUT entry that is to be executed next. For example, if the LUT entry ends on a JMP_ON_CS value of 2, the index is stored as 2.

The software should reset the sequence pointers whenever the sequence ID is changed by updating the SEQID field in the IPCR or BFGENCR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						0		0				0		PREFETC...	ABRT_CLR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							0	0							0
W								IPPTRC								BFPTRC
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25-24 —	Reserved
23-20 —	Reserved
19-18 —	Reserved
17	Prefetch disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
PREFETCH_DIS	This field should be configured to disable the prefetch mechanism of receiver. It is not based on dynamic programming. Therefore, it should be programmed initially at once. When this field is set, then during an ongoing flash memory read, any subsequent AHB read is checked for buffer hit. However, after the end of flash memory read, as soon as chip select is deasserted, any subsequent AHB read results in flushing of the current AHB buffer data and issues fresh flash memory transaction if the AHB buffer data is not updated with the flash memory.
16 ABRT_CLR	Flash memory Abort/AHB buffer clear This is a dynamic field. Writing a 1 to it, irrespective of the prefetch disable, clears the AHB buffer pointers and also aborts any ongoing flash memory transaction (if any) and rejects any ongoing AHB read with an error response (if any). QuadSPI sets this field to 0 after clearing the AHB buffer pointers.
15-9 —	Reserved
8 IPPTRC	IP pointer clear This is a self-clearing field. 1b - Clears the sequence pointer for IP accesses as defined in IPCR.
7-1 —	Reserved
0 BFPTRC	Buffer pointer clear This is a self-clearing field. 1b - Clears the sequence pointer for AHB read accesses as defined in BFGENCR.

38.4.2.34 Serial Flash Memory A1 Top Address Register (SFA1AD)

Offset

Register	Offset
SFA1AD	180h

Function

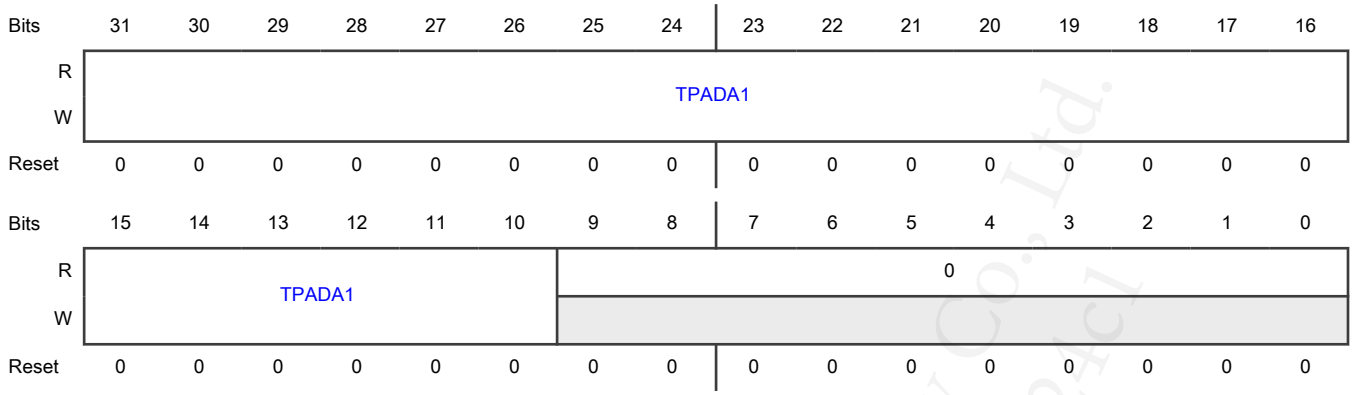
This register provides the address mapping for serial flash memory A1. The difference between SFA1AD[TPADA1] and AMBA_BASE defines the size of the memory map for serial flash memory A1.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c40e5-4821-48-8-b251390d3ab604c1

Diagram



Fields

Field	Function
31-10 TPADA1	Top address for serial flash memory A1 In effect, TPADxx is the first location of the next memory.
9-0 —	Reserved

38.4.2.35 Serial Flash Memory A2 Top Address Register (SFA2AD)

Offset

Register	Offset
SFA2AD	184h

Function

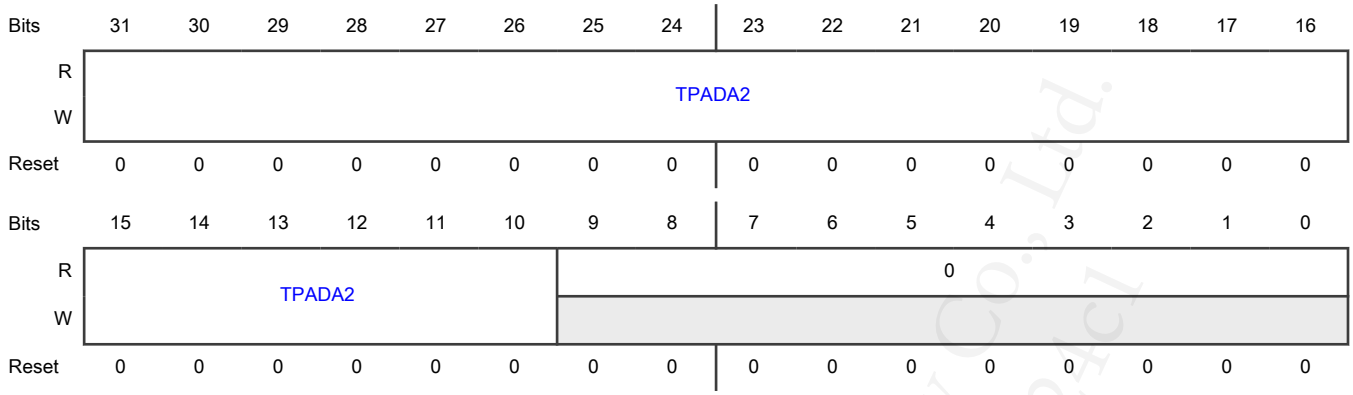
This register provides the address mapping for serial flash memory A2. The difference between SFA2AD[TPADA2] and SFA1AD[TPADA1] defines the size of the memory map for serial flash memory A2.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-10 TPADA2	Top address for serial flash memory A2 In effect, TPxxAD is the first location of the next memory.
9-0 —	Reserved

38.4.2.36 Serial Flash Memory B1 Top Address Register (SFB1AD)

Offset

Register	Offset
SFB1AD	188h

Function

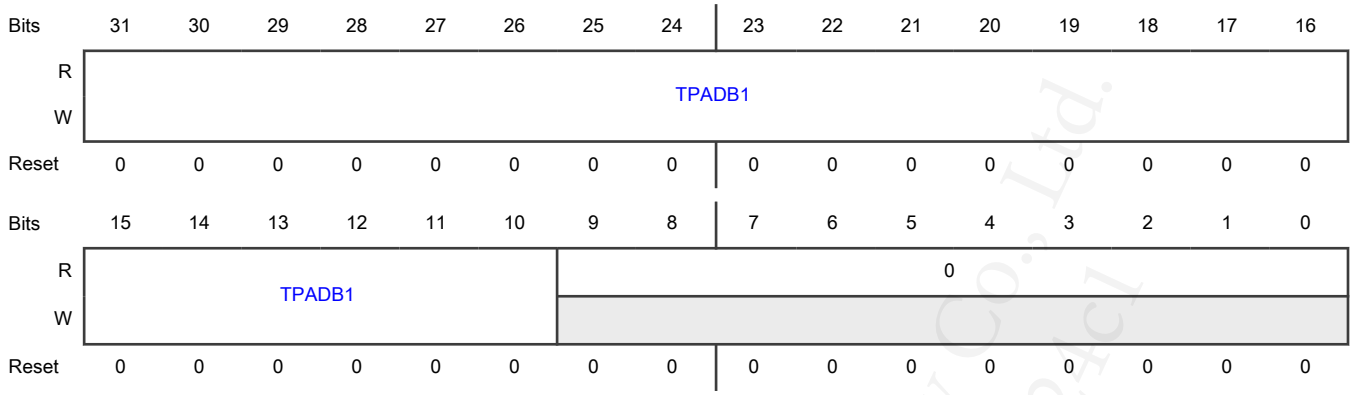
This register provides the address mapping for serial flash memory B1. The difference between SFB1AD[TPADB1] and SFA2AD[TPADA2] defines the size of the memory map for serial flash memory B1.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-10 TPADB1	Top address for serial flash memory B1. In effect, TPxxAD is the first location of the next memory.
9-0 —	Reserved

38.4.2.37 Serial Flash Memory B2 Top Address Register (SFB2AD)

Offset

Register	Offset
SFB2AD	18Ch

Function

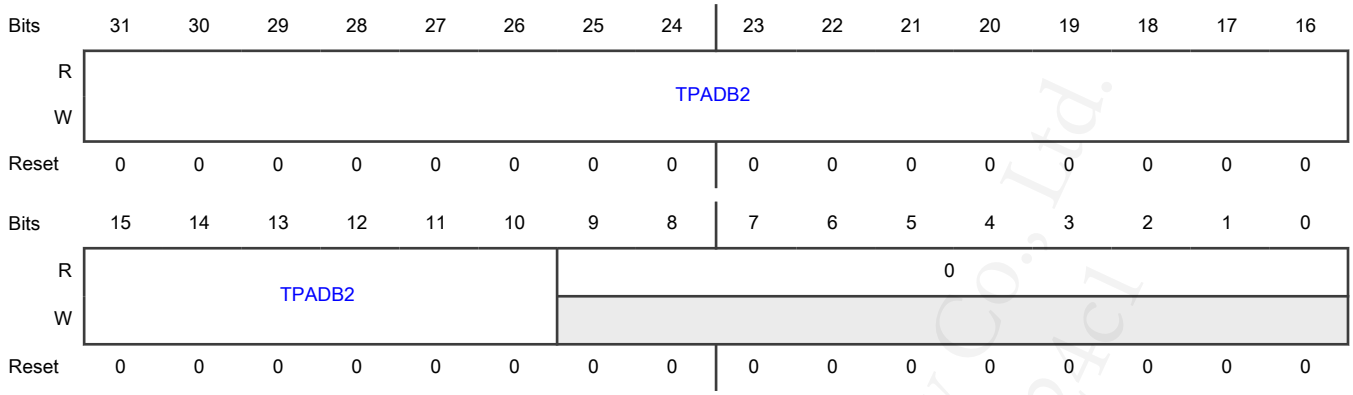
This register provides the address mapping for serial flash memory B2. The difference between SFB2AD[TPADB2] and SFB1AD[TPADB1] defines the size of the memory map for serial flash memory B2.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-10 TPADB2	Top address for serial flash memory B2. In effect, TPxxAD is the first location of the next memory.
9-0 —	Reserved

38.4.2.38 Data Learn Pattern Register (DLPR)

Offset

Register	Offset
DLPR	190h

Function

This register contains the information of the data to be used for data learning.

Special write-access is permitted if:

- SR[IP_ACC] = 0
- SR[AHB_ACC] = 0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLPV															
W																
Reset	1	0	1	0	1	0	1	0	0	1	0	1	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLPV															
W																
Reset	0	0	1	1	0	1	0	0	0	1	0	0	0	0	1	1

Fields

Field	Function
31-0 DLPV	Data learning pattern value This value is used for data learning in SDR/DDR modes along with DATA_LEARN instruction. Different patterns can be matched at selected IOs. That is, IO1 and IO3 using DLCR[DLP_SEL_FA] and DLCR[DLP_SEL_FB], DLPR[31:16] for IO3, and DLPR[15:0] for IO1. DLP pattern for IO1 and IO3 should be stored, bit-wise, in the little endian format. For details, see Data learning.

38.4.2.39 Flash Memory A Failing Address Status Register (FAILA_ADDR)

Offset

Register	Offset
FAILA_ADDR	194h

Function

This register provides the flash memory address where data learning or parity error has occurred.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ADDR															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-32d3ab624c1

Fields

Field	Function
31-0 ADDR	Failing address for flash memory A

38.4.2.40 flash Memory B Failing Address Status Register (FAILB_ADDR)

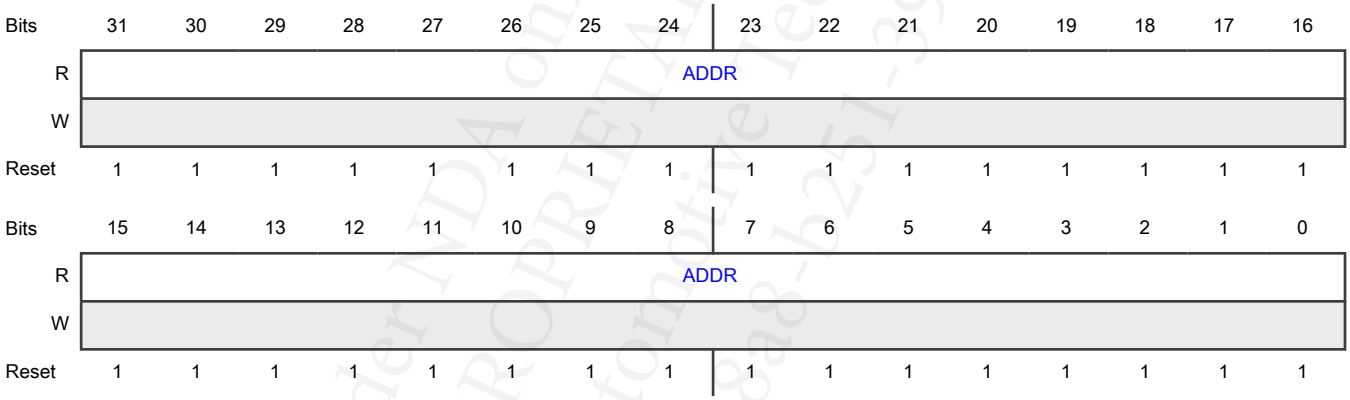
Offset

Register	Offset
FAILB_ADDR	198h

Function

This register provides the flash memory address where Data Learning or Parity error has occurred

Diagram



Fields

Field	Function
31-0 ADDR	Failing address for flash memory B

38.4.2.41 RX Buffer Data Register (RBDR0 - RBDR63)

Offset

For a = 0 to 63:

Register	Offset
RBDRa	200h + (a × 4h)

Function

These registers provide access to individual entries in the RX buffer. See Table 220 for the byte ordering scheme.

RBDR0 corresponds to the actual position of the read pointer within the RX buffer. The number of valid entries available depends on the number of RX buffer entries implemented and on the number of valid buffer entries available in the RX buffer.

Example 1 - RX buffer filled completely with 128 words: In this case, the address range for valid read access extends from RBDR0 to RBDR63.

Example 2 - RX buffer filled with five valid words: RX buffer fill level of RBSR[RDBFL] is 5. In this case, access to RBDR4 provides the last valid entry.

Any access beyond the range of valid RX buffer entries provides undefined results.

NOTE

To access data beyond RBDR[63], you must pop the data by using FR[RBDF]. See here for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 RXDATA	RX data This field contains the data associated with the related RX buffer entry. For data format and byte ordering, see Byte ordering of serial flash memory read data.

38.4.2.42 LUT Key Register (LUTKEY)

Offset

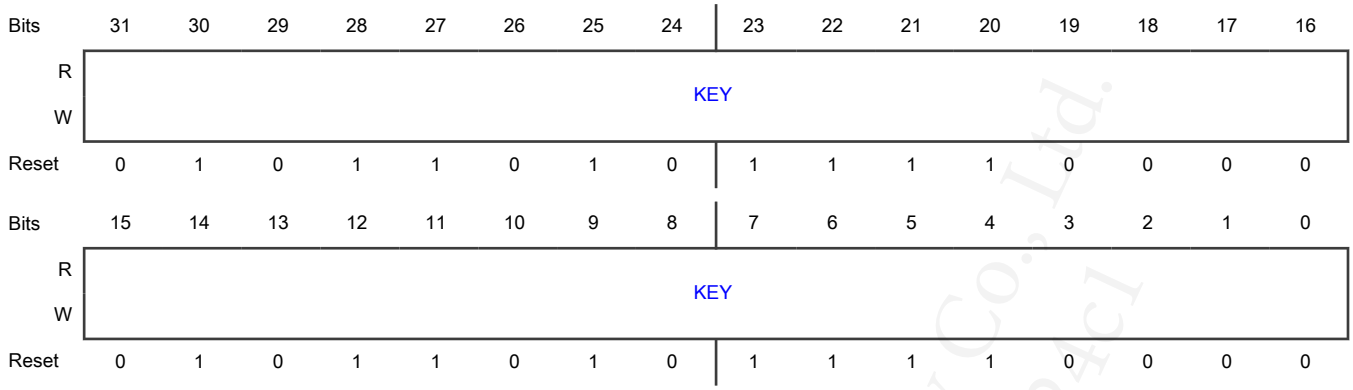
Register	Offset
LUTKEY	300h

Function

This register contains the key to lock and unlock the LUT. See LUT for details.

Special write-access is permitted in the "Anytime" mode.

Diagram



Fields

Field	Function
31-0 KEY	Key to lock or unlock the LUT The key is 0x5AF05AF0 and the read value is always 0x5AF05AF0.

38.4.2.43 LUT Lock Configuration Register (LCKCR)

Offset

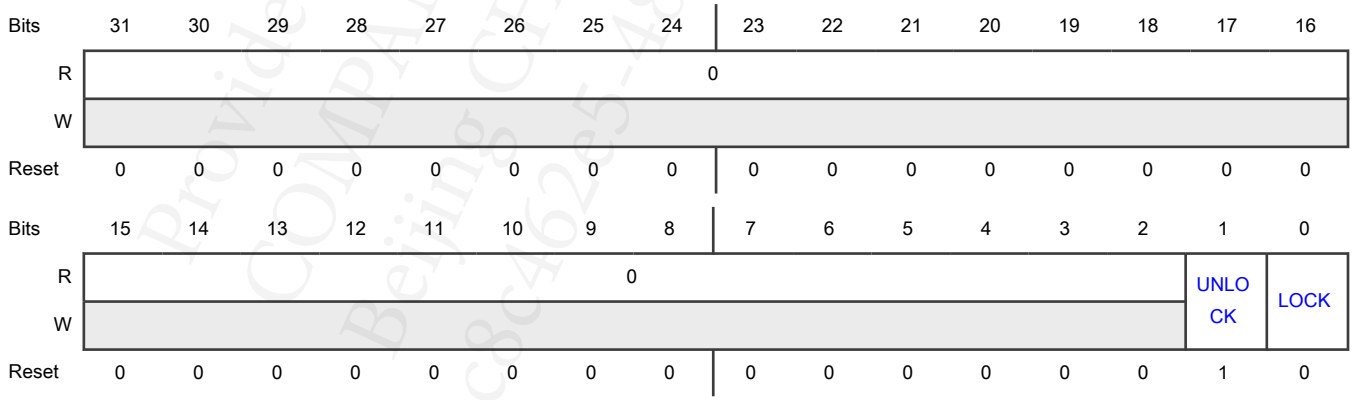
Register	Offset
LCKCR	304h

Function

This register is used along with the LUTKEY register to lock or unlock the LUT. This register should be written immediately after the LUTKEY register for the lock or unlock operation to be successful. See LUT for details. Setting both the LOCK and UNLOCK bits as "00" or "11" is not allowed.

Special write access is permitted after writing the LUT key register.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e25-4821-4ea8-b251-390d3ab624c1

Fields

Field	Function
31-2 —	Reserved
1 UNLOCK	Unlock LUT Unlocks the LUT when the following two conditions are met: • This register is written just after the LUT Key Register (LUTKEY). • The LUT key register was written with the 0x5AF05AF0 key.
0 LOCK	Lock LUT Locks the LUT when the following conditions are met: • This register is written just after the LUT Key Register (LUTKEY). • The LUT key register is written with the 0x5AF05AF0 key.

38.4.2.44 LUT Register (LUT0)

Offset

Register	Offset
LUT0	310h

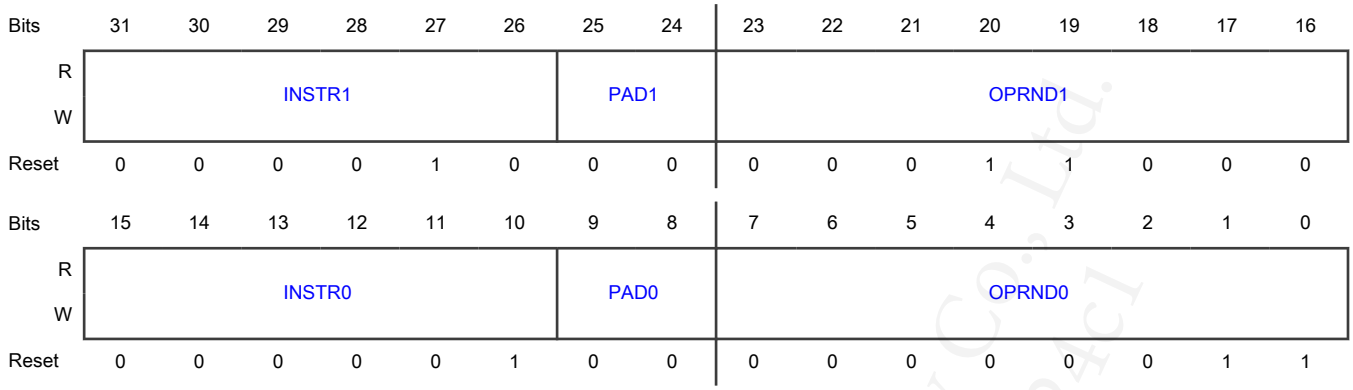
Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 80 LUT registers. These 80 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[75] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT79. A maximum of 16 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482... -b251-390d3ab624c1

Diagram



Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - 8 Pads
23-16 OPRND1	Operand for INSTR1
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - 8 Pads
7-0 OPRND0	Operand for INSTR0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

38.4.2.45 LUT Register (LUT1)

Offset

Register	Offset
LUT1	314h

Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 80 LUT registers. These 80 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[75] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT79. A maximum of 16 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INSTR1						PAD1		OPRND1							
Reset	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INSTR0						PAD0		OPRND0							
Reset	0	0	0	1	1	1	0	0	0	0	0	0	1	0	0	0

Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - 8 Pads
23-16	Operand for INSTR1

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3a6...d1

Table continued from the previous page...

Field	Function
OPRND1	
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - 8 Pads
7-0 OPRND0	Operand for INSTR0

38.4.2.46 LUT Register (LUT2 - LUT79)

Offset

For a = 2 to 79:

Register	Offset
LUTa	310h + (a × 4h)

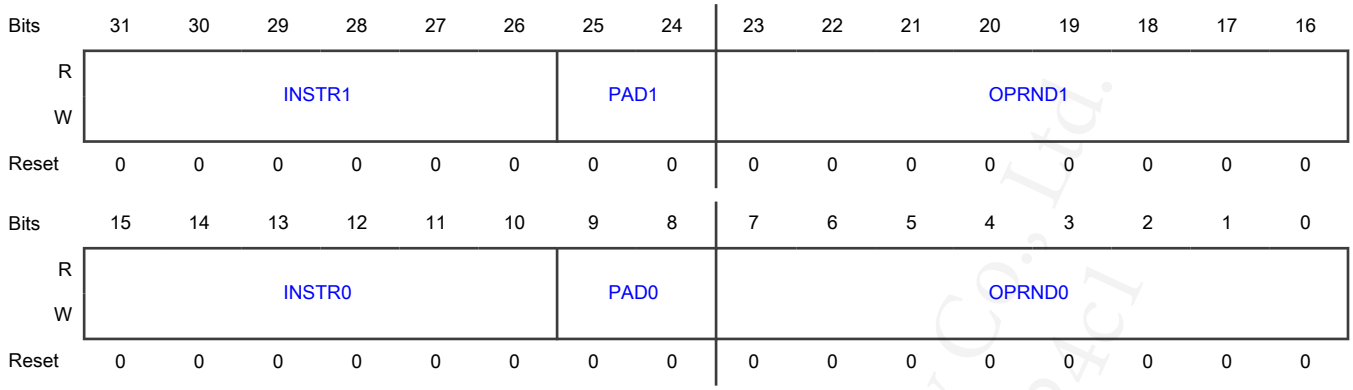
Function

A sequence of instruction-operand pairs may be pre-populated in the LUT according to the device connected on board. Each instruction-operand pair is of 16 bits (2 bytes) each. Every sequence preprogrammed by Program Sequence Engine in the LUT is referred to by its index. The LUT registers act as lookup tables for sequences of instructions. The programmable sequence engine executes the instructions in these sequences to generate a valid serial flash memory transaction. There are a total of 80 LUT registers. These 80 registers are divided into groups of 5 registers that make a valid sequence. Therefore, LUT[0], LUT[5], LUT[10] ..... LUT[75] are the starting registers of a valid sequence. Each of these sets of 5 registers can have a maximum of 10 instructions. Reset value of the register shown below is only applicable to LUT2 to LUT79. A maximum of 16 sequences can be defined at one time. See LUT that describes the LUT registers in detail.

Special write-access is permitted if the LUT is unlocked.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHTC Automotive Technology Co., Ltd.
c8c462e5-4f68-4ca8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-26 INSTR1	Instruction 1
25-24 PAD1	Pad information for INSTR1 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - 8 Pads
23-16 OPRND1	Operand for INSTR1
15-10 INSTR0	Instruction 0
9-8 PAD0	Pad information for INSTR0 00b - 1 Pad 01b - 2 Pads 10b - 4 Pads 11b - 8 Pads
7-0 OPRND0	Operand for INSTR0

38.4.3 Serial flash memory address assignment

The serial flash memory address assignment can be modified by writing into Serial Flash Memory A1 Top Address Register (SFA1AD) and Serial Flash Memory A2 Top Address Register (SFA2AD) for device A and into Serial Flash Memory B1 Top Address Register (SFB1AD) and Serial Flash Memory B2 Top Address Register (SFB2AD) for device B. The following table shows how different access modes are related to the address specified for the next SFM command. Note that this address assignment is valid for both IP and AHB commands.

Table 212. Serial flash memory address assignment

Parameter	Function	Access mode
QuadSPI_AMBA_BASE ((31:10) - 22 bits)	QuadSPI AHB base address First address of the serial flash memory device as presented to the QuadSPI controller. This might be the base of the serial flash memory in the system address map or it may be a remapping, for instance to 0h, performed by the system. (See the system address map file attached to this document)	
QuadSPI_ARDB_BASE	First address of the QuadSPI Rx buffer on system memory map	
TOP_ADDR_MEMA1(TPADA1)	Top address for the external flash memory A1 (first device of the dual-die flash memory A, or the first of the two independent flash memories sharing the IOFA)	Any access to the address space between TOP_ADDR_MEMA1 and QuadSPI_AMBA_BASE is routed to serial flash memory A1.
TOP_ADDR_MEMA2(TPADA2)	Top address for the external flash memory A2 (second device of the dual-die flash memory A, or the second of the two independent flash memories sharing the IOFA)	Any access to the address space between TOP_ADDR_MEMA2 and TOP_ADDR_MEMA1 is routed to serial flash memory A2.
TOP_ADDR_MEMB1(TPADB1)	Top address for the external flash memory B1 (first device of the dual-die flash memory B, or the first of the two independent flash memories sharing the IOFB)	Any access to the address space between TOP_ADDR_MEMB1 and TOP_ADDR_MEMA2 is routed to serial flash memory B1.
TOP_ADDR_MEMB2(TPADB2) b	Top address for the external flash memory B2 (second device of the dual-die flash memory B or the second of the two independent flash memories sharing the IOFB)	Any access to the address space between TOP_ADDR_MEMB2 and TOP_ADDR_MEMB1 is routed to serial flash memory A2.

38.5 Flash memory mapped AMBA bus

QuadSPI_AMBA_BASE defines the address to be used as the start address of the serial flash memory device, as defined by the system memory map. Note that this may be a remapping of the physical address of the serial flash memory in the system. See the system address map for details.

Table 213. QuadSPI AMBA bus memory map

Address	Register name
QuadSPI_AMBA_BASE to (TOP_ADDR_MEMA2 - 1h)	• See Memory-mapped serial flash memory data—individual flash memory mode on flash memory A. • For information about byte ordering, see Table 220 and Table 225.
TOP_ADDR_MEMA2 to (TOP_ADDR_MEMB2 - 1h)	• See Memory-mapped serial flash memory data—individual flash memory mode on flash memory B. • For information about the byte ordering, see Table 220 and Table 225.
QuadSPI_AMBA_BASE to (TOP_ADDR_MEMB2 - 1h)	• See Parallel flash memory mode. • For information about the byte ordering, see Table 224 and Table 225.
QuadSPI_ARDB_BASE to... (32 * 4 Byte) QuadSPI_ARDB_BASE + 1FFh	• See AHB RX Data Buffer Register (ARDB0 - ARDB127). • For information about the byte ordering, see Table 220 and Table 222.

NOTE

Any read access to non-implemented addresses provides undefined results.

In case of single-die flash memory devices, TOP_ADDR_MEMA2 and TOP_ADDR_MEMB2 should be initialized/programmed to TOP_ADDR_MEMA1 and TOP_ADDR_MEMB1, respectively (in effect, setting the size of these devices to 0). This ensures that the complete memory map is assigned to only one flash device.

The parallel flash memory mode is valid only for commands related to data read and data write in single I/O mode from serial flash memory. The first device of flash memory A needs to be paired with the first device of flash memory B and the second device of flash memory A has to be paired with the second device of flash memory B in parallel mode. This mode is selected via BFGENCR[PAR_EN] for all masters in the AHB driven mode and through IPCR[PAR_EN] in the IP-driven mode. In the parallel mode, the incoming address (SFAR address in case of IP-initiated transactions and the incoming AHB address in case of AHB-initiated transactions) is divided by 2 and sent to the two flash memories connected in parallel.

Any IP command other than data read and write (through one pad) in the parallel flash memory mode results in the assertion of FR[IUEF].

In individual flash memory modes, the 3/4 address bytes (as programmed in the instruction/operand in the sequence) available for the flash memory address are determined by SFADR[23:0] or SFADR[31:0] as provided in the table shown above.

In parallel flash memory mode, both flash memories are read with the same starting address of 3/4 (as programmed in the instruction/operand in the sequence) bytes in size. This address is derived from SFADR[24:1] or SFADR[31:1] as provided in the table above. The LSB of the SFADR field is used to select the appropriate bits of both the flash memory devices to combine the bytes corresponding to the selected address.

38.5.1 AHB bus access read considerations

Note that all logic in the QuadSPI module implementing the AHB bus access is designed to read the content of an external serial flash memory device. Therefore, the following restrictions apply to the QuadSPI module with respect to accesses to the AHB bus:

- AHB bus read access types—NONSEQ and BUSY—are fully supported.
- AHB read access type—SEQ—is treated in the same way as NONSEQ. See Flash memory mapped AMBA bus for details.

- Early burst termination is not supported for AHB transactions.
- An AHB error response is provided on occurrence of:
  - Parity error
  - ECC error from flash memory
  - Data learning failure

  Also, interrupts (if enabled) are provided along with the AHB error response.

38.5.2 Memory-mapped serial flash memory data—individual flash memory mode on flash memory A

Starting with address QuadSPI_AMBA_BASE, the content of the first external serial flash memory device is mapped into the address space of the device containing the QuadSPI module. Serial flash memory byte address 0h corresponds to bus address, QuadSPI_AMBA_BASE, in an increasing order. Assuming that a dual-die flash memory is connected to the first set of external pads, the address space is divided into two parts, one for each device of the dual-die package. See the following table for the address mapping. The byte ordering for 32-bit access is provided in Table 220 and for 64-bit read access, the byte ordering is provided in Table 225.

Table 214. Memory-mapped individual flash memory mode—flash memory A address scheme

Memory-mapped address 32-bit access	Memory-mapped address 64-bit access	Serial flash memory byte address	Flash memory device
QuadSPI_AMBA_BASE + 0h	QuadSPI_AMBA_BASE + 0h	0h to 3h	A1
QuadSPI_AMBA_BASE + 4h		4h to 7h	
...	...	...	
TOP_ADDR_MEMA1 - 8h	TOP_ADDR_MEMA1 - 8h	(TOP_ADDR_MEMA1- 8h) to (TOP_ADDR_MEMA1 - 0h4 -0h1)	
TOP_ADDR_MEMA1 - 4h		(TOP_ADDR_MEMA1 - 4h) to (TOP_ADDR_MEMA1 - 0h1)	
TOP_ADDR_MEMA1 + 4h	TOP_ADDR_MEMA1 + 0h	0h to 3h	A2
TOP_ADDR_MEMA1 + 0h4		4h to 7h	
...	...	...	
TOP_ADDR_MEMA2 - 8h	TOP_ADDR_MEMA2 - 8h	(TOP_ADDR_MEMA2 - 8h) to (TOP_ADDR_MEMA2 - 4h - 1h)	
TOP_ADDR_MEMA2 - 4h		(TOP_ADDR_MEMA2 - 4h) to (TOP_ADDR_MEMA2 - 1h)	

The available address range depends on the size of the external serial flash memory device. Any access beyond the size of the external serial flash memory provides undefined results.

For details concerning the read process, see Flash memory read.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJU Automotive Technology Co., Ltd.
c8e42e5-4921-48a8-b251-390d3ab0...

38.5.3 Memory-mapped serial flash memory data—individual flash memory mode on flash memory B

Starting with address TOP_ADDR_MEMA2, the content of the first external serial flash memory devices is mapped into the address space of the device containing the QuadSPI module. Serial flash memory address byte address 0h corresponds to bus address, TOP_ADDR_MEMA2, in an increasing order. Assuming that a dual-die flash memory is connected to the first set of external pads, the address space is divided into two parts, one for each device of the dual-die package. See the following table for address mapping. The byte ordering for 32-bit access is provided in Table 220 and for 64-bit read access, the byte ordering is provided in Table 225.

Table 215. Memory-mapped individual flash memory mode—flash memory B address scheme

Memory-mapped address 32-bit access	Memory-mapped address 64-bit access	Serial flash memory byte address	Flash memory device
TOP_ADDR_MEMA2 + 0h	TOP_ADDR_MEMA2 + 0h	0h to 3h	B1
TOP_ADDR_MEMA2 + 0h4		0h_0004 to 0h_0007	
...	...	...	
TOP_ADDR_MEMB1 - 0h8	TOP_ADDR_MEMB1 - 0h8	(TOP_ADDR_MEMB1- TOP_ADDR_MEMA2 - 0h8) to (TOP_ADDR_MEMB1 - TOP_ADDR_MEMA2 - 0h4 -0h1)	
TOP_ADDR_MEMB1 - 0h4		(TOP_ADDR_MEMB1- TOP_ADDR_MEMA2 - 0h4) to (TOP_ADDR_MEMB1- TOP_ADDR_MEMA2- 0h1)	
TOP_ADDR_MEMB1 + 0h	TOP_ADDR_MEMB1 + 0h	0 to 3h	B2
TOP_ADDR_MEMB1 + 0h4		0h_0004 to 0h_0007	
...	...	...	
TOP_ADDR_MEMB2 - 0h8	TOP_ADDR_MEMA2 - 0h8	(TOP_ADDR_MEMB2 - TOP_ADDR_MEMB1- 0h8) to (TOP_ADDR_MEMB2 - TOP_ADDR_MEMB1 - 0h4 - 0h1)	
TOP_ADDR_MEMB2 - 0h4		(TOP_ADDR_MEMB2- TOP_ADDR_MEMB1 - 0h4) to (TOP_ADDR_MEMB2 - TOP_ADDR_MEMB1 - 0h1)	

The available address range depends on the size of the external serial flash memory device. Any access beyond the size of the external serial flash memory provides undefined results.

For details concerning the read process, see Flash memory read.

38.5.4 Parallel flash memory mode

Any of the AHB flexible buffers can be configured to work in the parallel flash memory mode by programming the value for BFGENCR[PAR_EN] to 1. When the parallel mode is set, flash memory A1 is paired with flash memory B1 and flash memory A2

is paired with flash memory B2. In the parallel mode, the software should ensure that the size of flash memory A1(A2) is equal to the size of flash memory B1(B2).

Reads from any even AHB bus address provide bits [7:4] of both serial flash memory devices and reads from any odd AHB bus address provide bits [3:0] of both flash memory devices. See the following table for address mapping. The byte ordering for 32-bit access Table 222and 64-bit read access is provided in Table 225.

Table 216. Memory-mapped parallel flash memory mode address scheme

Memory-mapped address 32-bit access	Memory-mapped address 64-bit access	Serial flash memory A byte address	Serial flash memory B byte address
QuadSPI_AMBA_BASE + 0h	QuadSPI_AMBA_BASE + 0h For details, see Parallel mode and Dual-die flash memories.	0h - 1h	0h - 1h
QuadSPI_AMBA_BASE + 4h		2h - 3h	2h - 3h
QuadSPI_AMBA_BASE + 8h	QuadSPI_AMBA_BASE + 8h	4h - 5h	4h - 5h
QuadSPI_AMBA_BASE + Ch		6h - 7h	6h - 7h
...	...	...	...
TOP_ADDR_MEMB2 - 8h	TOP_ADDR_MEMB2 - 8h	(TOP_ADDR_MEMB2 - QuadSPI_AMBA_BASE - 8h)/2	(TOP_ADDR_MEMB2 - QuadSPI_AMBA_BASE - 8h)/2
TOP_ADDR_MEMB2 - 4h		(TOP_ADDR_MEMB2 - QuadSPI_AMBA_BASE - 4h)/2 +1h	(TOP_ADDR_MEMB2 - QuadSPI_AMBA_BASE- 4h)/2 + 1h

The available address range covers 27 address bits, corresponding to 128 MB per flash memory device. The usable space depends on the size of the external serial flash memory devices. Any access beyond the size of the external serial flash memory provides undefined results.

For details concerning the read process, see Flash memory read.

38.5.5 ARDB register descriptions

NOTE

See the system memory map in this document for the base address of the QuadSPI AHB RX data buffer.

38.5.5.1 ARDB memory map

QuadSPI_ARDB base address: 4100_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h - 1FCh	AHB RX Data Buffer Register (ARDB0 - ARDB127)	32	RO	0000_0000h

38.5.5.2 AHB RX Data Buffer Register (ARDB0 - ARDB127)

Offset

For a = 0 to 127:

Register	Offset
ARDBa	0h + (a × 4h)

Function

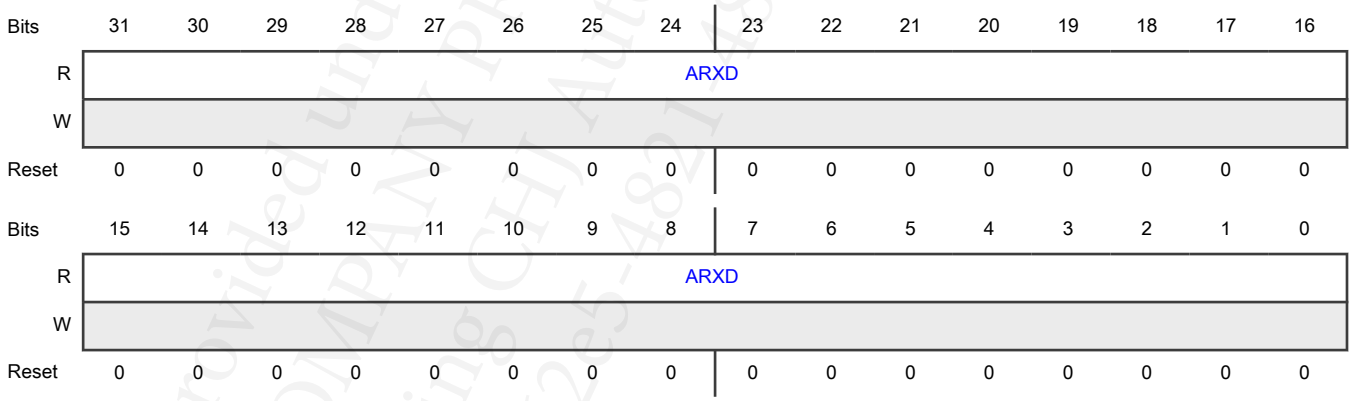
This register is used to read the buffer content of the RX buffer from successive addresses. ARDB0 corresponds to the RX buffer register entry corresponding to the current value of the read pointer in an increasing order.

The increment of the read pointer depends on the access scheme (DMA or flag-driven). See Flash memory read, RX buffer, data read through register interface, and AHB read for the description of successive accesses to the RX buffer content. See Byte Ordering of Serial Flash Memory Read Data for the byte ordering scheme.

Valid address range accessible in the ARDB*n* range depends on the number of RX buffer entries implemented and on the number of valid buffer entries available in the RX buffer.

- Example 1 - RX buffer filled completely with 128 words: In this case, the address range for valid read access extends from ARDB0 to ARDB127.
- Example 2 - RX buffer filled with five valid words; RX buffer fill level RBSR[RDBFL] is 5. In this case, an access to ARDB4 provides the last valid entry.

Diagram



Provided under IHS COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-4821-48a8-b251-3c03ab624c1

Fields

Field	Function
31-0 ARXD	ARDB provided RX buffer data Byte order (endianness) is identical to the RX buffer data registers.

38.6 Functional description

This section provides a functional description of the QuadSPI module.

38.6.1 Serial flash memory access schemes

The following access schemes are possible, depending on the serial flash devices attached to the QuadSPI module.

Table 217. Access schemes for serial flash memory data access

Access scheme	One flash memory device on port A	One flash memory device on port B	Two identical flash memory devices connected to port A and port B
Individual flash memory mode: access to flash memory A	**Yes**	**N/a**	**Yes**
Individual flash memory mode: access to flash memory B	**N/A**	**Yes**	**Yes**
Parallel flash memory mode: read from flash memory A and flash memory B	**N/A**	**N/A**	**Yes**

If two flash memory devices are accessed in the parallel flash memory mode, they are accessed with identical control signals. Special alignment on per-flash memory basis is not possible. The application needs to ensure that the identical signals are applicable to both the flash memory devices.

In the parallel flash memory mode, both the external serial flash memory devices appear as a single memory doubled in size with respect to one individual flash memory device.

If two different flash memory devices are attached, they can be operated only in the individual flash memory mode.

In the parallel flash memory mode, only data read commands and write command (in x1 mode) are supported. Any other IP command results in an error condition signaled by the assertion of FR[IUEF].

In the individual flash memory mode, all supported commands are available.

Unless explicitly noted, all the following descriptions relate to the individual flash memory mode.

38.6.2 Normal mode

This mode allows communication with an external serial flash memory device. Compared to the standard SPI protocol, this communication method uses up to four bidirectional data lines operating at high-data rates. The communication to the external serial flash memory device consists of an instruction code and optional address, mode, dummy, and data transfers. The flexible programmable core engine described below is immune to a wide variety of command or protocol differences in the serial flash memory devices provided by various flash memory vendors.

Beijing Semiconductor Automotive Technology Co., Ltd.
c8c2e54a-1251-390d-b624c1
COMPANY PROPRIETARY
Under NDA only

38.6.2.1 Programmable sequence engine

The core of the QuadSPI module is a programmable sequence engine that works on "instruction-operand" pairs. The core controller executes each programmed instruction sequentially. The complete list of instructions and the corresponding operands are provided in the following table.

Table 218. Instruction set

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
CMD	1d	N={0,1,2,3}d 0d - One pad 1d - Two pads 2d - Four pads 3d- Eight pads	8-bit command value	Provide the serial flash memory with operand on the number of pads specified
ADDR	2d		Number of address bits to be sent (for example, 24d => 24 address bits required)	Provide the serial flash memory with address cycles according to the operand on the number of pads specified The actual address to be provided is derived from the incoming address in case of AHB-initiated transactions and the value of SFAR in case of IPS-initiated transactions, if the value of SFACR[CAS] is set to 0. Otherwise, the actual address takes CAS into consideration.
DUMMY	3d		Number of dummy clock cycles (should be <= 64 and > 2 cycles)	Provide the serial flash memory with dummy cycles according the operand The PAD information defines the number of pads in input mode. For example, one pad implies that pad 1 is not driven, rest all are driven. **NOTE** If DLL is enabled and N dummy cycles are needed, you must program two back-to-back DUMMY instructions: DUMMY: N-2 followed by DUMMY:2.
MODE	4d		8-bit mode value	Provide the serial flash memory with 8-bit operand on the number of pads specified
MODE2	5d	N={0,1}d	2-bit mode value	Provide the serial flash memory with 2-bit operand on the number of pads[1] specified
MODE4	6d	N={0,1,2}d	4-bit mode value	Provide the serial flash memory with 4-bit operand on the number of pads[2] specified
READ	7d	N={0,1,2,3}d 0d - One pad 1d - Two pads 2d - Four pads 3d- Eight pads	Read data size in bytes (for AHB transactions, your application should ensure that data size	Read data from flash memory on the number of pads specified The data size can be overwritten by writing to the ADATSZ field of the BUFxCR registers for AHB-initiated transactions and to the IDATSZ field of the IP Configuration Register (IPCR) for IPS-initiated transactions.

Table continues on the next page...

Table 218. Instruction set (continued)

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
			is a multiple of 8 bytes)	
WRITE	8		Write data size in bytes	Write data on the number of pads specified The data size can be overwritten by writing to the IDATSZ field of IP Configuration Register (IPCR).
JMP_ON_CS	9d	NA	Instruction number	Every time the CS is deasserted, jump to the instruction pointed to by the operand. This instruction allows the programmer to specify the behavior of the controller when a new read transaction is initiated following a CS deassertion. **NOTE** This instruction is not supported if DLL is enabled.
ADDR_DDR	10d	N={0,1,2,3}d 0d - One pad 1d - Two pads 2d - Four pads 3d- Eight pads	Number of address bits to be sent (for example, 24d => 24 address bits required)	Provide the serial flash memory with address cycles according to the operand on the number of pads specified at each clock edge of serial flash memory clock. The actual address to be provided is derived from the incoming address in case of AHB-initiated transactions and the value of SFAR in case of IPS-initiated transactions if SFACR[CAS] is set to 0. Otherwise, the actual address takes CAS into consideration.
MODE_DDR	11d		8-bit mode value	Provide the serial flash memory with 8-bit operand on the number of pads specified at each clock edge of serial flash memory
MODE2_DDR	12d	N={0}d	2-bit mode value	Provide the serial flash memory with 2-bit operand on the number of pads specified at each clock edge of serial flash memory [3]
MODE4_DDR	13d	N={0,1}d	4-bit mode value	Provide the serial flash memory with 4-bit operand on the number of pads specified at each clock edge of serial flash memory [4].
READ_DDR	14d	N={0,1,2,3}d 0d - One pad 1d - Two pads 2d - Four pads 3d - Eight pads	Read data size in bytes (for AHB transactions, the application should ensure that data size is in multiple of 8 bytes)	Read data from flash memory on the number of pads specified at each clock edge of serial flash memory. The data size might be overwritten by writing to the ADATSZ field of the BUFxCR registers for AHB-initiated transactions and IDATSZ field of IP Configuration Register (IPCR) for IP initiated transactions.

Table continues on the next page...

Table 218. Instruction set (continued)

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
WRITE_DDR	15d		Write data size in bytes	Write data on the number of pads specified at each clock edge of serial flash memory The data size can be overwritten by writing to the IDATSZ field of IP Configuration Register (IPCR).
DATA_LEARN	16d	N={0,1,2,3}d 0d - One pad 1d - Two pads 2d - Four pads 3d- Eight pads	Operand[7]- • 1 - DDR mode • 0 - SDR mode Operand[6:0]- • Number of bits to be used for data learning. For example, if operand is 7, then a 7-bit data_learning pattern is read and compared.	Find the correct sampling point with the data learning pattern. When this instruction is encountered, the value of Sampling Register (SMPR) is ignored and the controller finds the correct sampling point on its own by sampling the data learning pattern.[5]
CMD_DDR	17d	N={0,1,2,3}d 0d - One pad 1d - Two pads 2d - Four pads 3d- Eight pads	8-bit command value	Provide the serial flash memory with the operand with number of pads specified at each clock edge of the serial flash memory.
CADDR	18d		Number of column address bits to be sent (8d means 8 bits of column address is to be sent)	Provide the serial flash memory with column address cycles according to the operand on the number of pads specified. The actual address to be provided to flash memory depends on the value of SFACR[CAS]. For example, if the value of SFACR[CAS] is 3, then the address to flash memory will be [2:0] of incoming address in case of AHB and the value of SFAR in case of IP. This is appended with 0 if SFACR[CAS] is less than number of pads for a flash memory.
CADDR_DDR	19d		Number of column address bits to be sent(8d means 8 bits	Provide the serial flash memory with column address cycles according to the operand on the number of pads specified at each clock edge of the serial flash memory. The actual address to be provided to flash memory depends on the value of SFACR[CAS]. For example, if CAS is 3, then the address to

Table continues on the next page...

Table 218. Instruction set (continued)

Instruction	Instruction encoding	Pins	Operand	Action on serial flash memories
			of column address is to be sent)	flash memory will be [2:0] of incoming address in case of AHB and the value of SFAR in case of IP. This is appended with 0 if CAS is less than the number of pads for a flash memory.
JMP_TO_SEQ	20d	NA	Sequence number	Every time CS is deasserted, jump to the LUT sequence is pointed to by the operand. This instruction allows the programmer to join two sequences and initiates the second flash memory transaction automatically. It can be used to join write-enables with flash memory write or data learn with read. **NOTE** This instruction is not supported if DLL is enabled.
STOP	0d	NA	NA	Stop execution; deassert CS

1. For a one-pad instruction, MODE2 takes two serial flash memory clock cycles on the flash memory interface.
2. For a one-pad instruction, MODE4 takes four serial flash memory clock cycles on the flash memory interface. For a four-pad instruction, MODE4 takes one serial flash memory clock cycle on the flash memory interface.
3. For a one-pad instruction, MODE2_DDR takes one serial flash memory clock cycle on the flash memory interface.
4. For a one-pad instruction, MODE4_DDR takes two serial flash memory clock cycles on the flash memory interface. For a four-pad instruction MODE4_DDR takes half a cycle on the serial flash memory interface.
5. It is not recommended to have 0h or FFh as the data learning pattern.

The programmable sequence engine allows you to configure the QuadSPI module according to the serial flash memory connected on board. This flexible structure is compatible with new command or protocol changes from different vendors.

38.6.2.2 Flexible read xAHB buffers

To reduce the latency of the reads for AHB masters, the data read from serial flash memory is buffered in flexible AHB buffers. There are four such flexible buffers. The size of each of these buffers is configurable with the minimum size being 0 bytes and maximum size being the size of the complete buffer instantiated (1024 bytes). The size of buffer 0 ranges from 0 to BUF0IND. The size of buffer 1 ranges from BUF0IND to BUF1IND, buffer2 from BUF1IND to BUF2IND and, buffer 3 ranges from BUF2IND to the size of the complete buffer (1024 bytes).

Each flexible AHB buffer is associated with the following:

- An AHB master: Optionally, buffer3 may be configured as an "all master" buffer by writing 1 to BUF3CR[ALLMST]. When buffer3 is configured in such a way, any access from a master not associated with any other buffer is routed to buffer3.
- A datasize field representing the amount of data to be fetched from the flash memory on every missed access.

The master port number of every incoming request is checked and the data is returned or fetched into the corresponding associated buffer. Every missed access to the buffer causes the controller to clear the buffer and fetch the BUFxCR[ADATSZ] amount of data from the serial flash memory. As such, you need not configure the buffer size to be greater than ADATSZ because the locations greater than ADATSZ are never used. For any AHB access, the sequence pointed to by BFGENCR[SEQID] is used for the initiated flash memory transaction. The data is returned to the master as soon as the requested amount is read from the serial flash memory. The controller; however, continues to prefetch the rest of the data in anticipation of a next consecutive request. See Figure 191 that shows flexible AHB buffers.

BFGENCR[SEQID] points to an index of the LUT. See LUT for details.

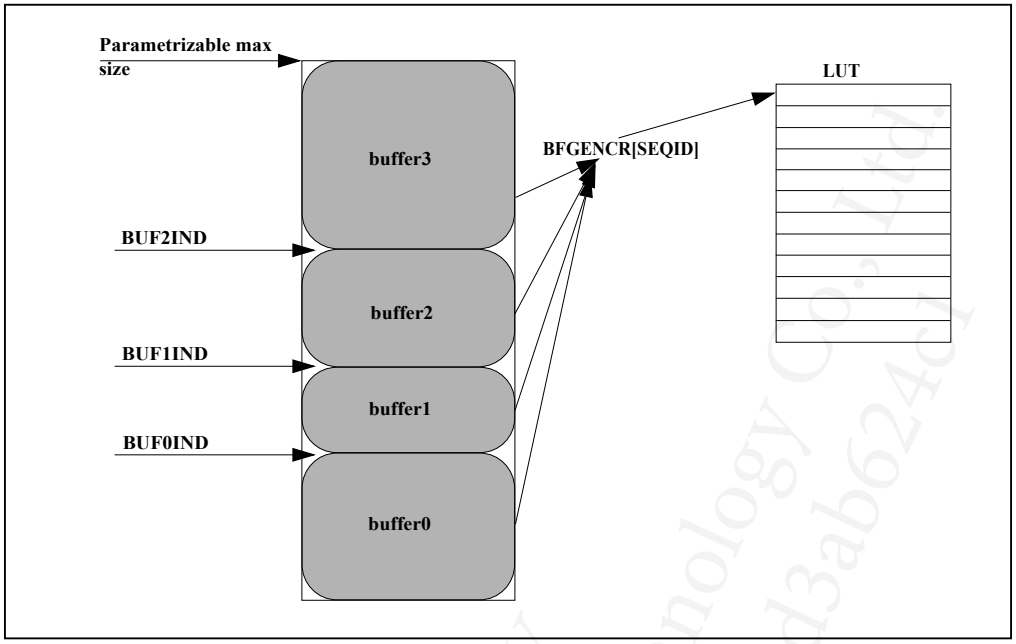


Figure 191. Flexible read AHB buffers

38.6.2.3 Abort mechanism during AHB read

Any ongoing read transaction is aborted if a request from the same master arrives for a location other than the location at which the transaction is going on. The abort can happen at any point of time.

38.6.2.4 HBURST support with AHB read

QuadSPI controller supports HBURST and HSIZE on the AHB read interface. HBURST indicates if the transfer forms part of a burst. Four, eight, and sixteen beat bursts are supported and the bursts might either be incrementing or wrapping. HSIZE indicates the size of the transfer, and supports 8-, 16-, 32-, and 64-bit data sizes. In case of WRAP accesses, QuadSPI generates aligned accesses to serial flash memory if there is no buffer hit for any incoming, non-sequential AHB read access. In case there is a buffer hit, the incoming address in the haddr line is latched as it is. If the total burst size is more than the data prefetch size, an error response is generated and the value of FR[AIBSEF] is configured as 1. The data prefetch size can be defined by BUFxCR[ADATSZ] or data size mentioned in the sequence pointed to by the SEQID field when ADATSZ is programmed as 0. In case of wrap burst, data prefetch size must be greater than or equal to the wrap burst size + 32 bytes. A few examples are shown in the figure below:

Provided by NXP for NXP only
COMPANY CONFIDENTIAL
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4b25-390d3ab624d1

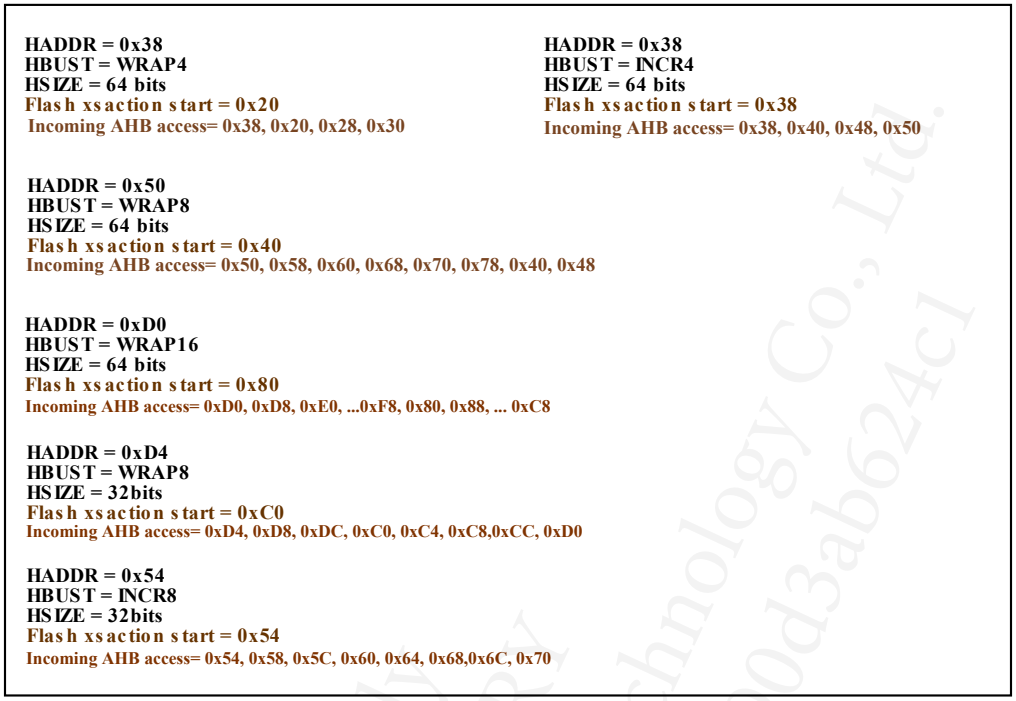


Figure 192. QuadSPI HBURST support

NOTE

The software must take care that the prefetch size should never be set less than the minimum data needed by any external interface to start processing.

38.6.2.5 LUT

The LUT consists of a number of pre-programmed sequences. Each sequence is basically a sequence of instruction-operand pairs, which when executed sequentially, generate a valid serial flash memory transaction. Each sequence can have a maximum of 10 instruction-operand pairs. The LUT can hold a maximum of 16 sequences. The figure below shows the basic structure of the sequence in the LUT.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a2c-b251-390d3ab624c1

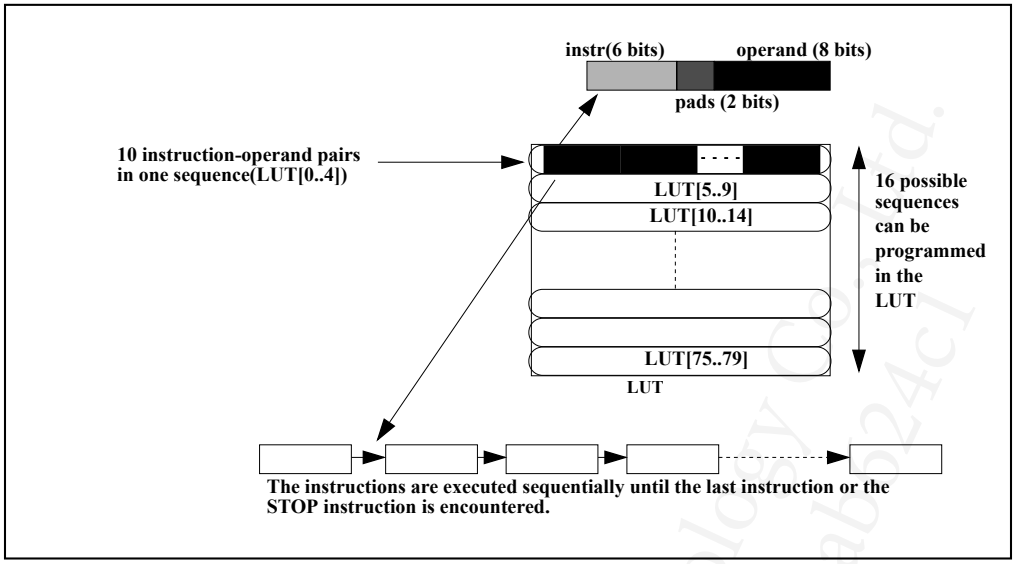


Figure 193. LUT and sequence structure

At reset, the index 0 of the LUT[0..4] is programmed with a basic read sequence as described in Reset sequence. After reset, the complete LUT may be reprogrammed according to the chip connected on board. To protect its contents, during a code runover, the LUT might be locked, after which a write to the LUT will not be successful until it has been unlocked again. The key for locking or unlocking the LUT is 5AF05AF0h, and the associated processes are as follows:

Locking the LUT

1. Write the key 5AF05AF0h into the LUT Key Register (LUTKEY).
2. Write 0b01 to the LUT Lock Configuration Register (LCKCR). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued). A successful write to this register locks the LUT.

Unlocking the LUT

1. Write the key 5AF05AF0h into the LUT Key Register (LUTKEY).
2. Write 0b10 to the LUT Lock Configuration Register (LCKCR). Note that this IPS transaction should immediately follow the above IPS transaction (no other IPS transaction can be issued in between). A successful write to this register unlocks the LUT.

The lock status of the LUT can be read from the LCKCR[UNLOCK] and LCKCR[LOCK] fields.

Some example sequences are defined in Example sequences. The reset sequence at LUT index 0 is shown in the following table.

Table 219. Reset sequence

Instruction	Pad	Operand	Comment
CMD	0h	3h	Read data byte command on one pad
ADDR	0h	18h	24 address bits to be sent on one pad
READ	0h	8h	Read 64 bits
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

38.6.2.6 Issuing SFM commands

Each access to the external device follows this sequence:

1. You must pre-populate the LUT with the serial flash memory command sequences that are required for the flash memory device being used.
2. The module executes the instructions in this sequence one by one. The transaction starts and the module configues the value of SR[BUSY].
3. Communication with the external serial flash memory device starts and the transaction executes.
4. After the transaction is complete (all transmit and receive operations with the external serial flash memory device are complete), the module resets SR[BUSY]. In case of an IP command, FR[TFF] is asserted.

For details, see Flash memory programming and Flash memory read.

You can trigger the processing of SFM commands in the QuadSPI module in one of the following ways:

Using IP commands

For IP commands, the required components need to be written into the following registers and in this sequence:

1. Write the serial flash memory address to be used as provided in the Serial Flash Address Register (SFAR). For IP commands not related to specific addresses, the base address of the related flash memory needs to be programmed. For example, for an instruction which does not require an address (that is, write enable instruction), the SFAR should be programmed with the base address of the memory the command is to be sent to.
2. Write the sequence ID and data size details in the IP Configuration Register (IPCR).
3. Note that writing a value to IPCR[SEQID] must be the last step of the sequence. It is possible to combine all the fields of the IPCR into one single write. See IP Configuration Register (IPCR) for details.

Using AHB commands

Any AHB memory-mapped access is routed to one of the buffers depending on the master port number of the request. If the access is a "miss," a new serial flash memory transaction is initiated. The transaction is based on the sequence pointed to by BFGENCR[SEQID] as described in Flexible read xAHB buffers.

An AHB access is considered memory mapped when the access is to the memory-mapped serial flash memories, as described in Memory Mapped Serial Flash Data - Individual Flash mode on Flash memory A and Memory Mapped Serial Flash Data - Individual Flash Memory mode on Flash memory B.

38.6.2.7 Flash memory programming

In all cases, the memory sector to be written needs to be erased first. The programming sequence is then initiated in the following way:

1. Check that SR[BUSY] is de-asserted or the value of the BUSY field is 0, also, check that the TX buffer is empty. If you need to discard the data present in the TX buffer (SR[TXNE]) then the TX buffer must be cleared by writing 1 to MCR[CLR_TXF].
2. Program the address related to the command in SFAR. If required, write the desired value to SFACR[CAS], otherwise write 0 to it. Also, write 1 to SFACR[WA] if the serial flash memory is a word addressable flash memory, or write 0 in case serial flash memory is byte addressable.
3. Provide initial data for the program command into the circular buffer through the TBDR. At least one words of data must be written into the TX buffer up to a maximum of 256 entries.
4. Program the IPCR to trigger the command. IPCR[SEQID] should point to an index of the LUT that has the flash memory program sequence pre-programmed. You must write 1 to IPCR[IDATSZ] to denote the size of the write.
5. Repeat step 3, depending on the amount of data required, until all of the required data is written to the TBDR. SR[TXFULL] can be used to check if the buffer is ready to receive more data. At any time, TBSR[TRCTR] can be read to check how many words have been written into the TX buffer.

After writing to IPCR[SEQID] (see step 4), the module starts executing the programmed sequence. The software ensures that the correct sequence is programmed into the LUT in accordance with the flash memory connected to the module. The data is fetched from the TX buffer. It consists of 256 entries of 32-bit sizes and is organized as a circular FIFO, the read pointer for which is incremented after each fetch. When all the data is transmitted, the QuadSPI module returns from the busy state to the idle state.

However, this is not true for the external device because the internal programming is still ongoing. You may monitor the relevant status information available from the serial flash memory device and ensure that the programming is done appropriately.

38.6.2.8 Flash memory read

Host access to the data stored in the external serial flash memory device is performed in two steps. First, the data must be read into the internal buffers and in the second step, these internal buffers can be read by the host.

Reading serial flash memory data into the QuadSPImodule internal buffers

A read access to the external serial flash memory device can be triggered in two different ways:

- IP command read: For reading flash memory data into the RX buffer, you must provide the correct sequence ID in IPCR[SEQID]. The sequence ID points to a sequence in the LUT. The software needs to ensure that a correct read sequence is programmed in the LUT in accordance with the serial flash memory device connected on board. You must program the SFAR , SFACR[CAS], and IPCRs. All available read commands supported by the external serial flash memory are possible.

  Optionally, it is possible to clear the RX buffer pointer prior to triggering the IP command by writing a 1 to MCR[CLR_RXF].

  Using these inputs, the complete transaction is built when IPCR[SEQID] is written to. The transaction related to the read access starts and the requested number of bytes is fetched from the external serial flash memory device into the RX buffer. As the read access is triggered by an IP command, the value of both SR[IP_ACC] and SR[BUSY] is set to 1.. A count of the number of entries currently in the Rx buffer can be obtained from RBSR[RDBFL].

  Communication with the external serial flash memory stops if the specified number of bytes are read (on successful completion of the transaction).

  NOTE

  In case of external DQS strobe based sampling, read data size (IDATSZ or LUT size) must be in multiples of 8 bytes.

- AHB command read: For reading flash memory data into the AHB buffer, you must:
  - Set up a read access by a master to the address range in the system memory map, which the external serial flash memory devices are mapped to.
  - Write a desired value to SFACR[CAS], if required, or write 0 to the field.
  - Program the buffer registers corresponding to the AHB master initiating the request, and this depends on the configuration of RBCT[RXBRD].
  - Provide the correct sequence ID in the BFGENCR. The software ensures that a correct read sequence is programmed in the LUT in accordance with the serial flash memory device connected on board. Flash memory device selection and access mode are determined by the address accessed in the AHB address space associated with the QuadSPI module (see Memory-mapped serial flash memory data—individual flash memory mode on flash memory A , Memory-mapped serial flash memory data—individual flash memory mode on flash memory B and Parallel flash memory mode.)

  On each AHB read access to the memory mapped area, the valid data in the AHB buffer is checked against the address requested in the actual read. When the AHB read request cannot be served from the content of the AHB buffer, the buffer is flushed and the controller executes the sequence pointed to by the sequence ID. The requested number of buffer entries defined in BUFxCR[ADATSZ] is then fetched from the external serial flash memory device into the internal AHB buffer. As the read access is triggered through the AHB bus, the value of SR[AHB_ACC] is set, driving SR[BUSY] in turn, until the transaction is complete. Communication with the external serial flash memory stops when the specified number of entries is filled.

Data transfer from the QuadSPI module internal buffers

The data read out from the external serial flash memory device by the QuadSPI module is stored in the internal buffers. The means of accessing the data from the buffer differs depending on which buffer the data is loaded to. See Block diagram for details on the two available buffers, the RX buffer and the AHB buffer, in this module. The buffer appears transparent to you and is non-memory mapped. See the "Flexible AHB Buffer" section for details.

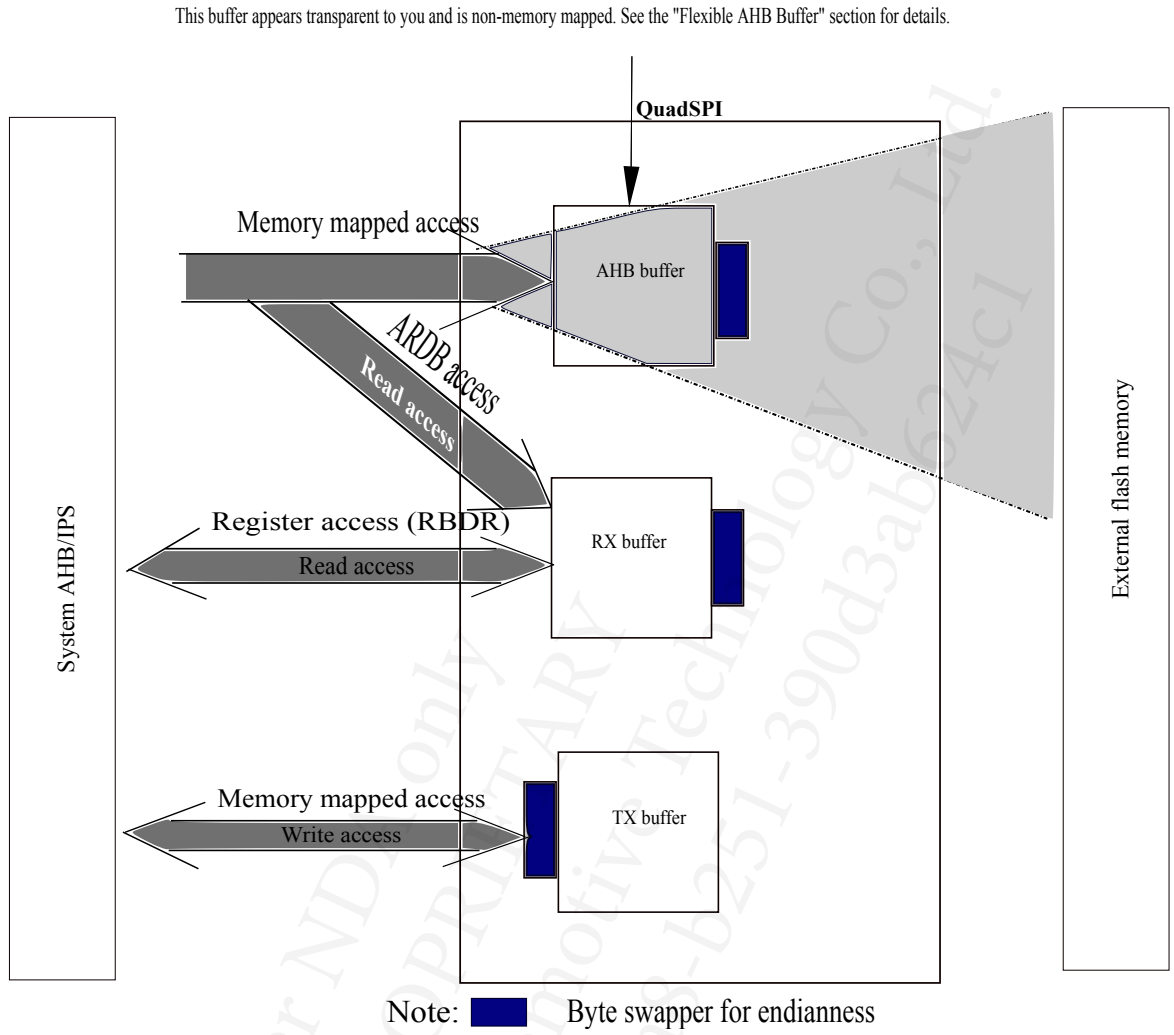


Figure 194. QuadSPI memory map

The RX buffer is implemented as FIFO of depth entries of 4 bytes. Its content is accessible in two different address areas, both referring to identical data and the same physical memory:

- In the IPS address space in the area associated with RX Buffer Data Register (RBDR0 - RBDR63).
- In the AHB address space in the area associated with AHB RX Data Buffer Register (ARDB0 - ARDB127).Two successive entries are accessed with one single 64-bit AHB read operation.

The RX buffer operation can be summarized as follows:

- RBCT[WMRK] determines at which fill level SR[RXWE] is asserted and how many entries are removed from the RX buffer on each buffer POP operation.
- SR[RXWE] indicates that the configured number of data entries is available in the RX buffer and RBSR[RDBFL] indicates how many valid entries are available in total.
- The first entry (RBDR0 or ARDB0) always corresponds to the first valid entry in the RX buffer.

For details, see RX Buffer Data Register (RBDR0 - RBDR63) and AHB RX Data Buffer Register (ARDB0 - ARDB127).

- Flag-based data read of the RX buffer is performed by polling SR[RXWE]. When it is asserted, the valid entries can be read either via the IPS address space (RBDRn) or the AHB address space (ARDBn). A buffer POP operation must be

triggered by the application by writing a 1 to FR[RBDF]. This automatically updates the FIFO to point to the next entry as defined by RBCT[WMRK]. For example, if WMRK is set to 3, then the buffer discards 16 bytes of data.

- DMA-controlled data read of the RX buffer is performed by using the DMA module. The application must ensure that the DMA controller of the related chip is programmed appropriately, as described in DMA usage.
- DMA-controlled read out is triggered fully automatically by the assertion of SR[RXWE]. The related buffer POP operation is also handled completely inside the QuadSPI module. As in the case explained here, accessing the RX buffer content either on RBDRn or ARDBn related addresses is equivalent.
- AHB buffer data read via memory-mapped access: This kind of access is performed by reading one of the addresses assigned to the external serial flash memory device(s) within the range specified in Table 213. If this is not the case, a memory-mapped AHB command read is triggered as described above. If the requested data is already available in the AHB buffer, it is provided directly to the host.

When an AHB access is made to the flash memory mapped address, the data is fetched and returned to the AHB interface. The AHB interface is stalled until the data is fetched. As soon as the data from the requested address is read by the QuadSPI module, the AHB read access is served. Therefore, it is possible to run sequential reads from the AHB buffer at arbitrary speed without the need to monitor any information about the availability of the data. Nevertheless, this access scheme stalls the AHB bus for the time required to read the data from the serial flash memory device. If you know that the access is sequential, a better way is to have a prefetch enabled by programming the value of BUFxCR[ADATSZ] so that the data is fetched into the buffer before the next sequential AHB access.

As long as the host restricts its accesses to the data present in the buffer and to the data currently fetched from the serial flash memory, it is possible to run the host read from the AHB buffer simultaneously with the serial flash memory read into the AHB buffer.

38.6.2.9 AHB write

This module supports a single master AHB write with these burst types — INCR4, INCR8, INCR16, and INCR unspecified 4n bytes and single 4n bytes — and these burst types are supported with all Hsizes. An AHB write transaction size can be expressed in multiples of 4 bytes, and a single transaction must have a minimum of 4 bytes.

The following restrictions apply to the QuadSPI module with respect to AHB write transactions:

- AHB transactions of WRAP* types are not supported. For these unsupported access types, the controller returns an error response and does not initiate any flash memory transaction.
- Back to back, fixed address AHB write is not supported.
- Early burst termination is not supported for AHB transactions.

The QuadSPI controller queues back-to-back AHB writes with continuous address and sends them in a single flash memory transaction. However, in case the AHB address of a burst is not in continuous addressing sequence to previous write burst, a new flash memory transaction is initiated.

Following is the programming sequence for an AHB write:

1. Check that SR[BUSY] is deasserted or is 0 and check that the TX buffer is empty. If there is any residue data present in the TX buffer (SR[TXNE]), the buffer must be cleared by writing '1'1 to MCR[CLR_TXF].
2. Based on the type of flash memory device that is used for page programming, AWRCR[PPW_WR_DIS] and AWRCR[PPW_RD_DIS] need to be programmed.

   The default reset value for both the fields is 0 and allows writing to flash memory device assuming a very low T_{pp} (page program wait time) period. This period is very high for most of the flash memory devices. After any write to a flash memory device, the module waits for the write to finish before proceeding further.

   This is controlled by the AWRCR[PPW_WR_DIS] and AWRCR[PPW_RD_DIS] field configurations.
3. Configure trigger level AWRCR[AWTRGLVL] to an initial buffering level of the TX FIFO. Whenever this trigger level is reached by the AHB write, the QuadSPI module triggers a write to the flash memory.
4. Program the BFGENCR[SEQID] to point to the LUT where flash memory write sequence is stored.

5. AHB write master can now enable the transactions to be written to the flash memory. The AHB master must ensure data throughout to avoid underflow.
6. When a flash memory write completes and flash memory CS is deasserted, the software must poll SR[BUSY] for the deasserted value, which is 0. This indicates that the AHB write is complete. In case an underflow has occurred, flash memory CS is deasserted with SR[TXNE] to know that an AHB write is pending with some residue data in the TX buffer.
7. Whenever flash memory CS is deasserted after an AHB write sequence, FR[PPWF] is set if any of AWRCR[PPW_RD/WR_DIS] is enabled. FR[PPWF] disables any further flash memory read/write based on AWRCR[PPW_RD]/AWRCR[WR_DIS]. After FR[PPWF] is set, it returns every incoming AHB transaction with an error response.
8. The software must clear FR[PPWF] to resume AHB transactions after the expiry of T_{pp} period of flash memory. It can also send IPS read command to the flash memory for checking if T_{pp} is over.
9. The software must ensure that the correct read and write LUT sequences are selected corresponding to the AHB read and write transactions. QuadSPI would malfunction if flash memory read sequence is selected during AHB write or vice versa.

38.6.2.10 Byte ordering of serial flash memory read data

The basic scheme is that the first byte read out of the serial flash memory device, which is addressed by SFAR[SFADR], corresponds to RBDR0[31:24] for IP command read. In contrast to that for AHB command read, the bytes are always positioned according to the byte ordering of the AHB bus.

- Byte ordering in individual flash memory mode

  The following table provides the byte ordering scheme of how the byte oriented data space of the serial flash memory device is mapped into one single 32-bit entry of the RX buffer or the AHB buffer. The table is valid within the following context:

  — Flash memory A or Flash memory B in individual flash memory mode

  — All AHB data read commands with 32-bit access size

Table 220. Byte ordering in individual flash memory mode

Serial flash memory byte numbering	3	2	1	0
Buffer entry bit position [31:0] (32-bit data width)	[31:24]	[23:16]	[15:8]	[7:0]

NOTE

For IP commands, the read size can be specified as number of bytes. If this number is not a multiple of four, then the last buffer entry is not completely filled with the missing higher numbered bytes at undefined values.

  For AHB commands and reads, starting from an address not aligned to 32-bit boundaries, the requested bytes are given at the appropriate positions according to the AMBA AHB specification.

- Byte ordering in parallel flash memory mode

  In the parallel flash memory mode, each byte is combined out of two- half bytes that are read or written in parallel from the two serial flash memory devices. The following tables show how the flash memory content is separated into the half-bytes and how the half-bytes are included with the RBDR0 register.

Table 221. Serial flash memory device half-byte ordering

Serial flash memory device byte #	Flash memory A bit position		Flash memory B bit position	
	[7:4]	[3:0]	[7:4]	[3:0]
0	fah0	fal0	fbh0	fbl0
1	fah1	fal1	fbh1	fbl1
2	fah2	fal2	fbh2	fbl2
3	fah3	fal3	fbh3	fbl3
4	fah4	fal4	fbh4	fbl4
5	fah5	fal5	fbh5	fbl5
6	fah6	fal6	fbh6	fbl6
7	fah7	fal7	fbh7	fbl7
8	fah8	fal8	fbh8	fbl8

The table entry naming reflects the half-byte positioning in the serial flash memory devices:

- <fa> h0 means Flash memory A, <fb>h0 means Flash memory B.
- fa<h>0 means half byte in high position, fa<l>o means half-byte in low position.
- fah<0> means physical byte address 0 in the serial flash memory device and fal<1> means physical byte address 1 in the serial flash memory device.

Table 222. Byte ordering in parallel flash memory mode - RX buffer

SFAR[SFADR] set to 0h								
RBDR0 ARDB0	fal1	fbl1	fah1	fbh1	fal0	fbl0	fah0	fbh0
RBDR1 ARDB1	fal3	fbl3	fah3	fbh3	fal2	fbl2	fah2	fbh2
SFAR[SFADR] set to 1h								
RBDR0 ARDB0	fal2	fbl2	fah2	fbh2	fal1	fbl1	fah1	fbh1
RBDR1 ARDB1	fal4	fbl4	fah4	fbh4	fal3	fbl3	fah3	fbh3

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHA Automotive Technology Co., Ltd.
c8c0265-4821-48a8-b251-390a3ab624c1

NOTE

For IP commands, the read size can be specified as number of bytes. If this number is not a multiple of four, the last buffer entry is not completely filled with the missing higher numbered bytes at undefined values.

Table 223. Byte ordering in parallel flash memory mode - TX buffer*

SFAR[SFADR] set to 0h								
TBDR0	fal1	fbl1	fah1	fbh1	fal0	fbl0	fah0	fbh0
TBDR1	fal3	fbl3	fah3	fbh3	fal2	fbl2	fah2	fbh2

* Applicable only for single I/O mode

Table 224. Byte ordering in parallel flash memory mode - AHB buffer

AHB address (32-bit access)	fal1	fbl1	fah1	fbh1	fal0	fbl0	fah0	fbh0
AHB address 800_0004h (32-bit access)	fal3	fbl3	fah3	fbh3	fal2	fbl2	fah2	fbh2

NOTE

For an AHB command read starting from an address not aligned to 32-bit boundaries or an AHB access size smaller than 32-bit, the requested bytes are given at the appropriate positions according to the AMBA AHB specification.

- Buffer entry ordering for 64-bit read access

  For read access via the AHB interface, a 64-bit access is possible. Each 64-bit access reads two 32-bit entries, simultaneously. The ordering of these 32-bit entries within the 64-bit word is provided in the following table.

Table 225. 64-bit read access buffer entry ordering

AHB read data bit position [63:0]	**[63:32]**	**[31:0]**
Buffer entry #	**Odd (1, 3, 5, ...)**	**Even (0, 2, 4, ...)**

38.6.2.11 Normal mode interrupt and DMA requests

The QuadSPI module has different flags that can only generate interrupt requests and one flag that can generate an interrupt as well as DMA requests. The following table lists the eight conditions. Note that the flags mentioned in the table are associated with the Flag Register (FR).

Table 226. Interrupt and DMA request conditions

Condition	Flag (FR)	DMA
Data learn pattern failure	DLPFF	-
TX buffer fill	TBFF	-
TX buffer underrun	TBUF	-
Illegal instruction error	ILLINE	-

Table continues on the next page...

Table 226. Interrupt and DMA request conditions (continued)

Condition	Flag (FR)	DMA
RX buffer drain	RBDF	X
RX buffer overflow	RBOF	-
AHB buffer overflow	ABOF	-
AHB sequence error	ABSEF	-
AHB illegal transaction error	AITEF	-
IP command usage error	IUEF	-
IP command trigger during AHB access error	IPAEF	-
IP command trigger could not be executed error	IPIEF	-
IP command related transaction finished	TFF	-

Each condition has a corresponsing field in the Flag Register (FR) and a request enable field in the DMA Request Select and Enable Register (RSER). The FR[RBDF] has separate enable fields for generating IRQ and DMA requests. Note that not all the fields have an individual IRQ line. See the chip's Interrupt Vector table for details.

- Transmit buffer fill interrupt request

  This indicates that the TX buffer can accept new data. The buffer is asserted if FR[TBFF] is asserted and if the value of the corresponding enable field, RSER[TBFIE], is 1. See TX buffer Operation for details on the assertion of FR[TBFF].

  Apart from IRQ, it is possible to handle the TX buffer fill by using the DMA. If the value of RSER[TBFDE] is 1, a DMA request is triggered when the number of available space in the TX buffer is more than the TBCT[WMRK] valid entries and the value of SR[TXWA] is set. The application must configure the environment appropriately (for example, the DMA controller) for the DMA transfer.

- Receive buffer drain interrupt or DMA request

  This is derived from FR[RBDF], indicating that the RX buffer of the QuadSPI module has data available from the serial flash memory device to be read by the host. It remains set as long as RBSR[RXWE] is configured. Also, RSER[RBDIE] enables the related IRQ.

  Apart from the IRQ, it is possible to handle the RX buffer drain by using the DMA. If the value of RSER[RBDDE] is 1, a DMA request is triggered when the RX buffer contains more than RBCT[WMRK] valid entries. The application must set the environment appropriately (for example, the DMA controller) for the DMA transfers.

- Buffer overflow/underrun interrupt request

  This is a combination of the following fields (all located in the Flag Register (FR) with the related enable bits in the DMA Request Select and Enable Register (RSER)):

  — TBUF - TX buffer underrun, enabled by TBUIE

  — RBOF - RX buffer overflow, enabled by RBOIE

  — ABOF - AHB buffer overflow, enabled by ABOIE

  — The transmit buffer underrun indicates that an underrun condition in the TX buffer has occurred. It is generated when a write instruction is triggered whilst the TX buffer is empty and the the value of RSER[TBUIE] is 1.

  — The receive buffer overflow indicates that an overflow condition in the RX buffer has occurred. It is generated when the RX buffer is full, an additional read transfer attempts to write into the RX buffer, and the value of RSER[RBOIE] is 1.

  — The AHB buffer overflow indicates that an overflow condition in the AHB buffer has occurred. It is generated when the AHB buffer is full, an additional read transfer attempts to write into the AHB buffer and the value of RSER[ABOIE] is 1.

— The data from the transfers that generated the individual overflow conditions is ignored.

- Serial flash memory command error interrupt request

  If the IPAEF, IPIEF, or IUEF fields in the FR are set, and the related interrupt enable bits in the RSER are also set, then an interrupt is requested.

- Transaction finished interrupt request

  The IP command transaction finished IRQ indicates the completion of the current IP command. It is triggered by FR[TFF] and is masked by RSER[TFIE].

38.6.2.12 TX buffer operation

The TX buffer provides the data used for page programming. For proper operation, it is required to provide at least one entry in the TX buffer prior to starting the execution of the page programming command. The application must ensure that the required number of data bytes is written into the TX buffer fast enough as long as the command is executed without a TX buffer overflow or underrun.

The QuadSPI module sets the FR[TBFF] field as long as the TX buffer is not full and can accept more data. At the end of write through TX buffer, you must clear FR[TBFF] to avoid unnecessary last TX buffer fill interrupt. However, there would always be a pending request asserted from QuadSPI controller at the end of any DMA transfer. When external DMA finishes transfer iteration, this request from QuadSPI is kept asserted for the next iteration loop.

NOTE

Even if the generation of DMA requests for filling the TX buffer is disabled by using RSER[TBFDE], the TX buffer still accepts a DMA transfer because of the last asserted pending request.

Disabling of DMA transfer should be controlled by an external DMA master.

When the QuadSPI module tries to pull data out of an empty TX buffer, FR[TBUB] signals the TX buffer underrun. The TX buffer underrun flag is also asserted when the TX buffer contains less than 32-bits of data and the QuadSPI module tries to pull out data from it. The current IP command leading to the underrun condition is continued until the specified number of bytes is sent to the serial flash memory device. Also, the data that is transferred is in the Fs format. This means, after the underrun flag is set under this condition, it returns Fs until the required number of bytes are not sent. This has been done to ensure that the software does not erase the whole sector after underrun and just reprogramming from failure point serves the purpose. When this sequence command is complete, FR[TBFF] is asserted, indicating that the TX buffer is ready to be written again.

The TX buffer overflow is not signaled explicitly, but TBSR[TRBFL] can monitor the TX buffer fill level.

For more information, see TX Buffer Status Register (TBSR) and Flag Register (FR).

38.6.2.13 Address scheme

Earlier, serial flash memories supported only a 24-bit address space, restricting the maximum memory size of the serial flash memory to 16 MB. The new memory specification supports two types of 32-bit addressing mode in addition to the legacy 24-bit address mode. It also supports segregation of address programmed into Row address and Column address of the flash memory, as per the requirement.

Extended address mode

In this mode, the legacy 24-bit commands are converted to accept 32-bit address commands. The flash memory needs to be configured for the 32-bit address mode. Also, while programming the LUT sequence in QuadSPI for 32-bit mode, the ADDR and ADDR_DDR commands should be programmed with 32d as the operand value, with SFACR[CAS] programmed to 0. If a flash memory needs some bits of the address as its column address, then it must be considered that a total of 12 bits are required by the flash memory; however, the number of bits should not exceed 32, because the maximum address supported by QuadSPI is 32 bits. Each of the memory vendors have a different way of enabling this mode (see the memory specification from memory vendors). For example, the command B7h sent to the Macronix flash memory enables it for the 32-bit address mode.

Extended address register

In this mode, the upper 8 bits of the 32-bit register are provided by the Extended address register in the memory, which provides a specific register that is updated according to the address to be accessed. This effectively converts the legacy 24-bit address

command into 32-bit address commands. The memories greater in size than 16 MB consist of banks of 16 MB each. The 8 bits occupied in the extended address register effectively enable a bank. For example, in Spansion memory, when the extended address register is updated with a value of 1h, with the help of the 17h command, it opens Bank1 of the memory. The consequent 24-bit address commands lead to Bank1. The extended address register needs to be updated with the respective value for access to other banks. This effectively converts the legacy 24-bit address command into 32-bit address commands.

Separation of address into rows and columns

This mode has been introduced for flash memories that need addresses segregated into rows and columns. The value in SFACR[CAS] defines the width of the column address required by a flash memory. The actual address to be provided is derived from the incoming address in case of AHB-initiated transactions and the value of SFAR in case of IPS-initiated transactions, if the value of SFACR[CAS] is 0. Otherwise, the actual address takes CAS into consideration. If the value of SFACR[CAS] is 3, then bits 26-3 of the programmed address are sent to the flash memory as its page address. This is in case the flash memory is operating in a 24-bit mode and bits 2-0 are sent as its column address. If a flash memory requirement for column address is less than the number of pads in which address has to be sent, then QuadSPI appends the remaining bits by 0. You must program the operand value in CADDR and CADDR_DDR command accordingly. It must be ensured that the total number of address bits requested by flash memory as its page and column address must not be more than 32 bits.

Word addressable mode for flash memory

This mode has been introduced for flash memories, which have a word-addressable memory. This means, each address of the flash memory contains one word (two bytes) of data. The value of SFACR [WA] is configured to 1 to enter this mode. QuadSPI internally divides the incoming address in the AHB bus or the address in the SFAR to map it to a valid flash memory location. For example, if the incoming address is 2004h, the controller re-maps this address to access the flash memory location 1002h. If not in this mode, the incoming address 2004h is mapped to flash memory location 2004h.

38.6.2.14 OTFAD support

If OTFAD is enabled in the system, any reads from the QuadSPI are first decrypted by the OTFAD engine and then returned to the AHB bus. If OTFAD is enabled, the prefetch never crosses the 1 KB boundary.

38.6.3 Module Disable mode

Module Disable mode is a block-specific mode that the QuadSPI can enter to disable serial flash memory clock and AHB command. This mode can be entered by:

- The host software: by writing a '1' to MCR[MDIS]

Below are the condition that must be fulfilled to enter the Module Disable mode:

- SR[BUSY] = 0
- SR[AHBTRN] = 0
- RBSR[RDBFL] = 0
- SR[RXDMA] = 0
- SR[TXDMA] = 0
- None of the flags in FR are enabled as interrupts is set

The conditions mentioned above ensures the following:

- There is no SFM command currently being executed.
- All the data read into the RX buffer from the serial flash memory have been fetched by the application.
- There is no current AHB access.
- There is no active DMA request.
- There is no enabled interrupt that is pending.

Certain read or write operations have a different effect when the QuadSPI is in the Module Disable mode. In the Module Disable mode, not all of the status and flag bits of the QuadSPI module are updated, and writing to them has no effect. Interrupt and DMA request signals cannot be cleared while in the Module Disable mode.

NOTE

It is illegal to issue a new SFM command starting two clock cycles prior to raising the request of entering the Module Disable mode until the QuadSPI stays in this mode.

38.6.4 Leaving Module Disable mode

In the Module Disable mode, the serial flash memory clock and AHB command to the QuadSPI module are switched off.

After the QuadSPI has left this mode and has returned to Normal mode, the execution of the first SFM command is deferred until the clock to drive that part of the module related to the serial flash memory device is available. Depending upon the point in time when the first SFM command is triggered, the actual execution of the command starts with a delay, respective with the re-enabling of the flash memory clock signal.

38.7 Initialization/application information

This section provides the initialization and application information of the QuadSPI module.

38.7.1 Power up and reset

The serial flash memory devices connected to the QuadSPI module might require special voltage characteristics of their inputs during power up or reset. The application must ensure this.

38.7.2 Available status/flag information

This section provides an overview of the different flags and statuses available, and their interdependencies for different use cases. The SR and FR are the related registers.

38.7.2.1 IP commands

See IP Configuration Register (IPCR) for additional details not explicitly covered in this paragraph.

- IP commands—normal operation

  Writing to IPCR[SEQID] triggers the execution of a new IP command. Given that this is a legal command, SR[IPACC] and SR[BUSY] are asserted simultaneously, immediately after the execution starts.

  After the instruction on the serial flash memory device is complete, these field deassert and FR[TFF] is configured.

- IP commands—error situations

  See Overview_of_Error_Flags for details.

38.7.2.2 AHB commands

See the "Reading serial flash memory data into the QuadSPI module internal buffers" topic in theFlash Memory Read section for details.

- AHB commands—normal operation

  Memory-mapped read access to a serial flash memory address not contained in the AHB buffer triggers the execution of an AHB command. Given that this is a legal command, SR[AHB_ACC] and SR[BUSY] are asserted simultaneously, immediately after the execution starts. After the instruction on the serial flash memory device is complete, these fields are deasserted.

- IP commands—error situations

  See Overview of FR error flags for details.

38.7.2.3 SFM commands

An SFM command consists of an instruction code and all other parameters (for example, size or mode bytes) needed for that specific instruction code. Triggering a command either initiates a transaction on the external serial flash memory or results in an error. See Table 227 for details on errors.

38.7.2.4 Overview of error flags

The following table provides an overview of the different error flags in the FR and additional error-related details.

Table 227. Overview of FR error flags

Error category	Error flag in FR	Command execution on serial flash memory device TFF behavior (in case of IP commands only)	Description
AHB error flag	ABOF	Flash memory transaction continues until it finishes	Set when the module tries to push data into the AHB buffer that exceeds the size of the AHB buffer. Occurs only because of the wrong programming of BUFxCR[ADATSZ].
AHB error flag	AITEF	Flash memory transaction is aborted	No response is generated from QuadSPI to AHB bus in case of illegal transaction. Also, the watchdog timer expires.
Miscellaneous error flag	DLPFF	Flash memory transaction continues until it finishes	Set when the DATA_LEARN instruction is encountered in a sequence but no sampling point is found for the data learning pattern.
Miscellaneous error flag	ILLINE	Flash memory transaction aborted	Illegal instruction error flag - set when an illegal instruction is encountered by the controller in any of the sequences.
Command arbitration error	IPIEF	TFF not asserted in conjunction with that command	IP command error - caused when IP access is currently in progress (IP_ACC is set) and during: • Write attempt to IPCR register • Write attempt to SFAR register • Write attempt to RBCT register
Command arbitration error	IPAEF	TFF not asserted in conjunction with that command	• AHB command already running, another IP command could not be executed • AHB command already running, write attempt to IPCR[SEQID]
IP command error	IUEF	–	• IP command usage error
Buffer-related error	RBOF	TFF is asserted on completion	• RX buffer overrun

Table continues on the next page...

Table 227. Overview of FR error flags (continued)

Error category	Error flag in FR	Command execution on serial flash memory device TFF behavior (in case of IP commands only)	Description
Buffer-related error	TBUF	TFF is asserted on completion	• TX buffer underrun

Note that only the buffer-related errors are associated with a transaction on the external serial flash memory. All the other errors do not trigger an actual transaction.

38.7.2.5 IP bus and AHB access command collisions

There are two flags related to this topic, FR[IPAEF] and FR[IPIEF]. See the "Reading serial flash memory data into the QuadSPI module internal buffers" topic of the Flash Memory read section for a description of the flags.

38.7.3 Flash memory device selection

Regardless of the SFM command (IP or AHB), the access mode is selected by specifying the 32-bit address value for the following SFM command.

For IP commands, the access mode is selected with the address programmed into the SFAR register. See Serial Flash Address Register (SFAR) for details.

For AHB commands, the access mode is determined by the memory-mapped address. See AMBA Bus Register Memory Map for details.

38.7.4 DMA usage

For a complete description of the DMA module, see the related DMA Controller chapter. This section only provides QuadSPI-specific DMA usage details.

38.7.4.1 DMA usage in normal mode

38.7.4.1.1 Bandwidth considerations

Careful consideration of the throughput rate of the entire chain (serial flash memory -> AHB bus / IP bus -> DMA controller) involved in the read/write data process is essential for a proper operation. Such analysis must take into account not only the data rate provided by the serial flash memory but also the data rate of the AHB bus and the performance of the DMA controller in reading/writing data from/to the RX/TX buffer.

Two figures must match for a proper operation, which means that the data rate provided by the serial flash memory device must not exceed the average RX buffer readout data rate. Otherwise, the longer this state persists, it results into an RX buffer overflow. Similarly, the data consumed by the serial flash memory device must not exceed the average TX buffer fill rate. If this persists, it leads to an underrun.

AHB bus side (data read)

The total number of bus cycles for each DMA minor loop completion is added from the following components:

The following table provides certain examples for typical use cases:

Case 1: DMA needs to read 4 bytes from SRAM and provide to QuadSPI. It costs total four bus clock cycles. Then, DMA handshake adds additional six bus clock cycles, resulting in a total of [6 + 4 * (32/4) = 38] bus clock cycles.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CUBIC Automotive Technology Co., Ltd.
c8c6e5-4e23-48a8-b251-390c3ab624c1

Table 228. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 60 MHz bus clock frequency
3	16	6+(16/4)*4 = 22	~366ns
7	32	6+(32/4)*4 = 38	~633ns
11	48	6+(48/4)*4 = 54	~900ns
15	64	6+(64/4)*4 = 70	~1166ns

Case 2: DMA needs to read 32 bytes from SRAM and provide to QuadSPI. DMA handshake takes an additional six bus cycles, with 32 bytes DMA read from SRAM costs (8 + 3) core clock cycles. DMA writes 32 bytes to QuadSPI, takes 2 * (32/4) = 16 bus cycles with one additional CPU access to QuadSPI, costing two bus clock cycles. This results in a total 6 + (8+3)/2 + 2 * (32/4) +2 = 30 bus clock cycles.

Table 229. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop >	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 80 MHz bus clock frequency
3	16	6 + (4+3) /2 + (16/4)*2 + 2 = 20	~333ns
7	32	6 + (8+3) /2 + (32/4)*2 + 2 = 30	~500ns
11	48	6 + (4+3)/2*3 + (48/4)*2 + 2 = 44	~733ns
15	64	6 + (8+3) + (64/4)*2 + 2 = 51	~810ns

NOTE

This table figure represents an ideal scenario; actual performance depends on how the chip integrates DMA and QuadSPI modules.

Serial flash memory device side (data read)

The number of serial flash memory cycles can be determined in the following way:

- Number of serial flash memory clock cycles is required to read 4 bytes, corresponding to one RX buffer entry (setup of command and address not considered): two cycles for quad DDR mode instructions in the parallel flash memory mode, two cycles for the Octal DDR mode (Hyperflash) in the individual flash memory mode, four cycles for quad (SDR) mode instruction in the parallel flash memory mode, or dual IO DDR mode instruction in the parallel flash memory mode, eight cycles for quad mode (SDR) instructions in individual flash memory mode, and so on.
- Overhead because of clock domain crossing: one cycle

The following table lists the number of clock cycles required to read the data from the serial flash memory corresponding to the different settings of RBCT[WMRK]:

Table 230. Access duration examples for serial flash memory side

RBCT[WMRK] setting	Num bytes per DMA loop [1]	Num SCKFx for 80 MHz SCKFx			Time duration of flash memory data readout for 80 MHz SCKFx (~12.5ns period)		
		IFM [2] quad	IFM quad DDR	PFM [3] quad DDR/ IFM octal DDR	IFM quad	IFM quad DDR	PFM quad DDR
0	4	8	4	2	~100ns	~50ns	~25ns
1	8	16	8	4	~200ns	~100ns	~50ns
3	16	32	16	8	~400ns	~200ns	~100ns
7	32	64	32	16	~800ns	~400ns	~200ns

1. DMA loop refers to one minor loop completion that is equivalent to one major loop iteration.
2. Individual flash memory mode
3. Parallel flash memory mode

NOTE

The table figure represents an ideal scenario; actual performance depends on how the chip integrates with DMA and QuadSPI modules.

From the examples given in the two tables above, it can be seen that depending on the relationship between the bus clock and serial flash memory clock frequencies, you could perform certain settings where the serial flash memory provides the read data faster than the AHB bus can read out the RX buffer. In the tables provided above, it is the case of PFM quad DDR mode with watermark up to three and other cases. In these cases, the RX buffer data keeps accumulating over time and eventually overflows. To avoid RX buffer overflow, the data transaction size should be small enough.

A complementary example is when the watermark is set to be too high. In such a case, the time taken by the DMA to read out the RX buffer entries should be lesser than the time taken by the controller to push in the remaining entries in the buffer.

IPS bus side (data write)

The total number of bus cycles for each DMA minor loop completion are added from the following components:

- Overhead for each minor loop, given by DMA controller: assume 10 cycles
- Overhead because to clock domain crossing: assume two cycles
- Number of bus clock cycles required for 16 bytes (128-bit write size): assume four cycles (read/write sequence of DMA controller)

Note that the size of the minor loop is determined by the size of TBCT[WMRK]; therefore, the overhead specified above distributes among (TBCT[WMRK]+1) write accesses of 32-bit each.

The following table provides some examples for typical use cases:

Table 231. Access duration examples for bus clock side

TBCT[WMRK]	Number of bytes per DMA loop [1]	Number of bus clock cycles for DMA minor loop	Time duration of DMA minor loop for 80 MHz bus clock frequency
3	16	12+4 = 16	~200ns
7	32	12+8 = 20	~250ns

1. DMA loop refers to one minor loop completion that is equivalent to one.

NOTE

The table figure represents an ideal scenario; actual performance depends on how the chip integrates with DMA and QuadSPI modules.

Serial flash memory device side (data write)

The number of serial flash memory cycles can be determined in the following way:

- Number of serial flash memory clock cycles required to write 16 bytes, corresponding to four TX buffer entry (setup of command and address not considered): Eight cycles for Octal DDR mode (Hyperflash memory) instructions in individual flash memory mode, 32 cycles for quad SDR writes in individual flash memory mode and 64 cycles in single IO SDR writes in parallel flash memory mode.
- Overhead due to clock domain crossing: one cycle

The following table lists the number of clock cycles required to read the data from the serial flash memory corresponding to the different settings of TBCT[WMRK]:

Table 232. Access duration examples for serial flash memory side

TBCT[WMRK] setting	Num bytes per DMA loop [1]	Num SCKFx			Time duration for consuming data at flash memory interface 100 MHz SCKFx (10ns period)[2]			Time for fifo to get empty[3]		
		IFM [4] quad	IFM octal DDR	[5] single IO SDR mode	IFM quad	IFM octal DDR	PFM single IO SDR	IFM quad	IFM octal DDR	PFM Single IO SDR
3	16	32	8	64	320ns	80ns	640ns	2240ns	560ns	4480ns
7	32	64	16	128	640ns	160ns	1280ns	1920ns	480ns	3840ns

1. DMA loop refers to one minor loop completion that is equivalent to one major loop iteration.
2. Not all flash memory devices support writes at 100 MHz. See the flash memory data sheet for the actual page program frequency supported.
3. The assumption for these timings is that the TX Fifo is full when the transaction is initiated.
4. Individual flash memory mode
5. Parallel flash memory mode

NOTE

The tables mentioned above are only examples which must be correlated with the DMA in the system.

Considering the examples provided in the two tables above for TX FIFO, it is evident that depending on the relationship between the bus clock and serial flash memory clock frequencies, there are settings possible where the serial flash memory consumes data faster than the IPS bus can write data in TX buffer. In these cases, a TX buffer underrun situation occurs. To avoid TX buffer underrun, the data transaction size should be large enough.

38.7.5 Flash memory devices address mapping

You can configure QuadSPI in following modes for the supported flash memory ports A and B:

- Single mode
- Serial mode
- Parallel mode

The sizes of the flash memory devices are mapped with the system memory space based on the configurations of the following registers:

- SFA1AD
- SFA2AD

- SFB1AD
- SFB2AD

The total memory region for the flash memory devices is mapped between QuadSPI_AMBA_BASE and TOP_ADDR_MEMB2 such that the corresponding CS is asserted based on SFA1AD, SFA2AD and SFB1AD register configurations.

38.7.5.1 Single mode

For single mode configuration, you must write the same value to SFB1AD and SFB2AD registers that you write to the SFA2AD register.

For dual-die flash memories, the values you write to SFB1AD and SFB2AD registers corresponds to the mapped top addresses of each die.

For single-die flash memories, you must write the same value to SFA2AD register that you write to the SFA1AD register.

Following is a programming example for single mode single-die flash memory:

- QuadSPI_AMBA_BASE - 1000_0000h
- SFA1AD[TPADA1] - 2000_0000h
- SFA2AD[TPADA2] - 2000_0000h
- SFB1AD[TPADB1] - 2000_0000h
- SFB2AD[TPADB2] - 2000_0000h

The following figure illustrates the memory mapping for single mode QuadSPI configuration.

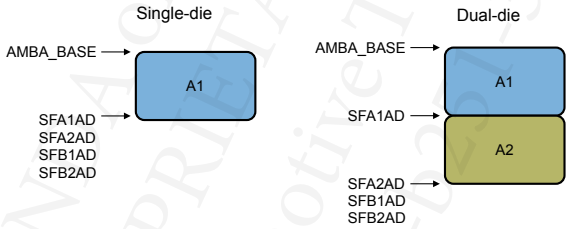


Figure 195. Memory map for Single mode

38.7.5.2 Serial mode

For serial mode configuration, you must write the same value to SFB1AD and SFB2AD registers that you write to the SFA2AD register.

For dual-die flash memories, the values you write to SFA1AD and SFA2AD registers corresponds to the mapped top addresses of each die.

For single-die flash memories, you must write the same value to SFA2AD register that you write to the SFA1AD register.

Following is a programming example for single mode single-die flash memory:

- QuadSPI_AMBA_BASE - 1000_0000h
- SFA1AD[TPADA1] - 2000_0000h
- SFA2AD[TPADA2] - 3000_0000h
- SFB1AD[TPADB1] - 5000_0000h
- SFB2AD[TPADB2] - 7000_0000h

The following figure illustrates the memory mapping for serial mode QuadSPI configuration.

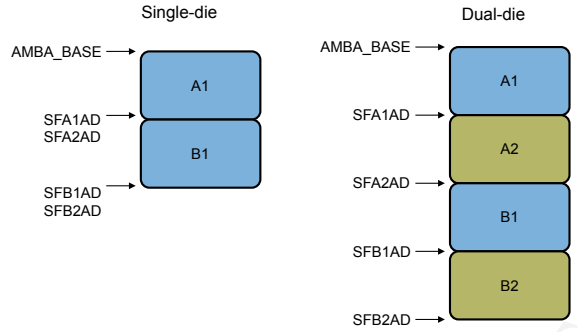


Figure 196. Memory map for Serial mode

38.7.5.3 Parallel mode

QuadSPI can access two flash memories in parallel. This increases the throughput of the QuadSPI by two times. Only read operations and x1 mode write are allowed in this mode. In case a write transaction is initiated in with more than one pad in the Parallel mode, the value of FR[IUEF] is set. When dual-die flash memories are accessed in the Parallel mode, it is mandatory for flash memory A1 to be the same size as B1 and A2 to be the same size as B2. The following figure shows how QuadSPI maps the incoming addresses to the different flash memories connected on board.

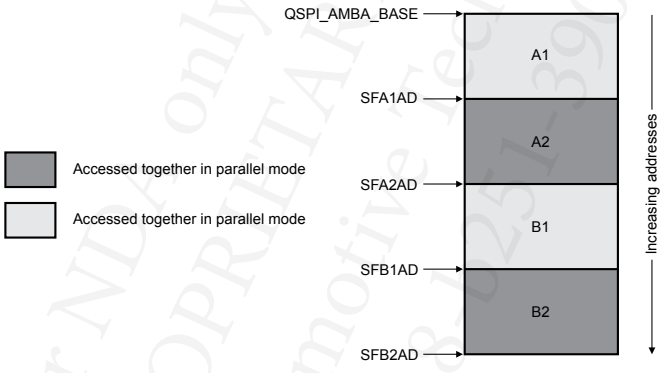


Figure 197. Flash memory addressing

Following is a programming example for parallel mode access (flash memory sizes are assumed to be 256 MB):

- QuadSPI_AMBA_BASE - 1h
- SFA1AD[TPADA1] - 2h
- SFA2AD[TPADA2] - 3h
- SFB1AD[TPADB1] - 4h
- SFB2AD[TPADB2] - 5h

To access the first location of the A1/B1 pair, the incoming address should be 1h. QuadSPI_AMBA_BASE is subtracted from this address and the result is divided by two. Therefore, addresses are provided to flash memories A1 and B1.

Flash memory address = memory mapped address - QuadSPI_AMBA_BASE/2

For memory-mapped address:

- 1000_0000h, flash memory address: 0h (or, the first address of flash memories A1 and B1)
- 1000_0004h, flash memory address: 2h
- 1000_0008h, flash memory address: 4h, and so on.

Similarly, to access the first location of the A2/B2 pair, the incoming address should be 3h.

Flash memory address = memory mapped address - SFA2AD/2

For memory-mapped address:

- 0h, flash memory address: 0h (or, the first address of flash memories A2 and B2)
- 4h, flash memory address: 2h
- 8h, flash memory address: 4h, and so on.

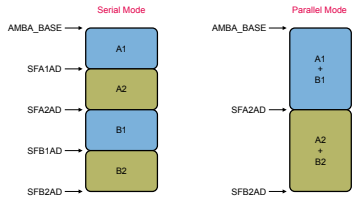


Figure 198. Memory map for Serial and Parallel modes

The software must ensure that when multiple flash memories are used, accesses should never cross a boundary between separate flash memory devices. For example, in the figure provided above, for Serial mode, accesses must not cross the A1 to A2, A2 to B1, or B1 to B2 boundaries. For Parallel mode, accesses must not cross the A1+B1 to A2+B2 boundaries. The software can manage this using the following methods:

- Reduce the data fetch amount of the AHB buffers to 64 bit so that no prefetch occurs. This prevents any legal access from crossing the boundary.
- Reserve a memory region, if prefetch is enabled, to the maximum size of the prefetch, prior to the boundary. As long as no accesses are made to this region of the memory, no access crosses the boundary.
- Ensure that cache does not prefetch across the boundary.

38.8 Byte ordering – endianness

The following topics show the byte ordering in 64-bit LE configuration for AHB buffer and 32-bit LE for TX/RX buffer.

38.8.1 Programming flash memory data

CPU writes instructions to the TBDR register, such as:

- Write TBDR: 4_03_02_01h
- Write TBDR: 8_07_06_05h

The following table shows the content against each TX buffer entry.

Table 233. Example of QuadSPI TX buffer

TX buffer entry	Content
0	4_03_02_01h
1	8_07_06_05h

Programming the TX buffer into the external serial flash memory device results in the following byte order to be sent to the serial flash memory:

- 01...02...03...04...05...06...07...08

38.8.2 Reading flash memory data into the RX buffer

Reading the content from the same address provides the following sequence of bytes, identical to the write case:

- 01...02...03...04...05...06...07...08

The following table shows the content against each TX buffer entry.

Table 234. Resulting RX buffer content

RX buffer entry	Content
0	4_03_02_01h
1	8_07_06_05h

38.8.2.1 Readout of the RX buffer through RBDRn

The RX buffer content appears at CPU read access through the peripheral bus interface in the following order:

- Read RBDR0: 4_03_02_01h
- Read RBDR1: 8_07_06_05h

38.8.2.2 Readout of the RX buffer through ARDBn

The RX buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- 32-bit access: Read ARDB0: 4_03_02_01h
- 32-bit access: Read ARDB1: 8_07_06_05h
- 64-bit access: Read ARDB0: 8_07_06_05_04_03_02_01h

38.8.3 Reading flash memory data into the AHB buffer

Reading the content from the same address as it was written to provides the following sequence of bytes, identical to the write case:

- 01...02...03...04...05...06...07...08

The following table shows the content against each TX buffer entry.

Table 235. Resulting AHB buffer content

AHB buffer entry	Content
0	8_07_06_05_04_03_02_01h

38.8.3.1 Readout of the AHB buffer through memory-mapped read

The AHB buffer content appears at read access on the AMBA AHB interface at the QuadSPI module boundary:

- 32-bit read access: 4_03_02_01h
- 32-bit read access: 8_07_06_05h
- 64-bit read access: 8_07_06_05_04_03_02_01h

38.9 Driving flash memory control signals in single and dual modes

In single and dual modes, the serial flash memory devices that can connect to the QuadSPI module expect additional control signals on the inputs, which are connected to IOFA[3], IOFA[2], IOFB[3], and IOFB[2] in the quad mode. For easy interfacing, the

outputs IOFA[3:2] for flash memory A and the IOFB[3:2] for flash memory B are driven to the logic state given by the configuration fields MCR[ISD3FA], MCR[ISD2FA], MCR[ISD3FB], and MCR[ISD2FB].

These outputs are driven all the time to the logic level programmed in the MCR except the time when quad commands of the serial flash memory are executed. See the specifications of the related serial flash memory device for details about the inactive level.

38.10 Serial flash memory devices

Several different vendors make flash memory devices with a QuadSPI interface. At present, there is no set standard for the QuadSPI instruction set. The most common commands currently have the same instruction code for all vendors; however, some commands are unique to specific vendors. Some of the example sequences are provided in the following sections.

38.10.1 Example sequences

This section provides the example sequences of the QuadSPI module.

Table 236. Exit 4 x I/O read enhance performance mode (XIP) (Macronix) and read status

Instruction	Pad	Operand	Description
CMD	0h	EBh	4xIO read command
ADDR	2h	18h	24-bit address to be sent on four pads
MODE	2h	0h	2 mode cycles (exit XIP)
DUMMY	0h	4h	4 dummy cycles
READ	2h	8h	Read 64 bits
CMD	0h	5h	Read Status register
READ	0h	1h	Status register data
STOP	0h	0h	Stop, instruction over

38.10.1.1 Read command (Spansion hyperflash memory)

This section provides the read command sequences (Spansion hyperflash memory) of the QuadSPI module.

Table 237. Read command (Spansion hyperflash memory)

Instruction	Pad	Operand	Comment
CMD_DDR	3h	A0h	Read command with continuous burst type
ADDR_DDR	3h	18h	24-bit row address
CADDR_DDR	3h	10h	16-bit column address with lower 3 bits valid and rest 0
Dummy	3h	0hF	15 dummy cycles
READ_DDR	3h	4h	32-bit data read on 8 pads
STOP	3h	0h	Stop, instruction over

Hyperflash memory is a word addressable flash memory. This means that each address accesses a word-wide (two-byte) data value. The software should ensure that when Hyperflash memory is connected to the controller, the value of SFACR[WA] must be 1. With this value, the controller remaps a byte addressable access to a word addressable access.

38.10.1.2 Read status register (Spansion hyperflash memory)

This section provides details related to the read status register of the QuadSPI module.

Table 238. Read status register (Spansion hyperflash memory)

Instruction	Instruction sequence	Pad	Operand	Description
CMD_DDR	Read Pre Command	3h	0h	Write command with wrapped burst type
ADDR_DDR		3h	18h	24-bit row address (0000AAh)
CADDR_DDR		3h	10h	16-bit column address with lower 3 bits valid and rest 0(0005h) treated as command
CMD_DDR		3h	0h	Write command with wrapped burst type
CMD_DDR		3h	70h	Write data to be sent to flash memory as pre-command
CMD_DDR	Command phase (fourth/final chip select phase)	3h	A0h	Read command with continuous burst type
ADDR_DDR		3h	18h	24-bit row address
CADDR_DDR		3h	10h	16-bit column address with lower 3 bits as valid and rest 0
DUMMY		3h	0hF	15 dummy cycles
READ_DDR		3h	4h	32-bit data read on 8 pads
STOP		3h	0h	Stop, instruction over

38.10.1.3 Word program (Spansion hyperflash memory)

This section provides the word program (Spansion hyperflash memory) of the QuadSPI module.

Table 239. Word program (Spansion hyperflash memory)

Instruction	Instruction sequence	Pad	Operand	Description
CMD_DDR	Unlock sequence 1 (first chip select phase)	3h	0h	Write command with wrapped burst type
CMD_DDR		3h	0h	8-bit address 00h treated as command
CMD_DDR		3h	0h	8-bit address 00h treated as command
CMD_DDR		3h	AAh	8-bit address AAh treated as command

Table continues on the next page...

Table 239. Word program (Spansion hyperflash memory) (continued)

CADDR_DDR		3h	10h	16-bit column address with lower 3 bits valid and rest 0(0005h) treated as command
CMD_DDR		3h	0h	Write command with wrapped burst type
CMD_DDR		3h	AAh	Write data to be sent to flash memory as pre-command
CMD_DDR	Unlock sequence 2 (second chip select phase)	3h	0h	Write command with wrapped burst type
CMD_DDR		3h	0h	8-bit address 00h treated as command
CMD_DDR		3h	0h	8-bit address 00h treated as command
CMD_DDR		3h	55h	8-bit address 55h treated as command
CADDR_DDR		3h	10h	16-bit column address with lower 3-bits valid and rest 0(0002h) treated as command
CMD_DDR		3h	0h	Write command with wrapped burst type
CMD_DDR		3h	55h	Write data to be sent to flash memory as pre-command
CMD_DDR	Program setup phase (third chip select phase)	3h	0h	Write command with wrapped burst type
CMD_DDR		3h	0h	8-bit address 00h treated as command
CMD_DDR		3h	0h	8-bit address 00h treated as command
CMD_DDR		3h	AAh	8-bit address AAh treated as command
CADDR_DDR		3h	10h	16-bit column address with lower 3 bits valid and rest 0(0005h) treated as command
CMD_DDR		3h	0h	Write command with wrapped burst type
CMD_DDR		3h	A0h	Write data to be sent to flash memory as pre-command

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-3003ab62ac

Table 239. Word program (Spansion hyperflash memory) (continued)

CMD_DDR	Command phase (fourth/final chip select phase)	3h	0h	Write command with wrapped burst type
ADDR_DDR		3h	18h	24-bit row address
CADDR_DDR		3h	10h	16-bit column address with lower 3 bits as valid and rest as 0
WRITE_DDR		3h	2h	2-byte data written on 8 pads (D1D2)
STOP		3h	0h	Stop, instruction over

38.10.1.4 Fast read sequence (Macronix/Numonyx/Spansion/Winbond)

The following table shows the fast read sequence for Macronix/Numonyx/Spansion/Winbond flash memories.

Table 240. Fast read sequence

Instruction	Pad	Operand	Description
CMD	0h	Bh	Fast read command = 0Bh
ADDR	0h	18h	24 address bits to be sent on one pad
DUMMY	0h	8h	Eight dummy cycles
READ	0h	4h	Read 32 bits on one pad
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

38.10.1.5 Fast dual I/O DT read sequence (Macronix)

The following table shows the fast dual I/O DT read sequence for Macronix flash memories.

Table 241. Fast dual I/O DT read sequence

Instruction	Pad	Operand	Description
CMD	0h	BDh	Fast dual I/O DT read command = BDh
ADDR_DDR	1h	18h	24 address bits to be sent on two pads in the DDR mode
MODE4_DDR	1h	0h	P2=P0 or P3=P1 is necessary. See the Macronix data sheet for details. One clock cycle for mode.
DUMMY	0h	6h	Six dummy cycles
READ_DDR	1h	4h	Read 32 bits on two pads in the DDR mode
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e54-6321-4888-b251-90d8a6b62c

38.10.1.6 Fast read quad output (Winbond)

The following table shows the fast read quad output sequence for Winbond memories

Table 242. Fast read quad output sequence

Instruction	Pad	Operand	Description
CMD	0h	6Bh	Fast read quad output command = 6Bh
ADDR	0h	18h	24 address bits to be sent on one pad
DUMMY	0h	8h	Eight dummy cycles
READ	2h	4h	Read 32 bits on four pads
JMP_ON_CS	0h	0h	Jump to instruction 0 (CMD)

38.10.1.7 4 x I/O read enhance performance mode (XIP) (Macronix)

The following table shows the 4 x I/O read enhance performance mode for Macronix flash memories. The enhanced performance mode is also known as XIP mode.

Table 243. Fast read quad output sequence

Instruction	Pad	Operand	Description
CMD	0h	EBh	4xI/O read command = EBh
ADDR	2h	18h	24 address bits to be sent on four pads
MODE	2h	A5h	Two mode cycles
DUMMY	0h	4h	Four dummy cycles
READ	2h	4h	Read 32 bits on four pads
JMP_ON_CS	0h	1h	Jump to instruction 1 (ADDR)

When in XIP mode, the software must ensure that all the flash memories connected to the controller are in this mode. As a part of initializing the controller, all the flash memories might be enabled with XIP by carrying out dummy reads.

38.10.1.8 Dual command page program (Numonyx)

The following table shows the dual command page program sequence for Numonyx flash memories.

Table 244. Dual command page program sequence

Instruction	Pad	Operand	Description
CMD	1h	2h	Dual command page program = 02h on 2 pads
ADDR	1h	18h	24 address bits to be sent on two pads
WRITE	1h	20h	Write 32 bytes on two pads
STOP	0h	0h	Stop, instruction over

Provided under NDA only
COMPANY PROPRIETARY
Beijing eCU Automotive Technology Co., Ltd.
c8c46285-482-48a8-b251-3900a3ab62d1

38.10.1.9 Sector erase (Macronix/Numonyx/Spansion)

The following table shows the Sector erase sequence for the Macronix/Numonyx/Spansion flash memories.

Table 245. Sector erase sequence

Instruction	Pad	Operand	Description
CMD	0h	20h	Sector erase command = 20h
ADDR	0h	18h	24 address bits to be sent on one pad
STOP	0h	0h	Stop, instruction over

38.10.1.10 Read status register (Macronix/Numonyx/Spansion/Winbond)

The following table shows the read status register sequence for Macronix/Numonyx/Spansion/Winbond flash memories.

Table 246. Read status register sequence

Instruction	Pad	Operand	Description
CMD	0h	0h5	Read status register command = 05h
READ	0h	0h1	Read status register data
STOP	0h	0h	Stop, instruction over

38.10.1.11 Data learn instruction sequence

The following table shows the data learn sequence for 4 I/O flash memory.

Table 247. Data learn instruction sequence (4 I/O)

Instruction	Pad	Operand	Description
CMD	0h	6Bh	Fast read quad command = 6Bh
ADDR_DDR	2h	18h	24 address bits in the DDR mode to be sent on four pads
DUMMY	2h	8h	8-cycle dummy
DATA_LEARN	2h	1h	1-byte data learn
READ_DDR	2h	6h	Read 6 bytes in 4 pads[1]
JMP_ON_CS	0h	0h	Jump to Inst 0 (CMD)

1. Data learn instruction needs to be followed by a read DDR instruction of minimum 6 bytes in case of 4 I/O mode.

The following table shows the data learn sequence for 8 I/O (Spansion hyperflash memory) flash memory.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4605-482f-48a6-b251-390d3ab62461

Table 248. Data learn instruction sequence (8 I/O)

Instruction	Pad	Operand	Description
CMD_DDR	3h	AOh	Read command for hyperflash memory
ADDR_DDR	3h	18h	24-bit row address[1]
CADDR_DDR	3h	10h	16-bit column address
DUMMY	3h	Fh	15-cycle dummy
DATA_LEARN	3h	1h	1-byte data learn
READ_DDR	3h	Ch	Read 12 bytes in 8 pads[2]
STOP	3h	0h	Stop, instruction over

1. The address needs to be aligned, which means, no latency should be there between the RWDS edges.
2. The data learn instruction needs to be followed by a read DDR instruction of minimum 12 bytes in case of 8 I/O mode.

The following table shows the data learn sequence for 4 I/O flash memory.

Table 249. Data learn instruction sequence (4 I/O)

Instruction	Pad	Operand	Description
CMD	0h	6Bh	Fast read quad command = 6Bh
ADDR_DDR	2h	18h	24 address bits in DDR mode to be sent on four pads
DUMMY	2h	8h	8 dummy cycles
DATA_LEARN	2h	1h	1-byte data learn
READ_DDR	2h	6h	Read 6 bytes in 4 pads
JMP_ON_CS	0h	0h	Jump to Inst 0 (CMD)

The following table shows the data learn sequence for 8 I/O (Spansion Hyperflash) flash memory.

Table 250. Data learn instruction sequence (8 I/O)

Instruction	Pad	Operand	Description
CMD_DDR	3h	AOh	Read command for hyperflash memory
ADDR_DDR	3h	18h	24-bit row address[1]
CADDR_DDR	3h	10h	16-bit column address
DUMMY	3h	Fh	15-cycle dummy
DATA_LEARN	3h	1h	1-byte data learn
READ_DDR	3h	8h	Read 8 bytes in 8 pads
STOP	3h	0h	Stop, instruction over

1. The address needs to be aligned, which means, no latency should be there between the RWDS edges.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
8c46e2e5-482f-4a8-b251-390d3a

38.10.2 Dual-die flash memories

Certain serial flash memory vendors provide dual-die packages that are essentially two devices (dies) stacked within the same package to increase the memory capacity of a single package. These two devices within a package share the same data and clock pins, but have individual chip selects. QuadSPI controller provides support for two dual-die packages to be connected simultaneously. The following figure shows the two dual-die packages and the naming conventions used in this document. For simplicity, the data pins are shown to be unidirectional.

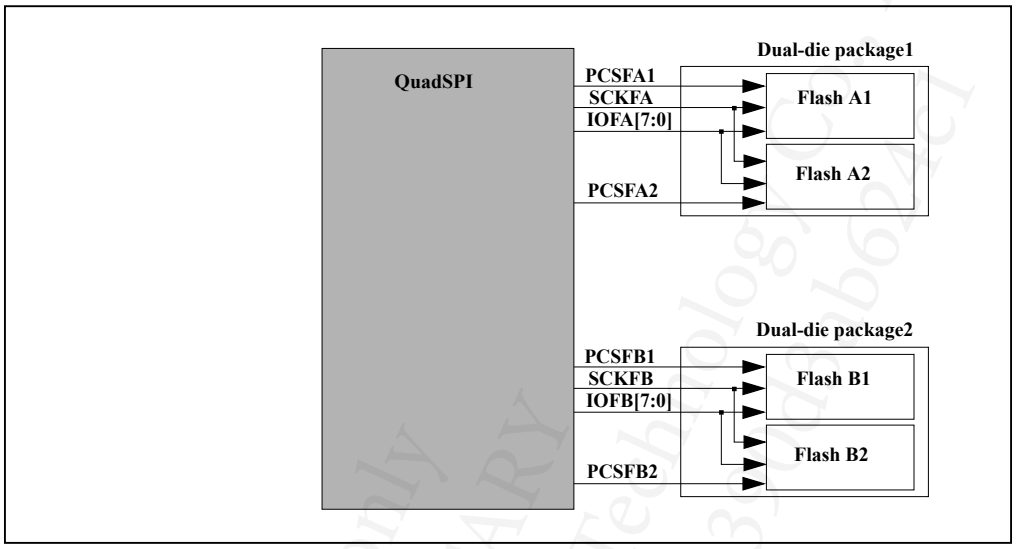


Figure 199. Generic configuration for dual-die support

Because the two devices within a package share the same I/O pads, they cannot function in the parallel mode. The software should ensure that when QuadSPI is configured in the parallel mode, the two selected flash memory devices belong to different dual-die packages.

38.10.3 Boot initialization sequence

The following is the recommended sequence of steps for booting from QuadSPI:

NOTE

Before you start, you must ensure that the system is out of reset and flash memory is available. Corresponding clocks and I/O pins must be configured.

1. Request is sent to QuadSPI for address 0h of flash memory.

   The reset command sequence in QuadSPI has 3h (basic read command) that is applicable to the flash memories.
2. The first few bytes of data are read from the flash memory. These bytes contain the following information:
   - The total sizes of all the flash memories connected on board
   - Support of the DDR mode
   - Frequency of the DDR operation
   - Mode entry sequence
   - 24-bit or 32-bit addressing (assuming 2- to 4-bit for the first accesses)
3. All the serial flash memories are configured:
   - Quad mode is enabled
   - Dummy reads to enter XIP

4. QuadSPI is configured:
   - Parallel enable set
   - LUT is configured for the highest performance reads
   - DDR mode is enabled (if applicable)
   - Buffers are configured
5. Serial flash memory clock frequency increases.
6. Boot reads happen in the parallel, DDR enabled, quad output mode.

38.11 Sampling of serial flash memory input data

38.11.1 Basic description

QuadSPI is used to read data from the serial flash memory device. Depending on the actual implementation, there is a delay between the internal clocking in the QuadSPI module and the external serial flash memory device. See the following figure for an overview of this scheme.

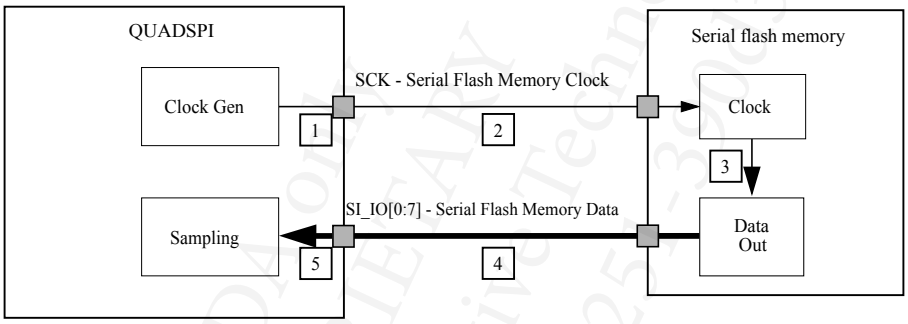


Figure 200. Serial flash memory sampling clock overview

The rising edge of the internal reference clock is taken as timing reference for the data output of the serial flash memory. After a time of $t_{Del,total}$ the data arrives at the internal sampling stage of the QuadSPI module. Considering the figure provided here, the following parts of the delay chain contribute to $t_{Del,total}$:

- Output delay of the serial flash memory clock output of the device containing the QuadSPI module
- Wire delay of application/PCB from the device containing the QuadSPI module to the external serial flash memory device
- Clock to data out delay of the external serial flash memory device, including input and output delays
- Wire delay of application/PCB from the external serial flash memory device to the device containing the QuadSPI module
- Device delay corresponding to the input data

NOTE

The $t_{Del,total}$ is specific to the characteristics of the actual implementation.

38.11.2 DQS sampling method

38.11.2.1 Basic description

In the DQS mode, the data strobe signal (DQS/RWDS) is used to sample the read data. Here, both DQS and the data sent by the flash memory move in the same direction; therefore, it is relatively easier to achieve at higher frequencies.

When using DQS for SDR reads, QuadSPI internally samples the incoming data on the rising edge of the strobe signal.

The next figure shows the sampling read data in the SDR mode using the DQS.

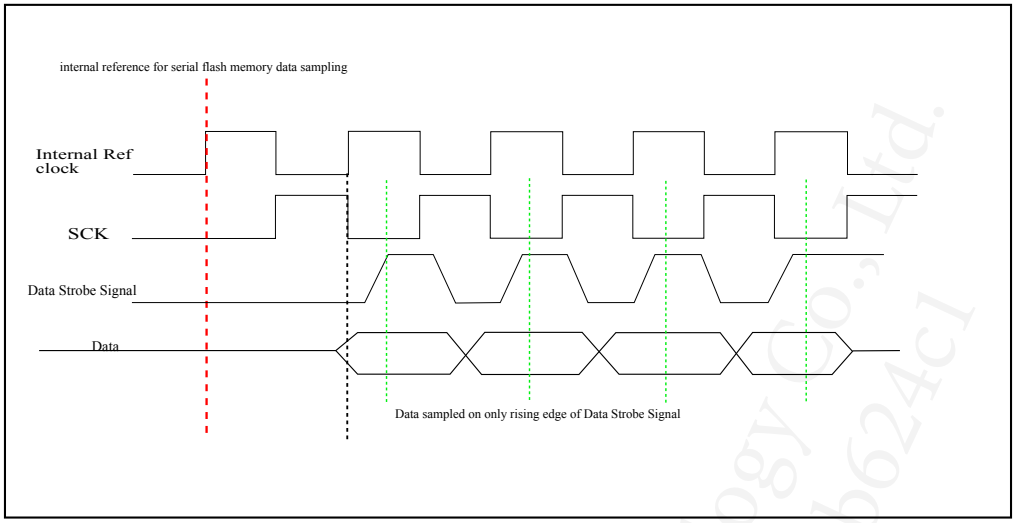


Figure 201. Data strobe functionality in SDR mode

When using the DQS for DDR reads, QuadSPI internally samples the incoming data on both the edges of the strobe signal. See the next figure for details.

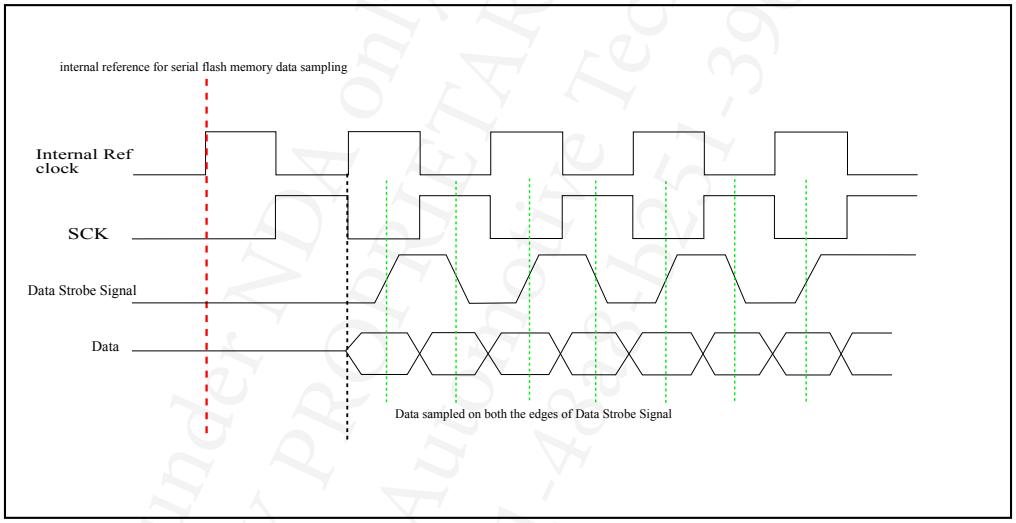


Figure 202. Data strobe functionality in DDR mode

38.11.2.2 External DQS

In serial flash memories supporting DQS, the data strobe signal is an output from the flash memory device that indicates when data is being transferred from the flash memory to the host controller. The data is then captured by the controller on:

- Only one edge (either rising or falling edge) of DQS signal in the SDR mode
- Both rising/falling edges of the DQS signal in the DDR mode

Some serial flash memories (for example, HyperFlash) provide the data strobe output (RWDS) with latency cycles included in between the ongoing read transactions. The data is sampled on both the edges of this signal taking into consideration that when no signal is provided by the flash memory between ongoing transactions, data is not sampled then, and at the end of the transaction, the required number of data to be fetched remains the same. See the next figure for details.

Processed under NDA only
NXP CONFIDENTIAL PROPRIETARY
Copyright (c) 2020 Autobrains Technology Co., Ltd.
Serial: 4821-4a81b251-390c3ab624c1

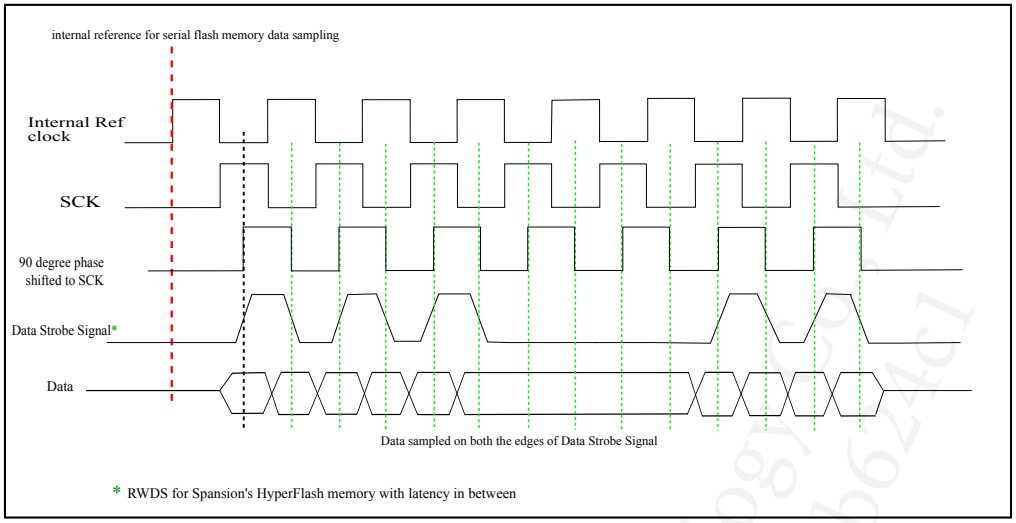


Figure 203. Data strobe functionality with latency included

This can be enabled by programming MCR[DQS_FA_SEL] = "11" for flash memory A and MCR[DQS_FB_SEL] = "11" for flash memory B. This mode supports the following configuration:

- Both high/low frequency delay chain manual programming using DLLCRA[SLV_DLY_COARSE] or DLLCRB[SLV_DLY_COARSE]
- DLL-assisted sampling only using DLLCRA[DLLEN] and DLLCRB[DLLEN]
- DLL-assisted auto data learning using DLLCRA[DLLEN] and DLLCRB[DLLEN] along with MCR{DLPEN]
- MCR[CK2_DCARS_FB] and MCR[CK2_DCARS_FA] should be disabled as DLL is used

NOTE

This mode may not be available on the chip. See the "Supported read modes" section in the chip-specific QuadSPI information for the read modes that this chip supports.

38.11.2.3 Pad loopback

The internal clock is loop-backed from the dummy internal pad to compensate data pad delays. This can be enabled by configuring the value of MCR[DQS_FA_SEL] as "01" for flash memory A and the value of MCR[DQS_FB_SEL] as "01" for flash memory B. This mode can be used with the following configuration:

- High/low frequency delay chain manual programming using DLLCRA[SLV_DLY_COARSE] or DLLCRB[SLV_DLY_COARSE]
- DLL-assisted sampling only using DLLCRA[DLLEN] and DLLCRL[DLLEN]
- DLL-assisted auto data learning using DLLCRA[DLLEN] and QuadPI_DLLCRB[DLLEN] along with MCR[DLPEN]

NOTE

This mode may not be available on the chip. See the "Supported read modes" section in the chip-specific QuadSPI information for the read modes that this chip supports.

38.11.2.4 External center-aligned read strobe

QuadSPI supports serial flash memories using the CRS protocol. Using this protocol, in addition to the SCK, the host controller provides a 90° phase shifted SCK that allows the flash memory to generate valid data timed off SCK edges and a DQS signal output that is timed off the 90° phase shifted SCK edges that is centered within the valid read data window. This allows the host controller to easily capture read data from the flash memory by using the data read strobe transitions and thereby allows

for a higher performance operation. As the flash memory is the originator of the center-aligned data read strobe as well as the read data, the temporal timing phase alignment between these signals is easier to guarantee across PVT (process, voltage, and temperature) corners.

The following figure shows data strobe functionality using CRS protocol.

NOTE

This mode might not be available on the chip. See the "Supported read modes" section in the chip-specific QuadSPI information for the read modes that this chip supports.

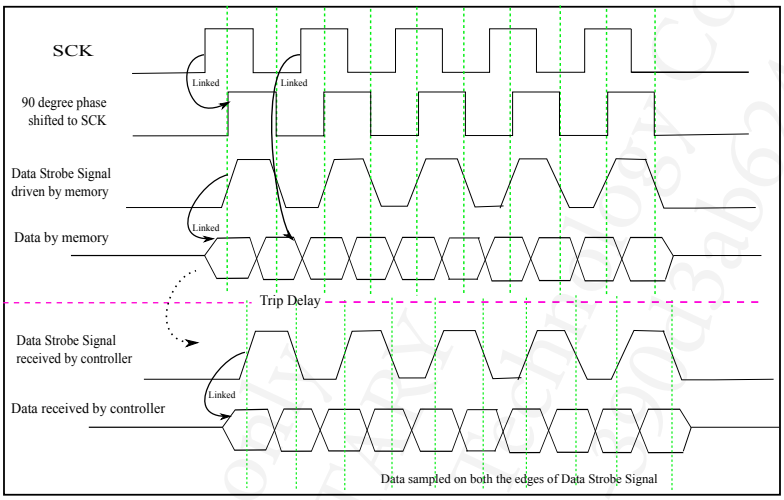


Figure 204. Data strobe functionality using the CRS protocol

38.12 Data learning

38.12.1 Basic description

Data learning is used to manage varying data valid windows from the flash memory as well as any variations in the chip based on PVT conditions in the DDR mode. QuadSPI provides this feature via the DATA_LEARN instruction for all flash memories, irrespective of whether the flash memory supports it. The DATA_LEARN instruction accepts an operand that defines the number of bits of the known pattern for which the data learning has to be done. The known pattern is provided in the DLPR register.

QuadSPI supports the following data learning:

- DQS

The following sections explain data learning in detail.

38.12.2 DQS sampling method

The semi-automatic data learning is supported in the DQS sampling method. To start data learning:

1. Configure a known pattern in the Data Learn register inside flash memories that support data learning. The pattern should be selected to have multiple low and high transitions in the data bus.

   NOTE

   Certain flash memories provide data learning as a feature for DDR data reads.

2. Set up a QuadSPI DDR data read sequence, inserting a DATA_LEARN instruction before the READ_DDR instruction. The flash memory returns the data learning pattern in between dummy cycles in every read command. Each I/O gives the same DLP value for every clock edge.

NOTE

For flash memories that *do not* support data learning, configure a known location in flash memory with a known pattern. The pattern should not have 0h or FFh included. The pattern should be selected to have multiple low and high transitions in the data bus. Set up a QuadSPI DDR data read sequence, inserting a DATA_LEARN instruction before the READ_DDR instruction. The address in the SFAR register should point to the known location within the pattern. Configure the known pattern in DLPR[DLPV]. See Programming the data learning pattern in flash memory for details on how the data has to be ordered in memory for correct operation.

3. Select a sampling point. See chip-specific QuadSPI information for selection of the sampling point.
4. Initiate the read via a peripheral transaction.
5. QuadSPI reads the data from the flash memory. It then encounters the DATA_LEARN instruction and samples the incoming data on both edges (rising and falling) of the DQS.

NOTE

If the data from the flash memory does not match the data learning pattern, the FR[DLPFF] flag is set. The QuadSPI module reports no sampling point automatically.

6. Repeat steps 2-7 with varying VT conditions for a particular process until FR[DLPFF] is set. The sampling points where no FR[DLPFF] is set signify a valid setting.

In the case of multiple valid settings, the software should select the middle point. The sampling point should be fixed within the above selected point for the next READ transactions.

NOTE

Ensure that QuadSPI is not accessed during calibration.

NOTE

Ensure that there is no latency between DQS edges while performing data learning using the DQS sampling method.

This feature might be used to auto-calibrate the sampling point for DDR reads, and auto-calibration might be triggered at fixed intervals, or depending on a change in PVT conditions.

In case the sampling point needs to be changed based on initiating options (such as, DLL lock status, time period, or software forced interrupt), data learn instruction has to be re-initiated.

The following figure shows data learning in DQS mode.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-9213-1901313b6241

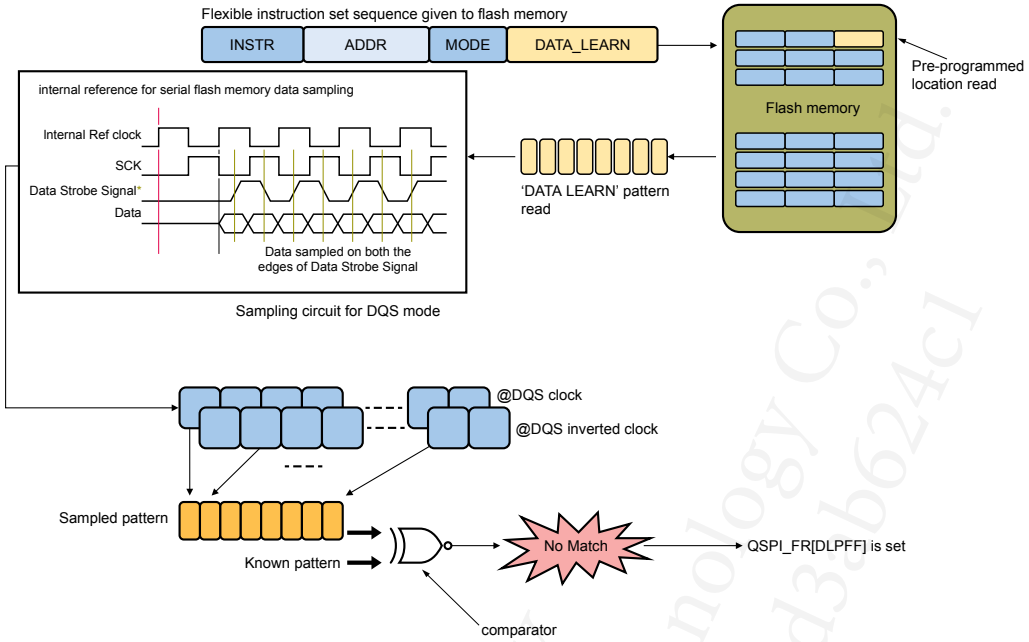


Figure 205. Data learning with DQS sampling method

NOTE

This mode might not be available on the chip. See the chip-specific QuadSPI information for the read modes that this chip supports.

38.12.3 Programming the data learning pattern in flash memory

Exmaple scnerio: If the value of DLPR[DLPV] is 43h, QuadSPI tries to match this sample on every incoming data line for single, dual, quad, or octal configurations. The following table shows the data programmed in flash memory with endianness taken into consideration.

Table 251. Programming the data learning pattern in flash memory

Endianness	Single IO	Dual IO	Quad IO	Octal IO
BE	43h	300Fh	0F0000FFh	00FF00000000FFFFh
LE	43h	0F30h	FF00000Fh	FFFF0000000000FFh

38.13 DLL and delay chain usage

The DLL is a general-purpose, dynamically adaptive clock delay module. It provides the ability to select a quantized delay (in fractions of the clock period) regardless of on-chip variations such as process, voltage, and temperature (PVT). The DLL is suitable for applications where accurate delay adjustment is required, such as in case of DDR interfaces.

DLL working concept

The DLL control loop consists of a counter, reference delay line, and phase detector which operate on the ref_clock reference clock input. The reference clock (clk_ref) is fed into the reference delay line. After reset, a single delay tap is selected. A phase detector is used to detect the condition where a half shift has occurred. In addition to this, signals are generated to either increment or decrement the counter, which controls the delay line (if the half-phase detect condition is not met). Any changes in the delay of the individual elements of the delay chain (because of PVT) automatically cause the phase detector logic to determine if a change in the counter value is required. After the half-phase shift is detected, an internal lock signal is generated, and after a lock event occurs, the slave delay chain is ready to be updated based on a triggering event.

Slave delay chain programming sequence—

Following is the programming sequence for DLL bypass mode.

1. Program DLLCRA[SLV_EN]=1, DLLCRA[SLV_DLL_BYPASS]=1, and DLLCRA[SLAVE_AUTO_UPDT]=0.
2. Program the following fields to provide the desired DQS delay for sampling: DLLCRA[SLV_FINE_OFFSET], DLLCRA[SLV_DLY_COARSE], and DLLCR[FREQEN]. See the chip-specific QuadSPI information for the supported programming settings.
3. Program DLLCRA[SLV_UPD]=1 to load these values in the slave delay chain.
4. Check the slave delay chain update status by polling DLLSR[SLVA_LOCK]=1 and clear DLLCRA[SLV_UPD] after confirming the update state.

NOTE

This programming sequence corresponds to flash memory A. Similar steps apply for flash memory B.

Following is the programming sequence for DLL auto update mode.

1. Program DLLCRA[SLV_EN]=1, DLLCRA[SLV_DLL_BYPASS]=0, and DLLCRA[SLAVE_AUTO_UPDT]=1.
2. Program the DLL configuration by using DLLCRA[DLL_REFCNTR] and DLLCRA[DLLRES]. See the chip-specific QuadSPI information for the supported DLL configuration settings.
3. Program the slave settings to delay DQS by using these fields: DLLCRA[SLV_FINE_OFFSET], DLLCRA[SLV_DLY_OFFSET], and DLLCR[FREQEN]. See the chip-specific QuadSPI information for the supported settings.
4. If offset delay needs to be updated on the slave chain, program DLLCRA[SLV_UPD]=1.
5. Enable DLL by programming DLLCRA[DLLEN]=1 and reset DLLCRA[SLV_UPD]=0. Slave delay chain is updated automatically and can be checked by polling DLLSR[SLVA_LOCK]==1

NOTE

This programming sequence corresponds to flash memory A. Similar steps apply for the flash memory B.

38.14 Data input hold requirement of flash memory

In the DDR mode, the data is valid only for half clock cycle. It is difficult to meet the data input hold time requirement for flash memory. QuadSPI internally delays the data sent to flash memory so that it becomes easy to meet the hold requirement. The FLSHCR[TDH] is used for this purpose. If FLSHCR[TDH] is configured to 0, then the data sent to flash memory is aligned with the internal reference clock of QuadSPI. If it is 1, then the data is aligned to 2x internal reference half clock.

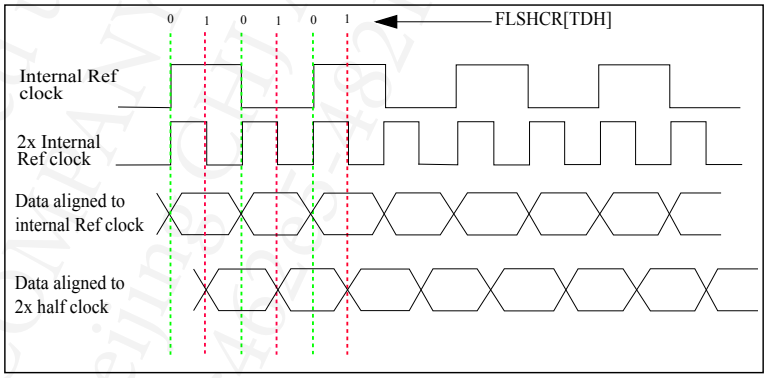


Figure 206. Data hold

38.15 Glossary

AHB	Advanced high-performance bus, a version of AMBA
AMBA	Advanced microcontroller bus architecture
APB	Advanced peripheral bus
BE	Big endian byte ordering
CRS	Center aligned read strobe
CS	Chip select
FRAD	Flash region access descriptor
I/O	Input output, I/O lines are also referred to as pads in this chapter
IFM	Individual flash memory mode
LE	Little endian
MDAD	Master domain access descriptors
MGID	Master-Group identifier
PFM	Parallel flash memory mode
PCS	Peripheral chip select
SCK	Serial communications clock
SFM	Serial flash memory

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 39
Ultra Secured Digital Host Controller (uSDHC)

39.1 Chip-specific uSDHC information

39.1.1 Chip-specific uSDHC information

uSDHC on this chip does not implement DLL_CTRL (60h) or DLL_STATUS (64h). Disregard their references in this document.

39.2 Introduction

uSDHC provides the interface between the host system and the SD/SDIO/MMC cards, as depicted in Figure 207. The module acts as a bridge, passing host bus transactions to the SD/SDIO/MMC cards by sending commands and performing data accesses to/from the cards. It handles the SD/SDIO/MMC protocols at the transmission level.

The following are brief descriptions of the cards supported by uSDHC:

- The Multi Media Card (MMC) is a universal low-cost data storage and communication media designed to cover a wide array of applications including mobile video and gaming. Previous MMC cards were based on a 7-pin serial bus with a single data pin, while the new high speed MMC communication is based on an advanced 11-pin serial bus designed to operate in the low-voltage range.
- The Secure Digital Card (SD) is an evolution of the old MMC technology. It is specifically designed to meet the security, capacity, performance, and environmental requirements inherent in newly emerging audio and video consumer electronic devices. The physical form factor, pin assignment, and data transfer protocol are forward compatible with the old MMC (with some additions).
- Under the SD protocol, system connection can be categorized into memory card, I/O card, and combo card, which have both memory and I/O functions. The memory card invokes a copyright protection mechanism that complies with the security of the SDMI standard. The I/O card, which is also known as SDIO card, provides high-speed data I/O with low-power consumption for mobile electronic devices. For the sake of simplicity, the next figure does not show cards with reduced size or mini cards.

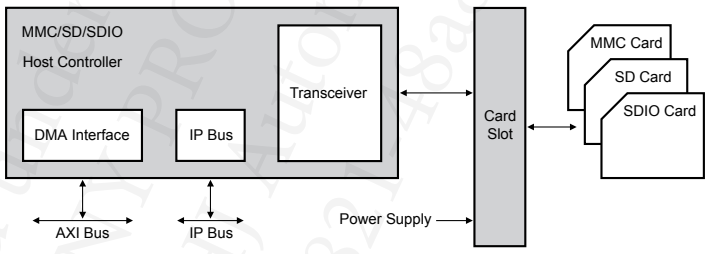


Figure 207. System connection of uSDHC

The following figure illustrates the block diagram for uSDHC.

Provided by NXP Only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c462e3-4824-4f1-9001-ab64c1

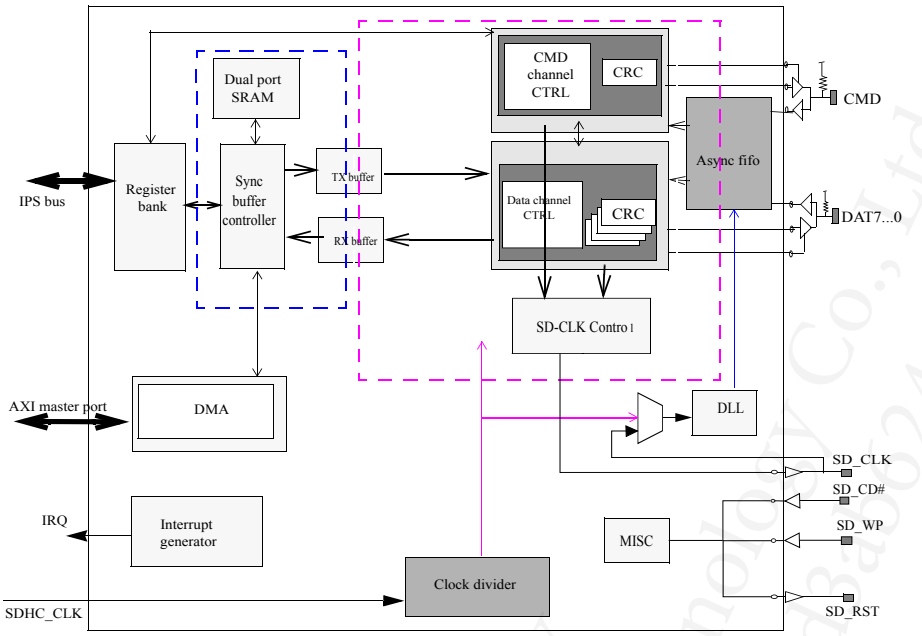


Figure 208. uSDHC block diagram

39.2.1 Features

Here are some of the features of the uSDHC module:

- Conforms to the SD Host Controller Standard Specification version 2.0/3.0
- Compatible with the MMC System Specification version 4.2/4.3/4.4/4.41/4.5/5.0/5.1
- Compatible with the SD Memory Card Specification version 3.0 and supports the Extended Capacity SD Memory Card
- Compatible with the SDIO Card Specification version 2.0/3.0
- Designed to work with SD Memory, miniSD Memory, SDIO, miniSDIO, SD Combo, MMC, MMC plus, and MMC RS cards
- Card bus clock frequency up to 208 MHz
- Supports 1-bit/4-bit SD and SDIO modes, and 1-bit/4-bit/8-bit MMC modes
  - Up to 832 Mbps of data transfer for SDIO cards using four parallel data lines in the Single Data Rate (SDR) mode
  - Up to 400 Mbps of data transfer for SDIO card using four parallel data lines in the Dual Data Rate (DDR) mode
  - Up to 832 Mbps of data transfer for SDXC cards using four parallel data lines in the Single Data Rate (SDR) mode
  - Up to 400 Mbps of data transfer for SDXC card using four parallel data lines in the Dual Data Rate (DDR) mode
  - Up to 1600 Mbps of data transfer for MMC cards using eight parallel data lines in the Single Data Rate (SDR) mode
  - Up to 3200 Mbps of data transfer for MMC cards using eight parallel data lines in the Dual Data Rate (DDR) mode
- Supports a single block/multi-block read and write
- Supports block sizes of 1 ~ 4096 bytes
- Supports the write protection switch for write operations
- Supports both synchronous and asynchronous abort
- Supports pause during the data transfer at block gap
- Supports SDIO Read Wait and Suspend Resume operations
- Supports Auto CMD12 for multi-block transfer

- Host can initiate non-data transfer command while data transfer is in progress
- Allows cards to interrupt the host in 1-bit and 4-bit SDIO modes; also supports interrupt period
- Embodies two fully configurable 256x32-bit FIFO for read/write data
- Supports internal DMA capabilities
- Support voltage selection by configuring vendor-specific register bit
- Supports Advanced DMA to perform linked memory access
- support Command queue mechanism

NOTE

This block can support all the above-listed speed mode and maximum data throughput. However, these may be specific to the device. See the corresponding chip-specific information or the device data sheet for accurate details.

39.2.2 Modes and operations

39.2.2.1 Data transfer modes

The uSDHC module can select the following modes for data transfer:

- SD 1-bit
- SD 4-bit
- MMC 1-bit
- MMC 4-bit
- MMC 8-bit
- Identification mode (up to 400 kHz)
- MMC full-speed mode (up to 26 MHz)
- MMC high-speed mode (up to 52 MHz)
- MMC HS200 mode(up to 200 MHz)
- MMC HS400 mode (200 MHz both edges)
- MMC DDR mode (52 MHz both edges)
- SD/SDIO full-speed mode (up to 25 MHz)
- SD/SDIO high-speed mode (up to 50 MHz)
- SD/SDIO UHS-I mode(up to 208 MHz in SDR mode, up to 50 MHz in the DDR mode)

NOTE

This block can support all the above listed speed mode and maximum clock frequency. However, these may be specific to the device. See the corresponding chip-specific information or the device data sheet for accurate details.

39.3 External signals

The following table describes the uSDHC external signals:

Table 252. uSDHC external signals

Signal	Description	Direction
CLK	Clock for MMC/SD/SDIO card	O

Table continues on the next page...

Table 252. uSDHC external signals (continued)

Signal	Description	Direction
CMD	CMD line connect to card	I/O
DATA7	DAT7 line in the 8-bit mode — Not used in other modes	I/O
DATA6	DAT6 line in the 8-bit mode — Not used in other modes	I/O
DATA5	DAT5 line in the 8-bit mode — Not used in other modes	I/O
DATA4	DAT4 line in the 8-bit mode — Not used in other modes	I/O
DATA3	DAT3 line in the 4/8-bit mode or configured as card detection pin. The bit may be configured as card detection pin in the 1-bit mode.	I/O
DATA2	DAT2 line or Read Wait in the 4-bit mode Read Wait in 1-bit mode	I/O
DATA1	DAT1 line in the 4/8-bit mode Also, used to detect interrupt in 1/4-bit mode	I/O
DATA0	DAT0 line in all the modes Also, used to detect busy state	I/O
CD_B	Card detection pin If not used (for the embedded memory), tie low to indicate that there is a card attached.	I
WP	Card write protect detect If not used (for the embedded memory), tie low to indicate that it is not write protected.	I
RESET_B	Card hardware reset signal, active low	O
VSELECT	IO power voltage selection signal	O
STROBE	Input clock for eMMC HS400 mode	I

39.3.1 Signals overview

uSDHC has 15 associated I/O signals.

- The CLK is an internally generated clock used to drive the MMC, SD, and SDIO cards.
- The CMD I/O is used to send commands and receive responses to and from the card. Eight data lines (DAT7~DAT0) are used to perform data transfers between the uSDHC module and the card.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing Qianxin Technology Co., Ltd
c0462e21-4821-4aa8-b251-390d3ab624c1

- The CD is card detection signal directly routed from the socket. This signal is active low (0). A low on CD# means that a card is inserted. It is optional for system implementation.
- The WP is write protection signals directly routed from the socket. This signal is active low (0). A high on WP means that the write protect switch is active. It is optional for system implementation.
- RST is an output signal used to reset the MMC card.
- VSELECT is an output signal used to change the voltage of the external power supplier. It is optional for system implementation.
- STROBE is input clock signal for eMMC HS400 mode.

If uSDHC needs to support a 4-bit data transfer, DAT7~DAT4 can also be optional and tied to high. If the uSDHC does not support the HS400 mode, STROBE can also be optional and tied to low.

39.4 uSDHC memory map/register definition

39.4.1 uSDHC register descriptions

This section includes the module memory map and detailed descriptions of all registers.

See the table below for the register memory map of uSDHC. All these registers only support 32-bit accesses.

NOTE

The uSDHC registers are 32-bit wide and only support 32-bit access.

39.4.1.1 uSDHC memory map

uSDHC base address: 402F_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	DMA System Address (DS_ADDR)	32	RW	0000_0000h
4h	Block Attributes (BLK_ATT)	32	RW	0001_0000h
8h	Command Argument (CMD_ARG)	32	RW	0000_0000h
Ch	Command Transfer Type (CMD_XFR_TYP)	32	RW	0000_0000h
10h	Command Response0 (CMD_RSP0)	32	RO	0000_0000h
14h	Command Response1 (CMD_RSP1)	32	RO	0000_0000h
18h	Command Response2 (CMD_RSP2)	32	RO	0000_0000h
1Ch	Command Response3 (CMD_RSP3)	32	RO	0000_0000h
20h	Data Buffer Access Port (DATA_BUFF_ACC_PORT)	32	RW	0000_0000h
24h	Present State (PRES_STATE)	32	RO	See description
28h	Protocol Control (PROT_CTRL)	32	RW	0880_0020h
2Ch	System Control (SYS_CTRL)	32	RW	0080_800Fh
30h	Interrupt Status (INT_STATUS)	32	W1C	0000_0000h
34h	Interrupt Status Enable (INT_STATUS_EN)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
38h	Interrupt Signal Enable (INT_SIGNAL_EN)	32	RW	0000_0000h
3Ch	Auto CMD12 Error Status (AUTOCMD12_ERR_STATUS)	32	RW	0000_0000h
40h	Host Controller Capabilities (HOST_CTRL_CAP)	32	RW	07F3_B407h
44h	Watermark Level (WTMK_LVL)	32	RW	0810_0810h
48h	Mixer Control (MIX_CTRL)	32	RW	8000_0000h
50h	Force Event (FORCE_EVENT)	32	WORZ	0000_0000h
54h	ADMA Error Status (ADMA_ERR_STATUS)	32	RO	0000_0000h
58h	ADMA System Address (ADMA_SYS_ADDR)	32	RW	0000_0000h
60h	DLL (Delay Line) Control (DLL_CTRL)	32	RW	0000_0000h
64h	DLL Status (DLL_STATUS)	32	RO	0000_0200h
68h	CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS)	32	RW	0000_0000h
70h	Strobe DLL control (STROBE_DLL_CTRL)	32	RW	0000_0000h
74h	Strobe DLL status (STROBE_DLL_STATUS)	32	RO	0000_0200h
C0h	Vendor Specific Register (VEND_SPEC)	32	RW	3000_7809h
C4h	MMC Boot (MMC_BOOT)	32	RW	0000_0000h
C8h	Vendor Specific 2 Register (VEND_SPEC2)	32	RW	0001_9006h
CCh	Tuning Control (TUNING_CTRL)	32	RW	0021_2800h
100h	Command Queuing Version (CQVER)	32	RO	0000_0510h
104h	Command Queuing Capabilities (CQCAP)	32	RO	0000_310Ah
108h	Command Queuing Configuration (CQCFG)	32	RW	0000_0000h
10Ch	Command Queuing Control (CQCTL)	32	RW	0000_0000h
110h	Command Queuing Interrupt Status (CQIS)	32	W1C	0000_0000h
114h	Command Queuing Interrupt Status Enable (CQISTE)	32	RW	0000_0000h
118h	Command Queuing Interrupt Signal Enable (CQISGE)	32	RW	0000_0000h
11Ch	Command Queuing Interrupt Coalescing (CQIC)	32	RW	0000_0000h
120h	Command Queuing Task Descriptor List Base Address (CQTDLBA)	32	RW	0000_0000h
124h	Command Queuing Task Descriptor List Base Address Upper 32 Bits (CQTDLBAU)	32	RW	0000_0000h
128h	Command Queuing Task Doorbell (CQTDBR)	32	RW	0000_0000h
12Ch	Command Queuing Task Completion Notification (CQTCN)	32	W1C	0000_0000h
130h	Command Queuing Device Queue Status (CQDQS)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
134h	Command Queuing Device Pending Tasks (CQDPT)	32	RO	0000_0000h
138h	Command Queuing Task Clear (CQTCLR)	32	RW	0000_0000h
140h	Command Queuing Send Status Configuration 1 (CQSSC1)	32	RW	0001_1000h
144h	Command Queuing Send Status Configuration 2 (CQSSC2)	32	RW	0000_0000h
148h	Command Queuing Command Response for Direct-Command Task (CQCRDCT)	32	RO	0000_0000h
150h	Command Queuing Response Mode Error Mask (CQRMEM)	32	RW	FDF9_A080h
154h	Command Queuing Task Error Information (CQTERRI)	32	RO	0000_0000h
158h	Command Queuing Command Response Index (CQCRI)	32	RO	0000_0000h
15Ch	Command Queuing Command Response Argument (CQCRA)	32	RO	0000_0000h

39.4.1.2 DMA System Address (DS_ADDR)

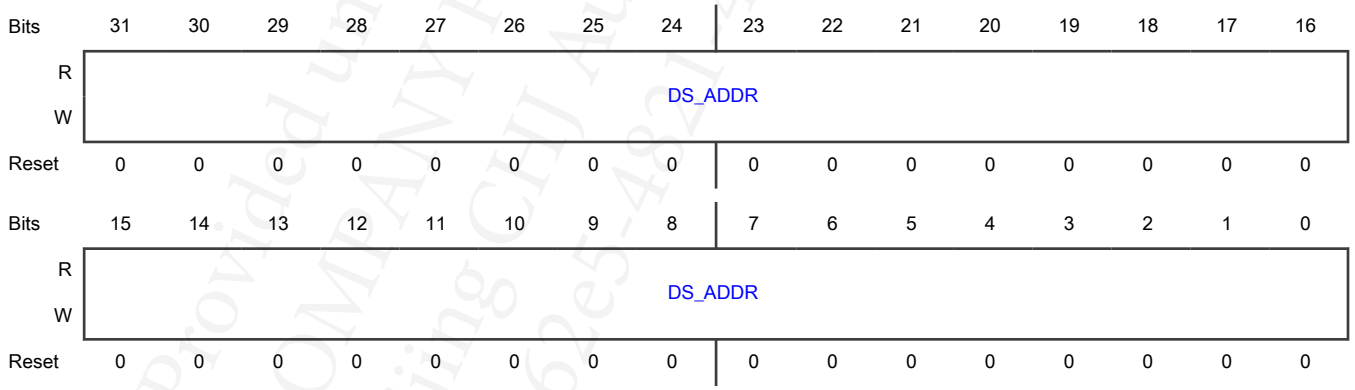
Offset

Register	Offset
DS_ADDR	0h

Function

This register contains the physical system memory address used for DMA transfers.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e54821748a-b251-3900a6a62a61

Fields

Field	Function
31-0 DS_ADDR	System address **DMA system address / argument 2** When ACMD23_ARGU2_EN is set to 0, SDMA uses this register as system address and supports only 32-bit addressing mode. Auto CMD23 cannot be used with SDMA. When ACMD23_ARGU2_EN is set to 1, SDMA uses ADMA System Address register (05Fh – 058h) instead of this register to support both 32-bit and 64-bit addressing. This register is used only for Argument2 and SDMA may use Auto CMD23. 1. **SDMA system address** Because the address must be word (4 bytes) aligned, the least two bits are reserved and always set to 0. When uSDHC stops a DMA transfer, this register points out the system address of the next contiguous data position. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). Read operation during transfers may return an invalid value. The host driver initializes this register before starting a DMA transaction. After DMA has stopped, the system address of the next contiguous data position can be read from this register. This register is protected during a data transfer. When data lines are active, write to this register is ignored. The host driver waits until the DLA field in the Present State register is cleared, before writing to this register. The uSDHC internal DMA does not support a virtual memory system. It only supports continuous physical memory access. Because this register supports dynamic address reflecting, when TC field is set, it automatically alters the value of internal address counter, so the software cannot change this register when TC field is set. Such restriction is also listed in Software restrictions. 2. **Argument 2** This register is used with the Auto CMD23 to set a 32-bit block count value to the argument of the CMD23 while executing Auto CMD23. If Auto CMD23 is used with ADMA, the full 32-bit block count value can be used. If Auto CMD23 is used without ADMA, the available block count value is limited by the Block Count register. 65535 blocks is the maximum value in this case.

39.4.1.3 Block Attributes (BLK_ATT)

Offset

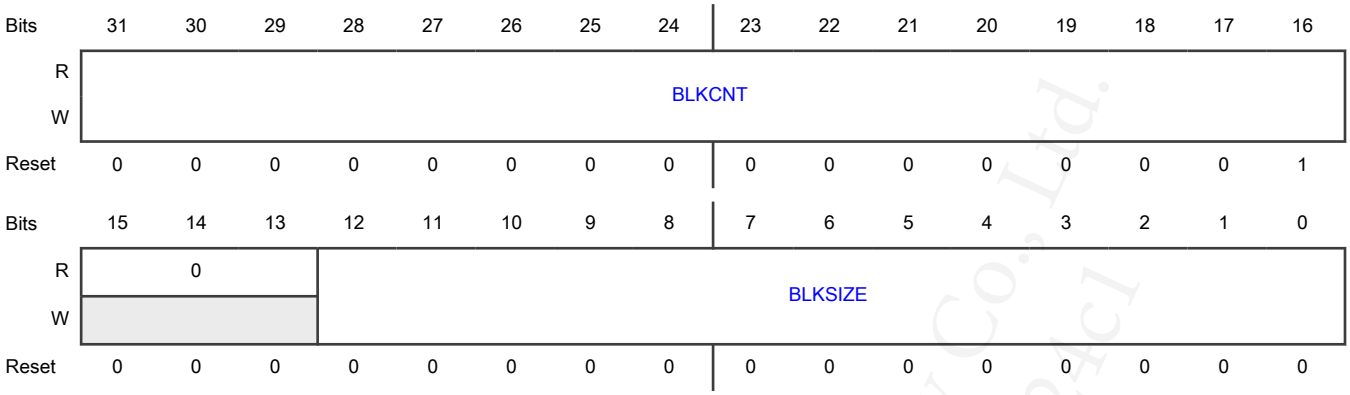
Register	Offset
BLK_ATT	4h

Function

This register is used to configure the number of data blocks and the number of bytes in each block.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c46225-4821-4a38-b2f1-590e5e24c0e

Diagram



Fields

Field	Function
31-16 BLKCNT	Blocks count for current transfer This field is enabled when the Block Count Enable field in the Transfer Mode register is set to 1 and is valid only for multiple block transfers. For single block transfer, this field always reads as 1. The host driver sets this field to a value between 1 and the maximum block count. The uSDHC module decrements the block count after each block transfer and stops when the count reaches zero. Setting the block count to zero results in no data blocks being transferred. This field should be accessed only when no transaction is executing (that is, after transactions are stopped). During data transfer, read operations on this field may return an invalid value and write operations are ignored. When saving transfer content because of a Suspend command, the number of blocks yet to be transferred can be determined by reading this field. The reading of this field should be applied after transfer is paused by stop at block gap operation and before sending the command marked as suspend. This is because when the Suspend command is sent out, uSDHC treats the current transfer as aborted and change the BLKCNT field back to its original value instead of keeping the dynamical indicator of the remaining block count. When restoring transfer content prior to issuing a Resume command, the host driver restores the previously saved block count. **NOTE** Although the BLKCNT field is 0 after reset, the read of reset value is 0x1. This is because when MSBSEL field is indicating a single block transfer, the read value of BLKCNT is always 1. 0000000000000000b - Stop count 0000000000000001b - 1 block 0000000000000010b - 2 blocks 1111111111111111b - 65535 blocks
15-13 —	Reserved
12-0	Transfer block size

Table continues on the next page...

Table continued from the previous page...

Field	Function
BLKSIZE	This field specifies the block size for block data transfers. Values ranging from 1 byte up to the maximum buffer size can be set. It can be accessed only when no transaction is executing (that is, after a transaction has stopped). Read operations during transfers may return an invalid value, and write operations are ignored. 0000000000000b - No data transfer 0000000000001b - 1 byte 0000000000010b - 2 bytes 0000000000011b - 3 bytes 0000000000100b - 4 bytes 0000111111111b - 511 bytes 0001000000000b - 512 bytes 0100000000000b - 2048 bytes 1000000000000b - 4096 bytes

39.4.1.4 Command Argument (CMD_ARG)

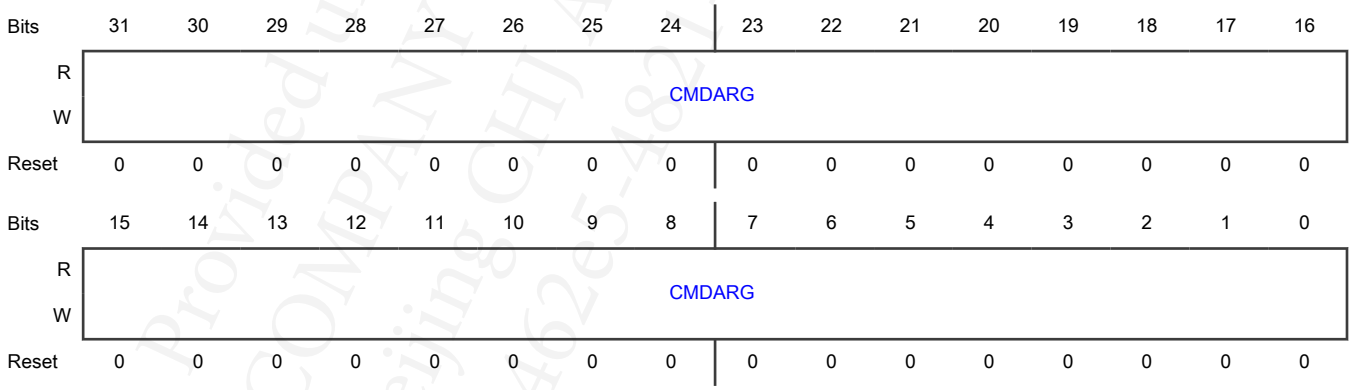
Offset

Register	Offset
CMD_ARG	8h

Function

This register contains the SD/MMC command argument.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482a-48a8-b251-390d3ab624c1

Fields

Field	Function
31-0 CMDARG	Command argument The SD/MMC command argument is specified as bits 39-8 of the command format in the SD or MMC specification. This field is write protected when the Command Inhibit (CMD) field in the Present State register is set.

39.4.1.5 Command Transfer Type (CMD_XFR_TYP)

Offset

Register	Offset
CMD_XFR_TYP	Ch

Function

This register is used to control the operation of data transfers. The host driver sets this register before issuing a command followed by a data transfer or before issuing a Resume command. To prevent data loss, uSDHC prevents writing to the bits, which are involved in the data transfer of this register, when data transfer is active. These fields are DPSEL, MBSEL, DTDSEL, AC12EN, BCEN, and DMAEN.

The host driver checks the Command Inhibit DAT field (PRES_STATE[CDIHB]) and the Command Inhibit CMD field (PRES_STATE[CIHB]) in the Present State register before writing to this register. When the CDIHB field in the Present State register is set, any attempt to send a command with data by writing to this register is ignored; when the CIHB field is set, any write to this register is ignored.

On sending commands with data transfer involved, it is mandatory that the block size is non-zero. Block count must also be non-zero, or indicated as a single block transfer (bit 5 of this register is '0' when written), or block count is disabled (bit 1 of this register is '0' when written), otherwise uSDHC ignores the sending of this command and do nothing. For write command, with all above restrictions, it is also mandatory that the write protect switch is not active (PRES_STATE[WPSPL] field of Present State register is '1'); otherwise, uSDHC also ignores the command.

If the commands with data transfer do not receive the response in 64 clock cycles, that is, if response time-out happens, uSDHC treats the external device, does not accept the command, and aborts the data transfer. In this scenario, the driver should issue the command again to retry the transfer. It is also possible that for some reason the card responds to the command but uSDHC does not receive the response, and if it is internal DMA (either simple DMA or ADMA) read operation, the external system memory is over-written by the internal DMA with data sent back from the card.

The table below shows the summary of how register settings determine the type of data transfer.

Table 253. Transfer type register setting for various transfer types

Multi/single block select	Block count enable	Block count	Function
0	Do not care	Do not care	Single transfer
1	0	Do not care	Infinite transfer
1	1	Positive number	Multiple transfer
1	1	Zero	No data transfer

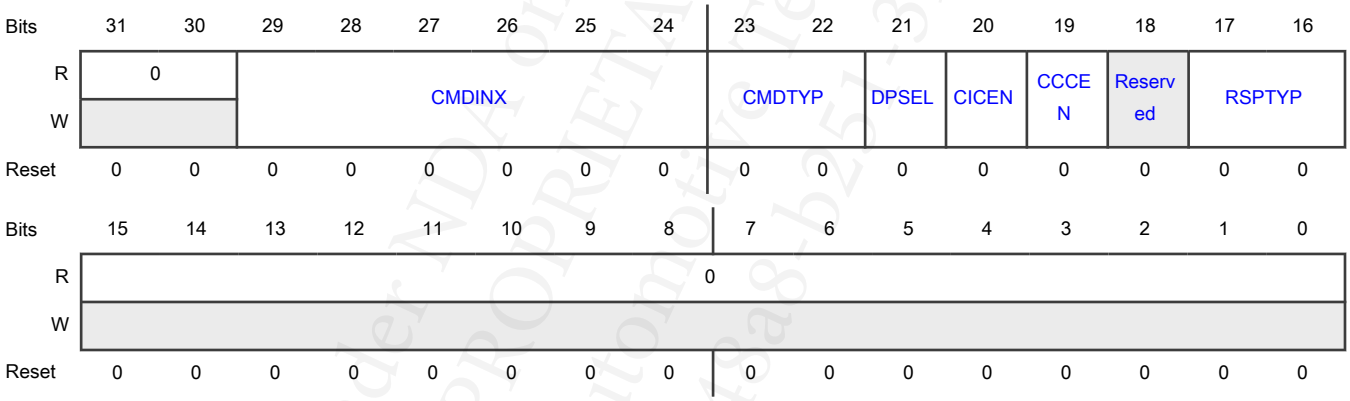
The table below shows the relationship between the Command Index Check Enable and the Command CRC Check Enable, regarding the Response Type bits as well as the name of the response type.

Table 254. Relationship between parameters and the name of the response type

Response type	Index check enable	CRC check enable	Name of response type
00	0	0	No response
01	0	1	R2
10	0	0	R3, R4
10	1	1	R1, R5, R6
11	1	1	R1b,R5b

- In the SDIO specification, response type notation for R5b is not defined. R5 includes R5b in the SDIO specification, but R5b is defined in this specification to specify that uSDHC checks the busy status after receiving a response. For example, usually CMD52 is used with R5, but the I/O abort command is used with R5b.
- The CRC fields for R3 and R4 are expected to be all 1 bits. The CRC check is disabled for these response types.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-24 CMDINX	Command index These fields are set to the command number that is specified in bits 45-40 of the command-format in the SD Memory Card Physical Layer Specification and SDIO Card Specification.
23-22 CMDTYP	Command type

Table continues on the next page...

Table continued from the previous page...

Field	Function
	There are three types of special commands: Suspend, Resume, and Abort. These fields are set to 00b for all other commands. • Suspend command: If the Suspend command succeeds, uSDHC assumes that the card bus has been released and that it is possible to issue the next command that uses the DATA line. Because uSDHC does not monitor the content of command response, it does not know if the Suspend command succeeded or not. It is the host driver's responsibility to check the status of the Suspend command and send another command marked as Suspend to inform uSDHC that a Suspend command was successfully issued. See Suspend Resume for more details. After the end bit of command is sent, uSDHC deasserts Read Wait for read transactions and stops checking busy for write transactions. In a 4-bit mode, the interrupt cycle starts. If the Suspend command fails, uSDHC maintains its current state, and the host driver restarts the transfer by setting the Continue Request field in the Protocol Control register. • Resume command: The host driver re-starts the data transfer by restoring the registers saved before sending the Suspend command and then sends the Resume command. The uSDHC module checks for a pending busy state before starting write transfers. • Abort command: If this command is set when executing a read transfer, uSDHC stops reads to the buffer. If this command is set when executing a write transfer, uSDHC stops driving the DATA line. After issuing the Abort command, the host driver should issue a software reset (Abort Transaction). 00b - Normal other commands 01b - Suspend CMD52 for writing bus suspend in CCCR 10b - Resume CMD52 for writing function select in CCCR 11b - Abort CMD12, CMD52 for writing I/O Abort in CCCR
21 DPSEL	Data present select This field is set to 1 to indicate that data is present and is transferred using the DATA line. It is set to 0 for the following: • Commands using only the CMD line (for example, CMD52) • Commands with no data transfer, but using the busy signal on DATA0 line (R1b or R5b (for example, CMD38)) **NOTE** In resume command, this field is set, and other bits in this register is set the same as when the transfer was initially launched. When the write protect switch is on, (that is, the WPSPL field is active as '0'), any command with a write operation ignored. When this field is set, while the DTDSEL field is 0, writes to the register Transfer Type are ignored. 0b - No data present 1b - Data present
20 CICEN	Command index check enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If this field is set to 1, uSDHC checks the Index field in the response to see if it has the same value as the command index. If it is not, it is reported as a Command Index Error. If this field is set to 0, the Index field is not checked. 0b - Disable command index check 1b - Enables command index check
19 CCCEN	Command CRC check enable If this field is set to 1, uSDHC checks the CRC field in the response. If an error is detected, it is reported as a Command CRC Error. If this field is set to 0, the CRC field is not checked. The number of bits checked by the CRC field value changes according to the length of the response. See RSPTYP[1:0] and Command Transfer Type (CMD_XFR_TYP). 0b - Disables command CRC check 1b - Enables command CRC check
18 —	Reserved
17-16 RSPTYP	Response type select 00b - No response 01b - Response length 136 10b - Response length 48 11b - Response length 48, check busy after response
15-0 —	Reserved

39.4.1.6 Command Response0 (CMD_RSP0)

Offset

Register	Offset
CMD_RSP0	10h

Function

This register is used to store part 0 of the response bits from the card.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c467e5-4821-48a8-9251-390d3ab5f4c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CMDRSP0	Command response 0 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this field for each response type.

39.4.1.7 Command Response1 (CMD_RSP1)

Offset

Register	Offset
CMD_RSP1	14h

Function

This register is used to store part 1 of the response bits from the card.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e54-2921-48a8-b251-390d8ab624c1

Fields

Field	Function
31-0 CMDRSP1	Command response 1 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this field for each response type.

39.4.1.8 Command Response2 (CMD_RSP2)

Offset

Register	Offset
CMD_RSP2	18h

Function

This register is used to store part 2 of the response bits from the card.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CMDRSP2	Command response 2 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this field for each response type.

39.4.1.9 Command Response3 (CMD_RSP3)

Offset

Register	Offset
CMD_RSP3	1Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c402e3-4821-4a8-b251-390d3ab624c1

Function

This register is used to store part 3 of the response bits from the card.

The table below describes the mapping of command responses from the SD bus to Command Response registers for each response type. In this table, R[ ] refers to a bit range within the response data as transmitted on the SD bus.

Table 255. Response bit definition for each response type

Response type	Meaning of response	Response field	Response register
R1,R1b (normal response)	Card status	R[39:8]	CMDRSP0
R1b (auto CMD12 response)	Card status for auto CMD12	R[39:8]	CMDRSP3
R2 (CID, CSD register)	CID/CSD register [127:8]	R[127:8]	{CMDRSP3[23:0], CMDRSP2, CMDRSP1, CMDRSP0}
R3 (OCR register)	OCR register for memory	R[39:8]	CMDRSP0
R4 (OCR register)	OCR register for I/O etc.	R[39:8]	CMDRSP0
R5, R5b	SDIO response	R[39:8]	CMDRSP0
R6 (publish RCA)	New published RCA[31:16] and card status[15:0]	R[39:9]	CMDRSP0

This table shows that most responses with a length of 48 (R[47:0]) have 32-bits of the response data (R[39:8]) stored in the CMDRSP0 register. Responses of type R1b (Auto CMD12 responses) have response data bits (R[39:8]) stored in the CMDRSP3 register. Responses with length 136 (R[135:0]) have 120-bits of the response data (R[127:8]) stored in the CMDRSP0, 1, 2, and 3 registers.

To be able to read the response status efficiently, uSDHC only stores part of the response data in the Command Response registers. This enables the host driver to efficiently read 32-bits of response data in one read cycle on a 32-bit bus system. Parts of the response, the Index field and the CRC, are checked by uSDHC (as specified by the Command Index Check Enable and the Command CRC Check Enable bits in the Transfer Type register) and generate an error interrupt if any error is detected. The bit range for the CRC check depends on the response length. If the response length is 48, uSDHC checks R[47:1], and if the response length is 136 the uSDHC checks R[119:1].

Because uSDHC may have a multiple block data transfer executing concurrently with a CMD_wo_DAT command, uSDHC stores the Auto CMD12 response in the CMDRSP3 register. The CMD_wo_DAT response is stored in CMDRSP0. This allows uSDHC to avoid overwriting the Auto CMD12 response with the CMD_wo_DAT and vice versa. When uSDHC modifies part of the Command Response registers, as shown in the table above, it preserves the unmodified bits.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMDRSP3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP3															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CMDRSP3	Command response 3 See Command Response3 (CMD_RSP3) for the mapping of command responses from the SD bus to this field for each response type.

39.4.1.10 Data Buffer Access Port (DATA_BUFF_ACC_PORT)

Offset

Register	Offset
DATA_BUFF_ACC_PORT	20h

Function

The Buffer Data Port register is for 32-bit data access by the Arm platform. When the internal DMA is enabled, any write to this field is ignored, and any read from this field always yields 0s.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATCONT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATCONT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46255-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-0 DATCONT	Data content This field is used to access the internal buffer.

39.4.1.11 Present State (PRES_STATE)

Offset

Register	Offset
PRES_STATE	24h

Function

The host driver can get status of uSDHC from this 32-bit read only register.

The host driver can issue CMD0, CMD12, CMD13 (for memory) and CMD52 (for SDIO) when the DATA lines are busy during a data transfer. These commands can be issued when Command Inhibit (CMD) is set to zero. Other commands are issued when Command Inhibit (DATA) is set to zero. Possible changes to the SD Physical Specification may add other commands to this list in the future.

NOTE

The reset value of Present State register depends on board connectivity.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLSL								CLSL	0			WPSPL	CDPL	0	CINST
W																
Reset	u	u	u	u	u	u	u	u	u	0	0	0	u	u	0	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TSCD	0		RTR	BREN	BWEN	RTA	WTA	SDOFF	PEROFF	HCKOFF	IPGOFF	SDSTB	DLA	CDIHB	CIHB
W																
Reset	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Fields

Field	Function
31-24 DLSL	DATA[7:0] line signal level

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4f0f-4f82-b251-37d3ab624c1

Table continued from the previous page...

Field	Function
	This field is used to check the DATA line level to recover from errors, and for debugging. This is especially useful in detecting the busy signal level from DATA0. The reset value is affected by the external pull-up / pull-down resistors. By default, the read value of this field after reset is 8'b11110111, when DATA3 is pulled down and the other lines are pulled up. 00000000b - Data 0 line signal level 00000001b - Data 1 line signal level 00000010b - Data 2 line signal level 00000011b - Data 3 line signal level 00000100b - Data 4 line signal level 00000101b - Data 5 line signal level 00000110b - Data 6 line signal level 00000111b - Data 7 line signal level
23 CLSL	CMD line signal level This field is used to check the CMD line level to recover from errors, and for debugging. The reset value is affected by the external pull-up / pull-down resistor, by default, the read value of this field after reset is 1'b1, when the command line is pulled up.
22-20 —	Reserved
19 WPSPL	Write protect switch pin level The Write Protect switch is supported for memory and combo cards. This field reflects the inverted value of the WP pin of the card socket. A software reset does not affect this field. The reset value is affected by the external write protect switch. If the WP pin is not used, it should be tied low, so that the reset value of this field is high and write is enabled. 0b - Write protected (WP = 1) 1b - Write enabled (WP = 0)
18 CDPL	Card detect pin level This field reflects the inverse value of the CD_B pin for the card socket. Debouncing is not performed on this field. This field may be valid, but is not guaranteed, because of propagation delay. Use of this field is limited to testing because it must be debounced by software. A software reset does not affect this field. A write to the Force Event Register does not affect this field. The reset value is affected by the external card detection pin. This field shows the value on the CD_B pin (that is, when a card is inserted in the socket, it is 0 on the CD_B input, and consequently, the CDPL reads 1. 0b - No card present (CD_B = 1) 1b - Card present (CD_B = 0)
17 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 CINST	Card inserted This field indicates whether a card has been inserted. The uSDHC module debounces this signal so that the host driver does not need to wait for it to stabilize. Changing from a 0 to 1 generates a Card Insertion interrupt in the Interrupt Status register. Changing from a 1 to 0 generates a Card Removal interrupt in the Interrupt Status register. A write to the Force Event Register does not affect this field. The Software Reset For All in the System Control register does not affect this field. A software reset does not affect this field. 0b - Power on reset or no card 1b - Card inserted
15 TSCD	Tape select change done This field indicates the delay setting is effective after write CLK_TUNE_CTRL_STATUS register. 0b - Delay cell select change is not finished. 1b - Delay cell select change is finished.
14-13 —	Reserved
12 RTR	Re-Tuning Request (only for SD3.0 SDR104 mode,and EMMC HS200 mode) Host controller may request host driver to execute re-tuning sequence by setting this field when the data window is shifted by temperature drift and a tuned sampling point does not have a good margin to receive correct data. This field is cleared when a command is issued with setting Execute Tuning field in MIX_CTRL register. Changing of this field from 0 to 1 generates Re-Tuning Event. See Interrupt status registers for more detail. This field isn't set to 1 if Sampling Clock Select in the MIX_CTRL register is set to 0 (using fixed sampling clock). 0b - Fixed or well tuned sampling clock 1b - Sampling clock needs re-tuning
11 BREN	Buffer read enable This status field is used for non-DMA read transfers. The uSDHC module implements an internal buffer to transfer data efficiently. This read only flag indicates that valid data exists in the host side buffer. If this field is high, valid data greater than the watermark level exist in the buffer. A change of this field from 1 to 0 occurs when some reads from the buffer (read DATPORT (Base + 0x20)) are made and the buffer hasn't valid data greater than the watermark level. A change of this field from 0 to1 occurs when there is enough valid data ready in the buffer and the Buffer Read Ready interrupt has been generated and enabled. 0b - Read disable 1b - Read enable
10 BWEN	Buffer write enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This status field is used for non-DMA write transfers. The uSDHC module implements an internal buffer to transfer data efficiently. This read only flag indicates if space is available for write data. If this field is 1, valid data greater than the watermark level can be written to the buffer. A change of this field from 1 to 0 occurs when some writes to the buffer (write DATPORT (Base + 0x20)) are made and the buffer hasn't valid space greater than the watermark level. A change of this field from 0 to 1 occurs when the buffer can hold valid data greater than the write watermark level and the Buffer Write Ready interrupt is generated and enabled. 0b - Write disable 1b - Write enable
9 RTA	Read transfer active This status field is used for detecting completion of a read transfer. This field is set for either of the following conditions: • After the end field of the read command • When writing a 1 to the Continue Request field in the Protocol Control register to restart a read transfer A transfer complete interrupt is generated when this field changes to 0. This field is cleared for either of the following conditions: • When the last data block as specified by block length is transferred to the System, that is, all data are read away from uSDHC internal buffer. • When all valid data blocks have been transferred from uSDHC internal buffer to the System and no current block transfers are being sent because of the Stop At Block Gap Request being set to 1. 0b - No valid data 1b - Transferring data
8 WTA	Write transfer active This status field indicates a write transfer is active. If this field is 0, it means no valid write data exists in uSDHC. This field is set in either of the following cases: • After the end field of the write command • When writing 1 to the Continue Request field in the Protocol Control register to restart a write transfer This field is cleared in either of the following cases: • After getting the CRC status of the last data block as specified by the transfer count (Single and Multiple) • After getting the CRC status of any block where data transmission is about to be stopped by a Stop At Block Gap Request During a write transaction, a Block Gap Event interrupt is generated when this field is changed to 0, as result of the Stop At Block Gap Request being set. This status is useful for the host driver in determining when to issue commands during Write Busy state.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No valid data 1b - Transferring data
7 SDOFF	SD clock gated off internally This status field indicates that the SD clock is internally gated off, because of buffer over / under-run or read pause without read wait assertion, or the driver set FRC_SDCLK_ON field is 0 to stop the SD clock in idle status. This field is for the host driver to debug data transaction on the SD bus. 0b - SD clock is active. 1b - SD clock is gated off.
6 PEROFF	IPG_PERCLK gated off internally This status field indicates that the IPG_PERCLK is internally gated off. This field is for the host driver to debug transaction on the SD bus. When IPG_CLK_SOFT_EN is cleared, IPG_PERCLK is gated off, otherwise IPG_PERCLK is always active. 0b - IPG_PERCLK is active. 1b - IPG_PERCLK is gated off.
5 HCKOFF	HCLK gated off internally This status field indicates that the HCLK is internally gated off. This field is for the host driver to debug during a data transfer. 0b - HCLK is active. 1b - HCLK is gated off.
4 IPGOFF	Peripheral clock gated off internally This status field indicates that the peripheral clock is internally gated off. This field is for the host driver to debug. 0b - Peripheral clock is active. 1b - Peripheral clock is gated off.
3 SDSTB	SD clock stable This status field indicates that the internal card clock is stable. This field is for the host driver to poll clock status when changing the clock frequency. It is recommended to clear FRC_SDCLK_ON field in System Control register to remove glitches on the card clock when the frequency is changing. Before changing clock divisor value (SDCLKFS or DVS), host driver should make sure the SDSTB field is high. 0b - Clock is changing frequency and not stable. 1b - Clock is stable.
2 DLA	Data line active This status field indicates whether one of the DATA lines on the SD bus is in use.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	In the case of read transactions: This status indicates if a read transfer is executing on the SD bus. Changes in this value from 1 to 0, between data blocks, generates a Block Gap Event interrupt in the Interrupt Status register. This field is set in either of the following cases: • After the end field of the read command • When writing a 1 to the Continue Request field in the Protocol Control register to restart a read transfer This field is cleared in either of the following cases: • When the end field of the last data block is sent from the SD bus to uSDHC. • When the Read Wait state is stopped by a Suspend command and the DATA2 line is released. The uSDHC module waits at the next block gap by driving Read Wait at the start of the interrupt cycle. If the Read Wait signal is already driven (data buffer cannot receive data), uSDHC can wait for a current block gap by continuing to drive the Read Wait signal. It is necessary to support Read Wait to use the suspend / resume function. This field remains 1 during Read Wait. In the case of write transactions: This status indicates that a write transfer is executing on the SD bus. Changes in this value from 1 to 0 generate a Transfer Complete interrupt in the Interrupt Status register. This field is set in either of the following cases: • After the end field of the write command • When writing to 1 to the Continue Request field in the Protocol Control register to continue a write transfer This field is cleared in either of the following cases: • When the SD card releases Write Busy of the last data block, uSDHC also detects if the output is not busy. If the SD card does not drive the busy signal after the CRC status is received, uSDHC assumes the card drive "Not Busy". • When the SD card releases write busy, prior to waiting for write transfer, and because of a Stop At Block Gap Request. In the case of command with busy pending: This status indicates that a busy state follows the command and the data line is in use. This field is cleared when the DATA0 line is released. 0b - DATA line inactive 1b - DATA line active
1 CDIHB	Command Inhibit Data (DATA) This status field is generated if either the DAT Line Active or the Read Transfer Active is set to 1. If this field is 0, it indicates that uSDHC can issue the next SD / MMC Command. Commands with a busy signal

Table continues on the next page...

Table continued from the previous page...

Field	Function
	belong to Command Inhibit (DATA) (for example. R1b, R5b type). Changing from 1 to 0 generates a Transfer Complete interrupt in the Interrupt Status register. **NOTE** The SD host driver can save registers for a suspend transaction after this field has changed from 1 to 0. 0b - Can issue command that uses the DATA line 1b - Cannot issue command that uses the DATA line
0 CIHB	Command inhibit (CMD) If this status bit is 0, it indicates that the CMD line is not in use and uSDHC can issue a SD / MMC command using the CMD line. This field is set also immediately after the Transfer Type register is written. This field is cleared when the command response is received. Even if the Command Inhibit (DATA) is set to 1, commands using only the CMD line can be issued if this field is 0. Changing from 1 to 0 generates a Command Complete interrupt in the Interrupt Status register. If uSDHC cannot issue the command because of a command conflict error (see Command CRC Error) or because of a Command Not Issued By Auto CMD12 Error, this field remains 1 and the Command Complete is not set. The Status of issuing an auto CMD12 does not show on this field. 0b - Can issue command using only CMD line 1b - Cannot issue command

39.4.1.12 Protocol Control (PROT_CTRL)

Offset

Register	Offset
PROT_CTRL	28h

Function

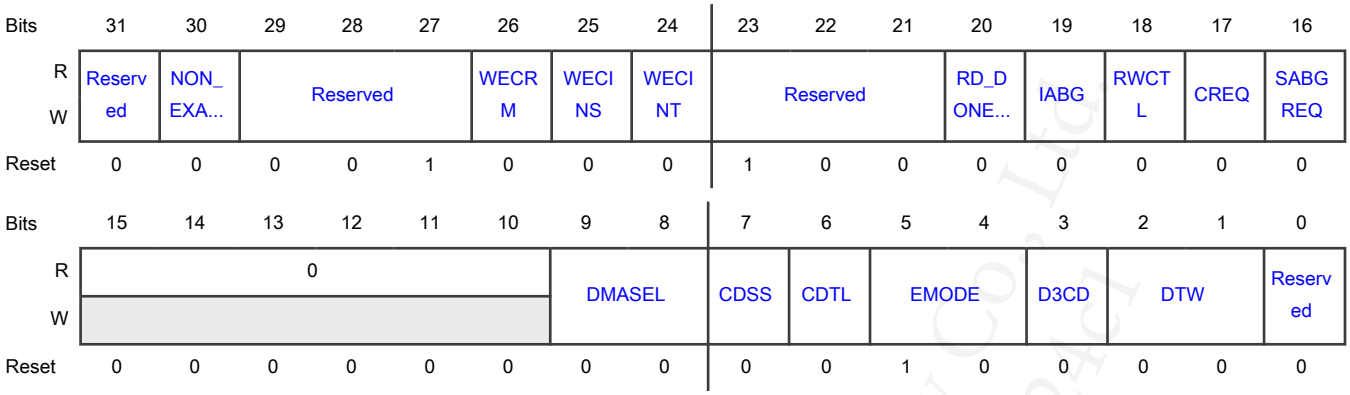
This register controls three cases to restart the transfer after stop at the block gap. Which case is appropriate depends on whether uSDHC issues a Suspend command or the SD card accepts the Suspend command.

- If the host driver does not issue a Suspend command, the Continue request is used to restart the transfer.
- If the host driver issues a Suspend command and the SD card accepts it, a Resume command is used to restart the transfer.
- If the host driver issues a Suspend command and the SD card does not accept it, the Continue request is used to restart the transfer.

Any time stop at block gap request stops the data transfer, the host driver waits for a Transfer Complete (in the Interrupt Status register), before attempting to restart the transfer. When restarting the data transfer by Continue Request, the host driver clears the Stop At Block Gap Request before or simultaneously.

Provided under NDA
COMPANY PROPRIETARY
Beijing CTI Automotive Technology Co. Ltd.
c8e62e54-21a8-4a8-b251-3...a624c1

Diagram



Fields

Field	Function
31 —	Reserved Always write as 0
30 NON_EXACT_BLK_RD	Non-exact block read Current block read is non-exact block read. It is only used for SDIO. 0b - The block read is exact block read. Host driver does not need to issue abort command to terminate this multi-block read. 1b - The block read is non-exact block read. Host driver needs to issue abort command to terminate this multi-block read.
29-27 —	Reserved
26 WECRM	Wakeup event enable on SD card removal This field enables a wakeup event, via a card removal, in the Interrupt Status register. FN_WUS (Wakeup Support) in CIS does not affect this field. When this field is set, the Card Removal Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Removal Status and uSDHC interrupt. 0b - Disables wakeup event enable on SD card removal 1b - Enables wakeup event enable on SD card removal
25 WECINS	Wakeup event enable on SD card insertion This field enables a wakeup event, via a card insertion, in the Interrupt Status register. FN_WUS (Wakeup Support) in CIS does not affect this field. When this field is set, the Card Insertion Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Insertion Status and uSDHC interrupt. 0b - Disable wakeup event enable on SD card insertion 1b - Enable wakeup event enable on SD card insertion

Table continues on the next page...

Table continued from the previous page...

Field	Function
24 WECINT	Wakeup event enable on card interrupt This field enables a wakeup event, via a card interrupt, in the Interrupt Status register. This field can be set to 1 if FN_WUS (Wakeup Support) in CIS is set to 1. When this field is set, the Card Interrupt Status and uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active to assert the Card Interrupt Status and uSDHC interrupt. 0b - Disables wakeup event enable on card interrupt 1b - Enables wakeup event enable on card interrupt
23-21 —	Reserved Always write as 3'b100
20 RD_DONE_NO_8CLK	Read performed number 8 clock According to the SD/MMC spec, for read data transaction, 8 clocks are needed after the end field of the last data block. So, by default(RD_DONE_NO_8CLK=0), eight clocks are active after the end field of the last read data transaction. However, these 8 clocks should not be active if user wants to use stop at block gap (include the auto stop at block gap in boot mode) feature for read and the RWCTL field (bit18) is not enabled. In this case, software should set RD_DONE_NO_8CLK to avoid these 8 clocks. Otherwise, the device might send extra data to uSDHC while uSDHC ignores these data. In a summary, this field should be set only if the use case needs to use stop at block gap feature while the device can't support the read wait feature.
19 IABG	Interrupt at block gap This field is valid only in 4-bit mode, of the SDIO card, and selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multiple block transfer. Setting to 0 disables interrupt detection during a multiple block transfer. If the SDIO card cannot signal an interrupt during a multiple block transfer, this field should be set to 0 to avoid an inadvertent interrupt. When the host driver detects an SDIO card insertion, it sets this field according to the CCCR of the card. 0b - Disables interrupt at block gap 1b - Enables interrupt at block gap
18 RWCTL	Read wait control The read wait function provided by this field is optional for SDIO cards. If the card supports read wait, set this field to enable use of the read wait protocol to stop read data using the DATA2 line. Otherwise, uSDHC has to stop the SD clock to hold read data, which restricts commands generation. When the host driver detects an SDIO card insertion, it sets this field according to the CCCR of the card. If the card does not support read wait, this field should never be set to 1; otherwise, DATA line conflicts might occur. If this field is set to 0, stop at block gap during read operation is also supported, but uSDHC stops the SD clock to pause reading operation. 0b - Disables read wait control and stop SD clock at block gap when SABGREQ field is set 1b - Enables read wait control and assert read wait without stopping SD clock at block gap when SABGREQ field is set

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 CREQ	Continue request This field is used to restart a transaction which was stopped using the stop at block gap request. When a suspend operation is not accepted by the card, it is also by setting this field to restart the paused transfer. To cancel stop at the block gap, set stop at block gap request to 0 and set this field to 1 to restart the transfer. The uSDHC module automatically clears this field, therefore it is not necessary for the host driver to set this field to 0. If both stop at block gap request and this field are 1, the continue request is ignored. 0b - No effect 1b - Restart
16 SABGREQ	Stop at block gap request This field is used to stop executing a transaction at the next block gap for both DMA and non-DMA transfers. Until the transfer complete is set to 1, indicating a transfer completion, the host driver leaves this field set to 1. Clearing both the stop at block gap request and continue request does not cause the transaction to restart. Read Wait is used to stop the read transaction at the block gap. The uSDHC module supports the stop at block gap request for write transfers, but for read transfers it requires that the SDIO card support read wait. Therefore, the host driver does not set this field during read transfers unless the SDIO card supports Read Wait and has set the read wait control to 1; otherwise, uSDHC stops the SD bus clock to pause the read operation during block gap. In the case of write transfers in which the host driver writes data to the Data Port register, the host driver sets this field after all block data is written. If this field is set to 1, the host driver does not write data to the Data Port register after a block is sent. Once this field is set, the host driver does not clear this field before the Transfer Complete field in Interrupt Status register is set, otherwise uSDHC's behavior is undefined. This field effects read transfer active, write transfer active, DATA Line Active and Command Inhibit (DATA) in the Present State register. 0b - Transfer 1b - Stop
15-10 —	Reserved
9-8 DMASEL	DMA select This field is valid while DMA (SDMA or ADMA) is enabled and selects the DMA operation. 00b - No DMA or simple DMA is selected. 01b - ADMA1 is selected. 10b - ADMA2 is selected. 11b - Reserved
7 CDSS	Card detect signal selection This field selects the source for the card detection. 0b - Card detection level is selected (for normal purpose). 1b - Card detection test level is selected (for test purpose).

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 CDTL	Card detect test level This field is enabled while the card detection signal selection is set to 1 and it indicates card insertion. 0b - Card detect test level is 0, no card inserted 1b - Card detect test level is 1, card inserted
5-4 EMODE	Endian mode This field supports all three endian modes in data transfer. See Data buffer for more details. 00b - Big endian mode 01b - Half word big endian mode 10b - Little endian mode 11b - Reserved
3 D3CD	DATA3 as card detection pin If this field is set, DATA3 should be pulled down to act as a card detection pin. Be cautious when using this feature, because DATA3 is also a chip-select for the SPI mode. A pull-down on this pin and CMD0 might set the card into the SPI mode, which uSDHC does not support. 0b - DATA3 does not monitor card insertion 1b - DATA3 as card detection pin
2-1 DTW	Data transfer width This field selects the data width of the SD bus for a data transfer. The host driver sets it to match the data width of the card. Possible data transfer width is 1-bit, 4-bits or 8-bits. 00b - 1-bit mode 01b - 4-bit mode 10b - 8-bit mode 11b - Reserved
0 —	Reserved

39.4.1.13 System Control (SYS_CTRL)

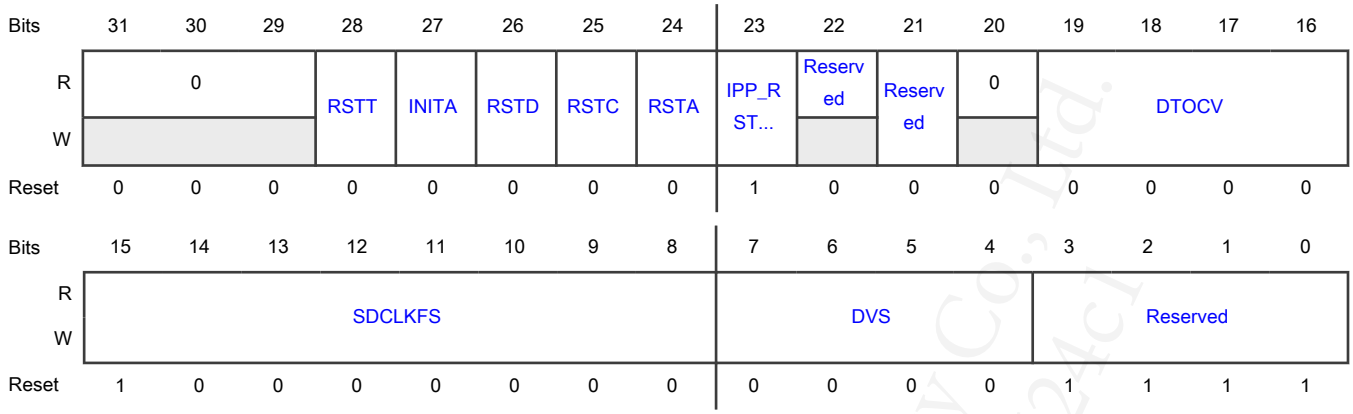
Offset

Register	Offset
SYS_CTRL	2Ch

Function

This register provides control of the system. See detail in the field description.

Diagram



Fields

Field	Function
31-29 —	Reserved
28 RSTT	Reset tuning When set this field to 1, it resets tuning circuit. After tuning circuits are reset, bit value is 0. Clearing execute_tuning field in AUTOCMD12_ERR_STATUS also sets this field to 1 to reset tuning circuit
27 INITA	Initialization active When this field is set, 80 SD-clocks are sent to the card. After the 80 clocks are sent, this field is self cleared. This field is very useful during the card power-up period when 74 SD-clocks are needed and the clock auto gating feature is enabled. Writing 1 to this bit when this field is already 1 has no effect. Writing 0 to this field at any time has no effect. When either of the CIHB and CDIHB fields in the Present State register are set, writing 1 to this field is ignored (that is, when command line or data lines are active, write to this field is not allowed). On the other-hand, when this field is set, that is, during initialization active period, it is allowed to issue command, and the command bit stream appears on the CMD pad after all 80 clock cycles are done. So, when this command ends, the driver can make sure the 80 clock cycles are sent out. This is very useful when the driver needs to send 80 cycles to the card and does not want to wait till this field is self cleared.
26 RSTD	Software reset for data line Only part of the data circuit is reset. DMA circuit is also reset. After this field is set, the software waits for self-clear. The following registers and bits are cleared by this field: • Data Port register — Buffer is cleared and initialized • Present State register — Buffer read enable — Buffer write enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	— Read transfer active — Write transfer active — DATA line active — Command Inhibit (DATA) • Protocol Control register — Continue request • Interrupt Status register — Buffer read ready — Buffer write ready — DMA interrupt — Block gap event — Transfer complete **NOTE** When reset, the software must make sure there is no incomplete data transferring. If there is data transfer going on, the software needs to wait TC or DC INT_STATUS register is set. 0b - No reset 1b - Reset
25 RSTC	Software reset for CMD line Only part of the command circuit is reset. After this field is set, the software waits for self-clear. The following registers and bits are cleared by this field: • Present State Register — Command Inhibit (CMD) • Interrupt Status Register — Command Complete 0b - No reset 1b - Reset
24 RSTA	Software reset for all This reset affects the entire host controller except for the card detection circuit. RSTA resets all the registers that can be reset by RSTC/RSTD. During its initialization, the host driver is set this field to 1 to reset uSDHC. The uSDHC module resets this field to 0 when the capabilities registers are valid and the host driver can read them. Additional use of Software Reset For All does not affect the value of the capabilities registers. After this field is set, it is recommended that the host driver reset the external card and re-initialize it. After this field is set, the software should wait for self-clear.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	In tuning process, after every CMD19 is finished, this field is set to retest uSDHC. **NOTE** When reset, the software must make sure there is no incomplete data transferring. If there is data transfer going on, the software needs to wait TC or DC INT_STATUS register is set. 0b - No reset 1b - Reset
23 IPP_RST_N	Hardware reset This field's value is output to card through pad directly to hardware reset pin of the card if the card supports this feature.
22 —	Reserved
21 —	Reserved
20 —	Reserved
19-16 DTOCV	Data timeout counter value This value determines the interval by which DAT line timeouts are detected. See the Data Timeout Error field in the Interrupt Status register for information on factors that dictate time-out generation. Time-out clock frequency is generated by dividing the base clock SDCLK value by this value. The host driver can clear the Data Timeout Error Status Enable (in the Interrupt Status Enable register) to prevent inadvertent time-out events. 0000b - SDCLK x 2^{32} 0001b - SDCLK x 2^{33} 0010b - SDCLK x 2^{18} 0011b - SDCLK x 2^{19} 1101b - SDCLK x 2^{29} recommend to use for other speed mode except HS400/HS200/SDR104 mode 1110b - SDCLK x 2^{30} recommend to use for HS200/SDR104 mode 1111b - SDCLK x 2^{31} recommend to use for HS400 mode
15-8 SDCLKFS	SDCLK frequency select This field is used to select the frequency of the SDCLK pin. The frequency is not programmed directly, rather this field holds the prescaler (of this register) and divisor (next register) of the Base Clock Frequency register. In Single Data Rate mode (DDR_EN field of MIXERCTRL is '0')

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Only the following settings are allowed: 80h) Base clock divided by 256 40h) Base clock divided by 128 20h) Base clock divided by 64 10h) Base clock divided by 32 08h) Base clock divided by 16 04h) Base clock divided by 8 02h) Base clock divided by 4 01h) Base clock divided by 2 00h) Base clock divided by 1 While in Dual Data Rate mode (DDR_EN field of MIXERCTRL is '1') Only the following settings are allowed: 80h) Base clock divided by 512 40h) Base clock divided by 256 20h) Base clock divided by 128 10h) Base clock divided by 64 08h) Base clock divided by 32 04h) Base clock divided by 16 02h) Base clock divided by 8 01h) Base clock divided by 4 00h) Base clock divided by 2 When the software changes the DDR_EN field, SDCLKFS might need to be changed also. In Single Data Rate mode, setting 00h bypasses the frequency prescaler of the SD clock. Multiple bits must not be set, or the behavior of this prescaler is undefined. The two default divider values can be calculated by the frequency of ipg_perclk and the following Divisor bits. The frequency of SDCLK is set by the following formula: Clock Frequency = (Base Clock) / (prescaler x divisor) For example, in Single Data Rate mode, if the Base Clock Frequency is 96 MHz, and the target frequency is 25 MHz, then choosing the prescaler value of 01h and divisor value of 1h yields 24 MHz, which is the nearest frequency less than or equal to the target. Similarly, to approach a clock value of 400 kHz, the prescaler value of 08h and divisor value of eh yields the exact clock value of 400 kHz. The reset value of this field is 80h, so if the input Base Clock (ipg_perclk) is about 96 MHz, the default SD clock after reset is 375 kHz. According to the SD Physical Specification Version 1.1 and the SDIO Card Specification Version 1.2, the maximum SD clock frequency is 50 MHz and is never exceed this limit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Before changing clock divisor value (SDCLKFS or DVS), host driver should make sure the SDSTB field is high. If setting SDCLKFS and DVS can generate the same clock frequency,(for example, in SDR mode, SDCLKFS = 01h is same as DVS = 01h.), SDCLKFS is highly recommended.
7-4 DVS	Divisor This field is used to provide a more exact divisor to generate the desired SD clock frequency. Note the divider can even support odd divisors without deterioration of duty cycle. Before changing clock divisor value (SDCLKFS or DVS), Host driver should make sure the SDSTB field is high. The settings are as follows: 0000b - Divide-by-1 0001b - Divide-by-2 1110b - Divide-by-15 1111b - Divide-by-16
3-0 —	Reserved Always write as 1.

39.4.1.14 Interrupt Status (INT_STATUS)

Offset

Register	Offset
INT_STATUS	30h

Function

An interrupt is generated when the normal interrupt signal enable is enabled and at least one of the status fields is set to 1. For all fields, writing 1 to a bit clears it; writing to 0 keeps the bit unchanged. More than one status can be cleared with a single register write. For card interrupt, before writing 1 to clear, it is required that the card stops asserting the interrupt, meaning that when the card driver services the interrupt condition; otherwise, the CINT field is asserted again.

The table below shows the relationship between the command timeout error and the command complete.

Table 256. uSDHC status for command timeout error/command complete bit combinations

Command CRC Error	Command timeout error	Meaning of the status
0	0	X
X	1	Response not received within 64 SDCLK cycles
1	0	Response received

The table below shows the relationship between the transfer complete and the data timeout error.

Table 257. uSDHC status for data timeout error/transfer complete bit combinations

Transfer complete	Data timeout error	Meaning of the status
0	0	X
0	1	Timeout occurred during transfer
1	X	Data transfer complete

The table below shows the relationship between the command CRC error and command timeout error.

Table 258. uSDHC status for command CRC error/command timeout error bit combinations

Command complete	Command timeout error	Meaning of the status
0	0	No error
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			DMAE	0	TNE	0	AC12E	0	DEBE	DCE	DTOE	CIE	CEBE	CCE	CTOE
W				W1C		W1C		W1C		W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CQI	TP	RTE	0			CINT	CRM	CINS	BRR	BWR	DINT	BGE	TC	CC
W		W1C	W1C	W1C				W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMAE	DMA error Occurs when an Internal DMA transfer has failed. This field is set to 1, when some error occurs in the data transfer. This error can be caused by either simple DMA or ADMA, depending on which DMA is in use. The value in DMA System Address register is the next fetch address where the error occurs. Because any

Table continues on the next page...

Table continued from the previous page...

Field	Function
	error corrupts the whole data block, the host driver restarts the transfer from the corrupted block boundary. The address of the block boundary can be calculated either from the current DS_ADDR value or from the remaining number of blocks and the block size. 0b - No error 1b - Error
27 —	Reserved
26 TNE	Tuning error: (only for SD3.0 SDR104 mode and EMMC HS200 mode) This field is set when an unrecoverable error is detected in a tuning circuit. By detecting Tuning Error, host driver needs to abort a command executing and perform tuning.
25 —	Reserved
24 AC12E	Auto CMD12 error Occurs when detecting that one of the fields in the Auto CMD12 Error Status register has changed from 0 to 1. This field is set to 1, not only when the errors in Auto CMD12 occur, but also, when the Auto CMD12 is not executed due to the previous command error. 0b - No error 1b - Error
23 —	Reserved
22 DEBE	Data end bit error Occurs either when detecting 0 at the end field position of read data that uses the DATA line, or at the end field position of the CRC. This field is not asserted in tuning process. 0b - No error 1b - Error
21 DCE	Data CRC error Occurs when detecting a CRC error when transferring read data that uses the DATA line, or when detecting the Write CRC status having a value other than 010. This field is not asserted in tuning process. 0b - No error 1b - Error
20	Data timeout error

Table continues on the next page...

Table continued from the previous page...

Field	Function
DTOE	Occurs when detecting one of following time-out conditions. • Busy time-out for R1b, R5b type • Busy time-out after Write CRC status • Read Data time-out. This field is not asserted in tuning process. 0b - No error 1b - Time out
19 CIE	Command index error Occurs if a command index error occurs in the command response. This field is not asserted in tuning process. 0b - No error 1b - Error
18 CEBE	Command end bit error Occurs when detecting that the end field of a command response is 0. This field is not asserted in tuning process. 0b - No error 1b - End bit error generated
17 CCE	Command CRC error Command CRC Error is generated in two cases. • If a response is returned and the Command Timeout Error is set to 0 (indicating no time-out), this field is set when detecting a CRC error in the command response. • The uSDHC module detects a CMD line conflict by monitoring the CMD line when a command is issued. If uSDHC drives the CMD line to 1, but detects 0 on the CMD line at the next SDCLK edge, then uSDHC aborts the command (Stop driving CMD line) and set this bit to 1. The Command Timeout Error should also be set to 1 to distinguish CMD line conflict. This field is not asserted in tuning process. 0b - No error 1b - CRC error generated
16 CTOE	Command timeout error Occurs only if no response is returned within 64 SDCLK cycles from the end field of the command. If uSDHC detects a CMD line conflict, in which case a Command CRC Error is also be set (as shown in Interrupt Status (INT_STATUS)), this field is set without waiting for 64 SDCLK cycles. This is because the command is aborted by uSDHC. This field is not asserted in tuning process.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No error 1b - Time out
15 —	Reserved
14 CQI	Command queuing interrupt This interrupt is asserted when at least one of the bits in CQIS register is set. This interrupt is cleared only by clearing the source interrupt in CQIS register.
13 TP	Tuning pass:(only for SD3.0 SDR104 mode and EMMC HS200 mode) Current CMD19 transfer is done successfully. That is, current sampling point is correct.
12 RTE	Re-tuning event: (only for SD3.0 SDR104 mode and EMMC HS200 mode) This status is set if Re-Tuning Request in the Present State register changes from 0 to 1. host controller requests host driver to perform re-tuning for next data transfer. Current data transfer (not large block count) can be completed without re-tuning. 0b - Re-tuning is not required. 1b - Re-tuning should be performed.
11-9 —	Reserved
8 CINT	Card interrupt This status field is set when an interrupt signal is detected from the external card. In 1-bit mode, uSDHC detects the Card Interrupt without the SD clock to support wakeup. In 4-bit mode, the card interrupt signal is sampled during the interrupt cycle, so the interrupt from card can only be sampled during interrupt cycle, introducing some delay between the interrupt signal from the SDIO card and the interrupt to the host system. Writing this field to 1 can clear this field, but as the interrupt source from the SDIO card does not clear, this field is set again. to clear this field, it is required to reset the interrupt source from the external card followed by a writing 1 to this field. When this status has been set, and the host driver needs to service this interrupt, the Card Interrupt Signal Enable in the Interrupt Signal Enable register should be 0 to stop driving the interrupt signal to the host system. After completion of the card interrupt service (It should reset the interrupt sources in the SDIO card and the interrupt signal might not be asserted), write 1 to clear this field, set the Card Interrupt Signal Enable to 1, and start sampling the interrupt signal again. 0b - No card interrupt 1b - Generate card interrupt
7 CRM	Card removal This status field is set if the Card Inserted field in the Present State register changes from 1 to 0. When the host driver writes this field to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state might possibly be changed when the host driver clears this field

Table continues on the next page...

Table continued from the previous page...

Field	Function
	and the interrupt event might not be generated. When this field is cleared, it is set again if no card is inserted. to leave it cleared, clear the Card Removal Status Enable field in Interrupt Status Enable register. 0b - Card state unstable or inserted 1b - Card removed
6 CINS	Card insertion This status field is set if the Card Inserted field in the Present State register changes from 0 to 1. When the host driver writes this field to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state might possibly be changed when the host driver clears this field and the interrupt event might not be generated. When this field is cleared, it is set again if a card is inserted. to leave it cleared, clear the Card Inserted Status Enable field in Interrupt Status Enable register. 0b - Card state unstable or removed 1b - Card inserted
5 BRR	Buffer read ready This status field is set if the Buffer Read Enable field, in the Present State register, changes from 0 to 1. See the Buffer Read Enable field in the Present State register for additional information. This field indicates that cmd19 is finished in tuning process. 0b - Not ready to read buffer 1b - Ready to read buffer
4 BWR	Buffer write ready This status field is set if the Buffer Write Enable field, in the Present State register, changes from 0 to 1. See the Buffer Write Enable field in the Present State register for additional information. 0b - Not ready to write buffer 1b - Ready to write buffer
3 DINT	DMA interrupt Occurs only when the internal DMA finishes the data transfer successfully. Whenever errors occur during data transfer, this field does not be set. Instead, the DMAE field is set. Either Simple DMA or ADMA finishes data transferring, this field is set. 0b - No DMA interrupt 1b - DMA interrupt is generated.
2 BGE	Block gap event If the Stop At Block Gap Request field in the Protocol Control register is set, this field is set when a read or write transaction is stopped at a block gap. If Stop At Block Gap Request is not set to 1, this field is not set to 1. In the case of a Read Transaction: This field is set at the falling edge of the DATA Line Active Status (When the transaction is stopped at SD bus timing). The Read Wait must be supported to use this function.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	In the case of Write Transaction: This field is set at the falling edge of Write Transfer Active Status (After getting CRC status at SD Bus timing). 0b - No block gap event 1b - Transaction stopped at block gap
1 TC	Transfer complete This field is set when a read or write transfer is completed. In the case of a Read Transaction: This field is set at the falling edge of the Read Transfer Active Status. There are two cases in which this interrupt is generated. The first is when a data transfer is completed as specified by the data length (after the last data has been read to the host system). The second is when data has stopped at the block gap and completed the data transfer by setting the Stop At Block Gap Request field in the Protocol Control register (after valid data has been read to the host system). In the case of a Write Transaction: This field is set at the falling edge of the DATA Line Active Status. There are two cases in which this interrupt is generated. The first is when the last data is written to the SD card as specified by the data length and the busy signal is released. The second is when data transfers are stopped at the block gap, by setting the Stop At Block Gap Request field in the Protocol Control register, and the data transfers are completed. (after valid data is written to the SD card and the busy signal released). In the case of a command with busy, this field is set when busy is deasserted. This field is not asserted in tuning process. 0b - Transfer does not complete 1b - Transfer complete
0 CC	Command complete This field is set when you receive the end field of the command response (except auto CMD12). See the Command Inhibit (CMD) in the Present State register. This field is not asserted in tuning process. 0b - Command not complete 1b - Command complete

39.4.1.15 Interrupt Status Enable (INT_STATUS_EN)

Offset

Register	Offset
INT_STATUS_EN	34h

Function

Setting any bit in this register to 1 enables the corresponding interrupt status be requested to the system. If any bit is set to 0, the corresponding interrupt request gets blocked.

- Depending on IABG field setting, uSDHC might be programmed to sample the card interrupt signal during the interrupt period and hold its value in the flip-flop. There are some delays on the Card Interrupt, asserted from the card, to the time the host system is informed.
- To detect a CMD line conflict, the host driver must set both Command Timeout Error Status Enable and Command CRC Error Status Enable to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			DMAESEN	0	TNESEN	0	AC12ESEN	0	DEBESEN	DCESEN	DTOESEN	CIESEN	CEBESEN	CCESEN	CTOESEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CQISEN	TPSEN	RTESEN	0			CINTSEN	CRMSEN	CINSSEN	BRRSEN	BWRSEN	DINTSEN	BGESEN	TCSEN	CCSEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMAESEN	DMA error status enable 0b - Masked 1b - Enabled
27 —	Reserved
26 TNESEN	Tuning error status enable 0b - Masked 1b - Enabled
25 —	Reserved
24 AC12ESEN	Auto CMD12 error status enable 0b - Masked 1b - Enabled
23 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 DEBESEN	Data end bit error status enable 0b - Masked 1b - Enabled
21 DCESEN	Data CRC error status enable 0b - Masked 1b - Enabled
20 DTOESEN	Data timeout error status enable 0b - Masked 1b - Enabled
19 CIESEN	Command index error status enable 0b - Masked 1b - Enabled
18 CEBESEN	Command end bit error status enable 0b - Masked 1b - Enabled
17 CCESEN	Command CRC error status enable 0b - Masked 1b - Enabled
16 CTOESEN	Command timeout error status enable 0b - Masked 1b - Enabled
15 —	Reserved
14 CQISEN	Command queuing status enable 0b - Masked 1b - Enabled
13 TPSEN	Tuning pass status enable 0b - Masked 1b - Enabled
12 RTESEN	Re-tuning event status enable 0b - Masked 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-9 —	Reserved
8 CINTSEN	Card interrupt status enable If this field is set to 0, uSDHC clears the interrupt request to the system. The Card Interrupt detection is stopped when this field is cleared and restarted when this field is set to 1. The host driver should clear the Card Interrupt Status Enable before servicing the Card Interrupt and should set this field again after all interrupt requests from the card are cleared to prevent inadvertent interrupts. 0b - Masked 1b - Enabled
7 CRMSEN	Card removal status enable 0b - Masked 1b - Enabled
6 CINSSEN	Card insertion status enable 0b - Masked 1b - Enabled
5 BRRSEN	Buffer read ready status enable 0b - Masked 1b - Enabled
4 BWRSEN	Buffer write ready status enable 0b - Masked 1b - Enabled
3 DINTSEN	DMA interrupt status enable 0b - Masked 1b - Enabled
2 BGESEN	Block gap event status enable 0b - Masked 1b - Enabled
1 TCSEN	Transfer complete status enable 0b - Masked 1b - Enabled
0 CCSEN	Command complete status enable 0b - Masked

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled

39.4.1.16 Interrupt Signal Enable (INT_SIGNAL_EN)

Offset

Register	Offset
INT_SIGNAL_EN	38h

Function

This register is used to select which interrupt status is indicated to the host system as the interrupt. These status fields all share the same interrupt lines. Setting any of these fields to 1 enables interrupt generation. The corresponding Status register field generates an interrupt when the corresponding interrupt signal enable field is set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			DMAEIEN	0	TNEIEIN	0	AC12EIEN	0	DEBEIEN	DCEIEN	DTOEIEN	CIEIEIN	CEBEIEN	CCEIEIN	CTOEIEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CQIIEN	TPIEN	RTEIEN	0			CINTIEN	CRMIEN	CINSIEN	BRRIEN	BWRIEN	DINTIEN	BGEIEN	TCIEN	CCIEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28 DMAEIEN	DMA error interrupt enable 0b - Masked 1b - Enable
27 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 TNEIEN	Tuning error interrupt enable 0b - Masked 1b - Enabled
25 —	Reserved
24 AC12EIEN	Auto CMD12 error interrupt enable 0b - Masked 1b - Enabled
23 —	Reserved
22 DEBEIEN	Data end bit error interrupt enable 0b - Masked 1b - Enabled
21 DCEIEN	Data CRC error interrupt enable 0b - Masked 1b - Enabled
20 DTOEIEN	Data timeout error interrupt enable 0b - Masked 1b - Enabled
19 CIEIEN	Command index error interrupt enable 0b - Masked 1b - Enabled
18 CEBEIEN	Command end bit error interrupt enable 0b - Masked 1b - Enabled
17 CCEIEN	Command CRC error interrupt enable 0b - Masked 1b - Enabled
16 CTOEIEN	Command timeout error interrupt enable 0b - Masked 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 —	Reserved
14 CQIIEN	Command queuing signal enable 0b - Masked 1b - Enabled
13 TPIEN	Tuning pass interrupt enable 0b - Masked 1b - Enabled
12 RTEIEN	Re-tuning event interrupt enable 0b - Masked 1b - Enabled
11-9 —	Reserved
8 CINTIEN	Card interrupt enable 0b - Masked 1b - Enabled
7 CRMIEN	Card removal interrupt enable 0b - Masked 1b - Enabled
6 CINSIEN	Card insertion interrupt enable 0b - Masked 1b - Enabled
5 BRRIEN	Buffer read ready interrupt enable 0b - Masked 1b - Enabled
4 BWRIEN	Buffer write ready interrupt enable 0b - Masked 1b - Enabled
3 DINTIEN	DMA interrupt enable 0b - Masked 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 BGEIEN	Block gap event interrupt enable 0b - Masked 1b - Enabled
1 TCIEN	Transfer complete interrupt enable 0b - Masked 1b - Enabled
0 CCIEN	Command complete interrupt enable 0b - Masked 1b - Enabled

39.4.1.17 Auto CMD12 Error Status (AUTOCMD12_ERR_STATUS)

Offset

Register	Offset
AUTOCMD12_ERR_STATUS	3Ch

Function

When the Auto CMD12 Error Status field in the Status register is set, the host driver checks this register to identify what kind of error the Auto CMD12 / CMD 23 indicated. Auto CMD23 errors are indicated in field 04-01. This register is valid only when the Auto CMD12 Error status field is set.

The table below shows the relationship between the Auto CMGD12 CRC Error and the Auto CMD12 Command Timeout Error.

Table 259. Relationship between command CRC error and command timeout error for auto CMD12

Auto CMD12 CRC error	Auto CMD12 timeout error	Type of error
0	0	No error
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

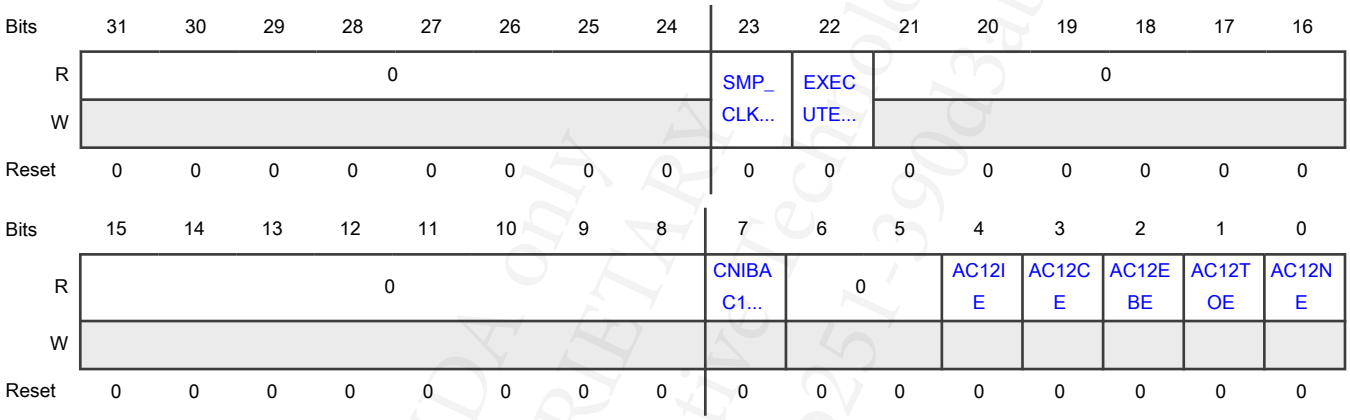
Changes in Auto CMD12 Error Status register can be classified in three scenarios:

- When uSDHC is going to issue an Auto CMD12
  - Set field 0 to 1 if the Auto CMD12 cannot be issued due to an error in the previous command
  - Set field 0 to 0 if the Auto CMD12 is issued

- At the end field of an Auto CMD12 response
  - — Check errors correspond to fields 1-4
  - — Set fields 1-4 corresponding to detected errors
  - — Clear fields 1-4 corresponding to detected errors
- Before reading the Auto CMD12 Error Status field 7
  - — Set field 7 to 1 if there is a command that can't be issued
  - — Clear field 7 if there is no command to issue

The timing for generating the Auto CMD12 Error and writing to the Command register are asynchronous. After that, field 7 is sampled when the driver is not writing to the Command register. So, it is suggested to read this register only when the AC12E field in Interrupt Status register is set. An Auto CMD12 Error Interrupt is generated when one of the error fields (0-4) is set to 1. The Command Not Issued By Auto CMD12 Error does not generate an interrupt.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 SMP_CLK_SEL	Sample clock select This field is used to select sampling clock to receive CMD and DATA. This field is functional only when TUNING_CTRL[STD_TUNING_EN] is set. Otherwise, this field is reserved. This field is controlled by re-tuning procedure and is valid after the tuning procedure is complete (AUTOCMD12_ERR_STATUS[EXECUTE_TUNING]=0). A value of '1' means that tuning has been completed successfully and a value of '0' implies the tuning has failed. Writing '1' to this field is meaningless and ignored when the tuning has failed. The tuning circuit is reset by writing '0' to this field. This field can be cleared by setting AUTOCMD12_ERR_STATUS[EXECUTE_TUNING]=1. Once the tuning circuit is reset, it takes time to complete tuning sequence. Therefore, host driver should keep this field as 1 to perform the re-tuning sequence in a short time. This field cannot be changed while the Host controller is receiving response or a read data block. 0b - Fixed clock is used to sample data 1b - Tuned clock is used to sample data

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 EXECUTE_TUNING	Execute tuning This field is used to start the tuning procedure. This field is functional only when TUNING_CTRL[STD_TUNING_EN] is set. Otherwise, this field is reserved. This field is set to start tuning procedure and automatically cleared when tuning procedure is completed. The result of tuning is indicated to AUTOCMD12_ERR_STATUS[SMP_CLK_SEL]. 0b - Tuning procedure is aborted 1b - Start tuning procedure
21-8 —	Reserved
7 CNIBAC12E	Command not issued by Auto CMD12 error Setting this field to 1 means CMD_wo_DAT is not executed due to an Auto CMD12 error (D04-D01) in this register. 0b - No error 1b - Not issued
6-5 —	Reserved
4 AC12IE	Auto CMD12 / 23 index error Occurs if the command index error occurs in response to a command. 0b - No error 1b - Error, the CMD index in response is not CMD12/23
3 AC12CE	Auto CMD12 / 23 CRC error Occurs when detecting a CRC error in the command response. 0b - No CRC error 1b - CRC error met in Auto CMD12/23 response
2 AC12EBE	Auto CMD12 / 23 end bit error Occurs when detecting that the end field of command response is 0 which should be 1. 0b - No error 1b - End bit error generated
1 AC12TOE	Auto CMD12 / 23 timeout error Occurs if no response is returned within 64 SDCLK cycles from the end field of the command. If this field is set to1, the other error status fields (2-4) have no meaning. 0b - No error 1b - Time out

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 AC12NE	Auto CMD12 not executed If memory multiple block data transfer is not started, due to a command error, this field is not set because it is not necessary to issue an Auto CMD12. Setting this field to 1 means uSDHC cannot issue the Auto CMD12 to stop a memory multiple block data transfer due to some error. If this field is set to 1, other error status fields (1-4) have no meaning. 0b - Executed 1b - Not executed

39.4.1.18 Host Controller Capabilities (HOST_CTRL_CAP)

Offset

Register	Offset
HOST_CTRL_CAP	40h

Function

This register provides the host driver with information specific to uSDHC implementation. The value in this register is the power-on-reset value and does not change with a software reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					VS18	VS30	VS33	SRS	DMAS	HSS	ADMAS	0	MBL		
W																
Reset	0	0	0	0	0	1	1	1	1	1	1	1	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RETUNING_MODE		USE_TUN...	Reserved	TIME_COUNT_RETUNING				Reserved					DDR50_S...	SDR104_...	SDR50_S...
W																
Reset	1	0	1	1	0	1	0	0	0	0	0	0	0	1	1	1

Fields

Field	Function
31-27 —	Reserved
26	Voltage support 1.8 V

Table continues on the next page...

Table continued from the previous page...

Field	Function
VS18	This field depends on the host system ability. 0b - 1.8 V not supported 1b - 1.8 V supported
25 VS30	Voltage support 3.0 V This field depends on the host system ability. 0b - 3.0 V not supported 1b - 3.0 V supported
24 VS33	Voltage support 3.3 V This field depends on the host system ability. 0b - 3.3 V not supported 1b - 3.3 V supported
23 SRS	Suspend / resume support This field indicates whether uSDHC supports Suspend / Resume functionality. If this field is 0, the Suspend and Resume mechanism, as well as the read wait, are not supported, and the host driver does not issue either Suspend or Resume commands. 0b - Not supported 1b - Supported
22 DMAS	DMA support This field indicates whether uSDHC can use the internal DMA to transfer data between system memory and the data buffer directly. 0b - DMA not supported 1b - DMA supported
21 HSS	High speed support This field indicates whether uSDHC supports High Speed mode and the host system can supply a SD clock frequency from 25 MHz to 50 MHz. 0b - High speed not supported 1b - High speed supported
20 ADMAS	ADMA support This field indicates whether uSDHC supports the ADMA feature. 0b - Advanced DMA not supported 1b - Advanced DMA supported
19	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
18-16 MBL	Max block length This field indicates the maximum block size that the host driver can read and write to the buffer in uSDHC. The buffer transfers block size without wait cycles. 000b - 512 bytes 001b - 1024 bytes 010b - 2048 bytes 011b - 4096 bytes
15-14 RETUNING_MODE	Retuning Mode This field selects retuning method. 00b - Mode 1 01b - Mode 2 10b - Mode 3 11b - Reserved
13 USE_TUNING_SDR50	Use Tuning for SDR50 This field is used to enable tuning for SDR50 mode 0b - SDR50 does not support tuning 1b - SDR50 supports tuning
12 —	Reserved
11-8 TIME_COUNT_RETUNING	Time counter for retuning This field indicates an initial value of the Retuning Timer for Re-Tuning Mode1 and 3.Setting to 0 disables Retuning Timer.
7-3 —	Reserved
2 DDR50_SUPPORT	DDR50 support This field indicates support of DDR50 mode.
1 SDR104_SUPPORT	SDR104 support This field indicates support of SDR104 mode.
0	SDR50 support

Table continues on the next page...

Table continued from the previous page...

Field	Function
SDR50_SUPPORT	This field indicates support of SDR50 mode.

39.4.1.19 Watermark Level (WTMK_LVL)

Offset

Register	Offset
WTMK_LVL	44h

Function

This register indicates configurability of write and read watermark levels (FIFO threshold). Their value can range from 1 to 128 words.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			Reserved					WR_WML							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			Reserved					RD_WML							
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-29 —	Reserved
28-24 —	Reserved
23-16 WR_WML	Write watermark level This field indicates the number of words used as the watermark level (FIFO threshold) in a DMA write operation. Also, the number of words as a sequence of write bursts in back-to-back mode. The maximum

Table continues on the next page...

Table continued from the previous page...

Field	Function
	legal value for the write watermark level is 128.When using internal DMA, WR_WML can be set any value between 1 and 128. To get max AXI throughput, we recommend to use integer multiple of 16.
15-13 —	Reserved
12-8 —	Reserved
7-0 RD_WML	Read watermark level This field indicates the number of words used as the watermark level (FIFO threshold) in a DMA read operation. Also, the number of words as a sequence of read bursts in back-to-back mode. The maximum legal value for the read water mark level is 128.When using internal DMA mode, RD_WML must be set to the value less than 16. To get max AXI throughput, recommend to use 16.

39.4.1.20 Mixer Control (MIX_CTRL)

Offset

Register	Offset
MIX_CTRL	48h

Function

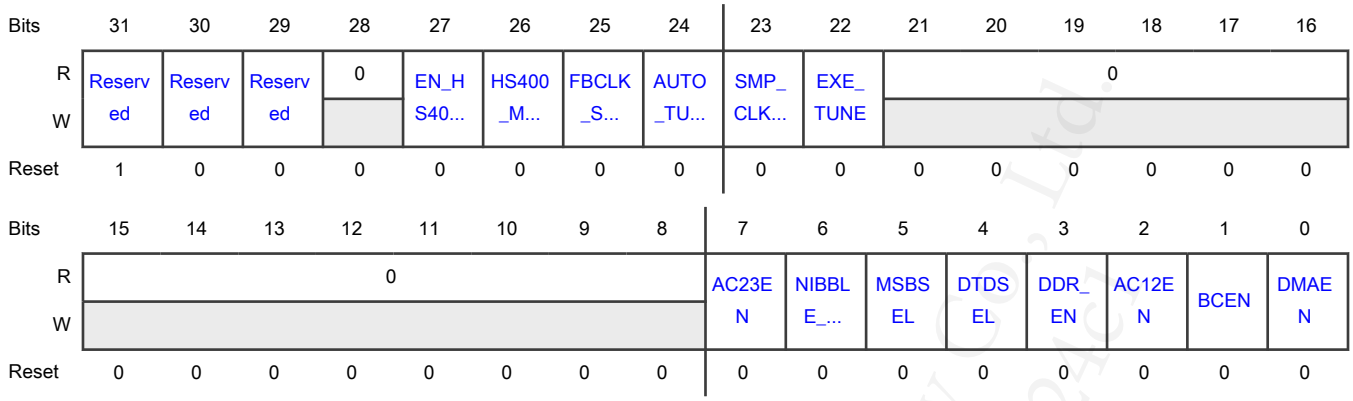
This register is used to DMA and data transfer. To prevent data loss, the software should check if data transfer is active before writing this register. These fields are DPSEL, MBSEL, DTDSEL, AC12EN, BCEN, and DMAEN.

Table 260. Transfer type register setting for various transfer types

Multi/single block select	Block count enable	Block count	Function
0	Do not care	Do not care	Single transfer
1	0	Do not care	Infinite transfer
1	1	Positive number	Multiple transfer
1	1	Zero	No data transfer

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a2-b251-390d3624c1

Diagram



Fields

Field	Function
31 —	Reserved Always write as 1.
30 —	Reserved Always write as 0.
29 —	Reserved Always write as 0.
28 —	Reserved
27 EN_HS400_MODE	Enable enhance HS400 mode Enhance HS400 enable
26 HS400_MODE	Enable HS400 mode HS400 Enable
25 FBCLK_SEL	Feedback clock source selection (Only used for SD3.0, SDR104 mode and EMMC HS200 mode) 0b - Feedback clock comes from the loopback CLK 1b - Feedback clock comes from the ipp_card_clk_out
24 AUTO_TUNE_EN	Auto tuning enable (Only used for SD3.0, SDR104 mode and and EMMC HS200 mode) 0b - Disable auto tuning 1b - Enable auto tuning
23 SMP_CLK_SEL	Clock selection When TUNING_CTRL[STD_TUNING_EN] is 0, this field is used to select Tuned clock or Fixed clock to sample data / cmd (Only used for SD3.0, SDR104 mode and EMMC HS200 mode)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Fixed clock is used to sample data / cmd 1b - Tuned clock is used to sample data / cmd
22 EXE_TUNE	Execute tuning: (Only used for SD3.0, SDR104 mode and EMMC HS200 mode) When TUNING_CTRL[STD_TUNING_EN] is 0, this field is set to 1 to indicate the host driver is starting tuning procedure. Tuning procedure is aborted by writing 0. 0b - Not tuned or tuning completed 1b - Execute tuning
21-8 —	Reserved
7 AC23EN	Auto CMD23 enable When this field is set to 1, the host controller issues a CMD23 automatically before issuing a command specified in the Command Register.
6 NIBBLE_POS	Nibble position indication This field indicates the nibble position in the DDR 4-bit mode. 0- the sequence is 'odd high nibble -> even high nibble -> odd low nibble -> even low nibble'; 1- the sequence is 'odd high nibble -> odd low nibble -> even high nibble -> even low nibble'.
5 MSBSEL	Multi / Single block select This field enables multiple block DATA line data transfers. For any other commands, this field can be set to 0. If this field is 0, it is not necessary to set the Block Count register. See Command Transfer Type (CMD_XFR_TYP). 0b - Single block 1b - Multiple blocks
4 DTDSEL	Data transfer direction select This field defines the direction of DATA line data transfers. The field is set to 1 by the host driver to transfer data from the SD card to uSDHC and is set to 0 for all other commands. If a CMD with the response data size cannot be divided by 64 bytes, is given without a STOP CMD, and is followed by a software reset, then you must write a 0 to DTDSEL before inserting a software reset. This is necessary to avoid any issue during the SD card reinitialization. Read multiple blocks will not be affected as it is required to have a STOP CMD. 0b - Write (Host to card) 1b - Read (Card to host)
3 DDR_EN	Dual data rate mode selection
2	Auto CMD12 enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
AC12EN	Multiple block transfers for memory require a CMD12 to stop the transaction. When this field is set to 1, uSDHC issues a CMD12 automatically when the last block transfer has completed. The host driver is not set this field to issue commands that do not require CMD12 to stop a multiple block data transfer. In particular, secure commands defined in File Security Specification (see reference list) do not require CMD12. In single block transfer, uSDHC ignores this field no matter it is set or not. 0b - Disable 1b - Enable
1 BCEN	Block count enable This field is used to enable the Block Count register, which is only relevant for multiple block transfers. When this field is 0, the internal counter for block is disabled, which is useful in executing an infinite transfer. 0b - Disable 1b - Enable
0 DMAEN	DMA enable This field enables DMA functionality. If this field is set to 1, a DMA operation begins when the host driver sets the DPSEL field of this register. Whether the simple DMA or the advanced DMA is active depends on the DMA Select field of the Protocol Control register. 0b - Disable 1b - Enable

39.4.1.21 Force Event (FORCE_EVENT)

Offset

Register	Offset
FORCE_EVENT	50h

Function

This register is not a physically implemented register. Rather, it is an address at which the Interrupt Status register can be written if the corresponding field of the Interrupt Status Enable Register is set. This register is a write only register and writing 0 to it has no effect. Writing 1 to this register sets the corresponding field of Interrupt Status register. A read from this register always results in 0's. to change the corresponding status fields in the Interrupt Status register, make sure to set IPGEN field in System Control Register so that peripheral clock is always active.

Forcing a card interrupt generates a short pulse on the DATA1 line, and the driver might treat this interrupt as a normal interrupt. The interrupt service routine might skip polling the card interrupt factor as the interrupt is self cleared.

Provided under NDA only
COMPANY PROPRIETARY
Beijing eSOL Automotive Technology Co., Ltd.
c8c46e5e-5421-48a8-b251-590d3a0624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0
W	FEVTCINT			FEVTDMAE		FEVTTNE		FEVTAC1...		FEVTDEBE	FEVTDCE	FEVTDTOE	FEVTCIE	FEVTCEBE	FEVTCCE	FEVTCTOE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	0		0	0	0	0	0
W									FEVTCNI...			FEVTAC1...	FEVTAC1...	FEVTAC1...	FEVTAC1...	FEVTAC1...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 FEVTCINT	Force event card interrupt Writing 1 to this field generates a short low-level pulse on the internal DATA1 line, as if a self-clearing interrupt was received from the external card. If enabled, the CINT field is set and the interrupt service routine might treat this interrupt as a normal interrupt from the external card.
30-29 —	Reserved
28 FEVTDMAE	Force event DMA error Forces the DMAE field of Interrupt Status register to be set
27 —	Reserved
26 FEVTTNE	Force tuning error Forces the TNE field of Interrupt Status register to be set
25 —	Reserved
24 FEVTAC12E	Force event Auto Command 12 error Forces the AC12E field of Interrupt Status register to be set
23 —	Reserved
22 FEVTDEBE	Force event data end bit error Forces the DEBE field of Interrupt Status register to be set

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 FEVTDCE	Force event data CRC error Forces the DCE field of Interrupt Status register to be set
20 FEVTDTOE	Force event data time out error Force the DTOE field of Interrupt Status register to be set
19 FEVTCIE	Force event command index error Forces the CCE field of Interrupt Status register to be set
18 FEVTCEBE	Force event command end bit error Forces the CEBE field of Interrupt Status register to be set
17 FEVTCCE	Force event command CRC error Forces the CCE field of Interrupt Status register to be set
16 FEVTCTOE	Force event command time out error Forces the CTOE field of Interrupt Status register to be set
15-8 —	Reserved
7 FEVTCNIBAC12E	Force event command not executed by Auto Command 12 error Forces the CNIBAC12E field in the Auto Command12 Error Status register to be set
6-5 —	Reserved
4 FEVTAC12IE	Force event Auto Command 12 index error Forces the AC12IE field in the Auto Command12 Error Status register to be set
3 FEVTAC12EBE	Force event Auto Command 12 end bit error Forces the AC12EBE field in the Auto Command12 Error Status register to be set
2 FEVTAC12CE	Force event auto command 12 CRC error Forces the AC12CE field in the Auto Command12 Error Status register to be set
1 FEVTAC12TOE	Force event auto command 12 time out error Forces the AC12TOE field in the Auto Command12 Error Status register to be set
0 FEVTAC12NE	Force event auto command 12 not executed Forces the AC12NE field in the Auto Command12 Error Status register to be set

Provided under NDA only
COMPANY PROPRIETARY
Bosch Automotive Technology Co., Ltd.
c0c45e6e-4821-48a8-b251-390d3ab624c1

39.4.1.22 ADMA Error Status (ADMA_ERR_STATUS)

Offset

Register	Offset
ADMA_ERR_STATUS	54h

Function

When an ADMA Error Interrupt has occurred, the ADMA error sates field in this register holds the ADMA state and the ADMA System Address register holds the address around the error descriptor.

For recovering from this error, the host driver requires the ADMA state to identify the error descriptor address as follows:

- ST_STOP: Previous location set in the ADMA System Address register is the error descriptor address.
- ST_FDS: Current location set in the ADMA System Address register is the error descriptor address.
- ST_CADR: This state is never set because it only increments the descriptor pointer and does not generate an ADMA error.
- ST_TFR: Previous location set in the ADMA System Address register is the error descriptor address.

In case of a write operation, the host driver should use the ACMD22 to get the number of the written block, rather than using this information, because unwritten data might exist in the host controller.

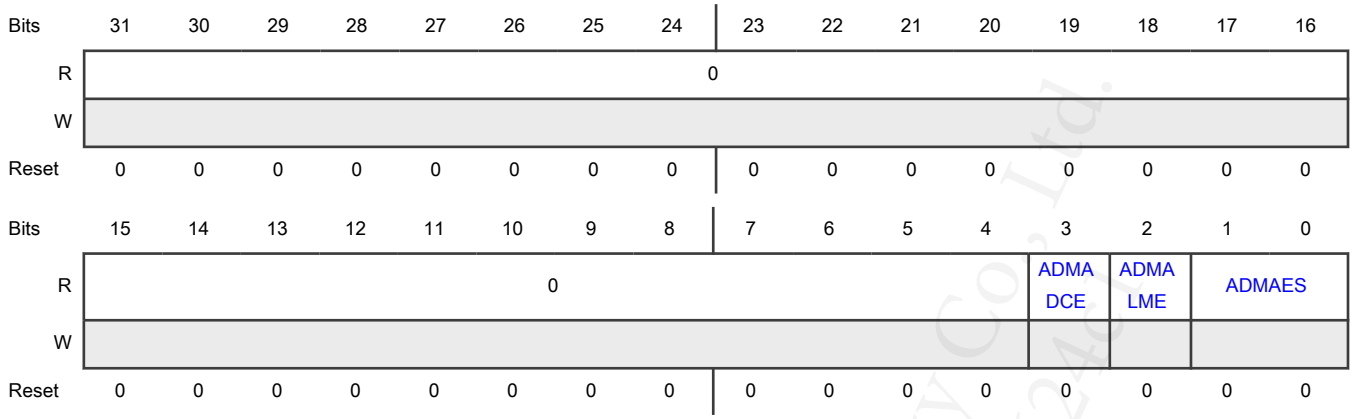
The host controller generates the ADMA Error Interrupt when it detects invalid descriptor data (Valid=0) in the ST_FDS state. The host driver can distinguish this error by reading the Valid field of the error descriptor.

Table 261. ADMA error state coding

D01-D00	ADMA error state (when error has occurred)	Contents of ADMA System Address register
00	ST_STOP (Stop DMA)	Holds the address of the next executable descriptor command
01	ST_FDS (Fetch descriptor)	Holds the valid descriptor address
10	ST_CADR (Change address)	No ADMA error is generated
11	ST_TFR (Transfer data)	Holds the address of the next executable descriptor command

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-43a8-b251-39013ac24d1

Diagram



Fields

Field	Function
31-4 —	Reserved
3 ADMADCE	ADMA descriptor error This error occurs when invalid descriptor fetched by ADMA. 0b - No error 1b - Error
2 ADMALME	ADMA length mismatch error This error occurs in the following two cases: • While the block count enable is being set, the total data length specified by the descriptor table is different from that specified by the block count and block length. • Total data length cannot be divided by the block length. 0b - No error 1b - Error
1-0 ADMAES	ADMA error state (when ADMA error is occurred) This field indicates the state of the ADMA when an error has occurred during an ADMA data transfer. See ADMA Error Status (ADMA_ERR_STATUS) for more details.

39.4.1.23 ADMA System Address (ADMA_SYS_ADDR)

Offset

Register	Offset
ADMA_SYS_ADDR	58h

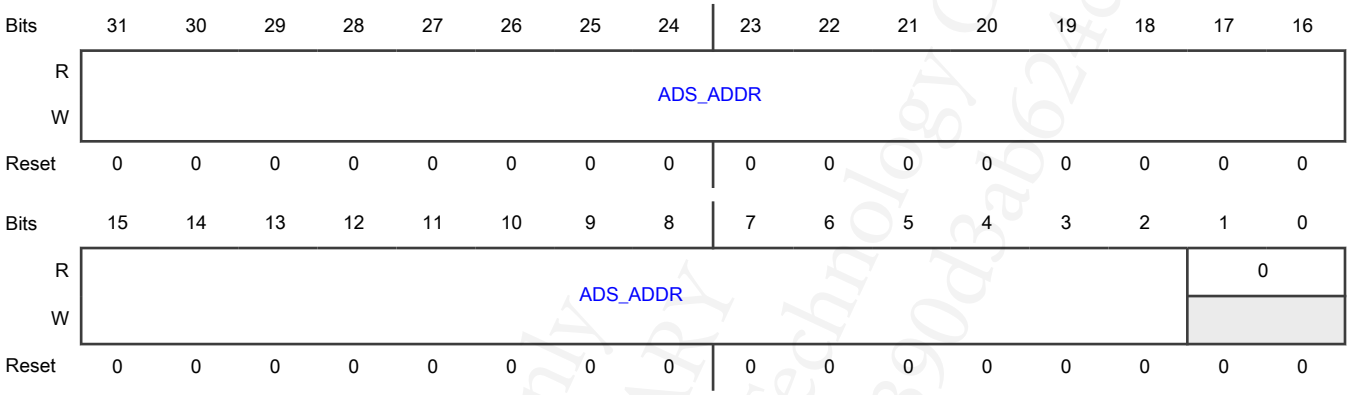
Preliminary NDA only
COMPANY PROPRIETARY
Beijing C&A Automotive Technology Co., Ltd.
c8c46255-8921-4518-b251-390a3ab624c4

Function

This register holds the word address of the executing command in the Descriptor table. At the start of ADMA, the host driver sets the start address of the Descriptor table. The ADMA engine increments this register address whenever fetching a Descriptor command. When the ADMA is stopped at the Block Gap, this register indicates the address of the next executable Descriptor command. When the ADMA Error Interrupt is generated, this register holds the valid Descriptor address depending on the ADMA state. The lower 2 bits of this register is tied to '0' so the ADMA address is always word aligned.

Because this register supports dynamic address reflecting, when TC field is set, it automatically alters the value of internal address counter, so the software cannot change this register when TC field is set. Such restriction is also listed in Software restrictions.

Diagram



Fields

Field	Function
31-2 ADS_ADDR	ADMA system address This field contains the physical system memory address used for ADMA transfers.
1-0 —	Reserved

39.4.1.24 DLL (Delay Line) Control (DLL_CTRL)

Offset

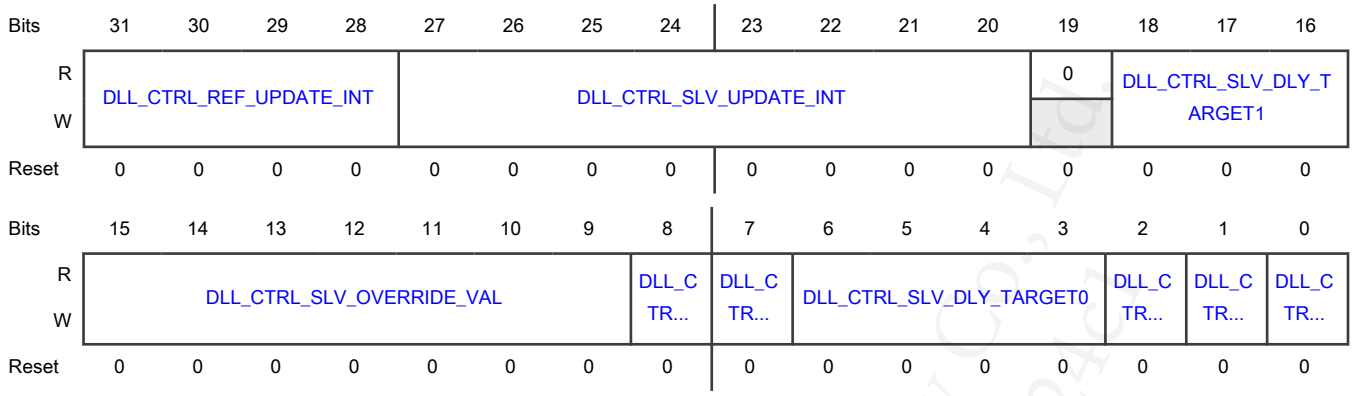
Register	Offset
DLL_CTRL	60h

Function

This register contains control fields for DLL.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-28 DLL_CTRL_REF_UPDATE_INT	DLL control loop update interval The interval cycle is (2 + REF_UPDATE_INT) * REF_CLOCK. By default, the DLL control loop updates every two REF_CLOCK cycles. It should be noted that increasing the reference delay-line update interval reduces the ability of the DLL to adjust to fast changes in conditions that might effect the delay (such as voltage and temperature)
27-20 DLL_CTRL_SLV_UPDATE_INT	Slave delay line update interval If default 0 is used, it means 256 cycles of REF_CLOCK. A value of 0x0f results in 15 cycles and so on. Note that software can always cause an update of the slave-delay line using the SLV_FORCE_UPDATE register. Note that the slave delay line also updates automatically when the reference DLL transitions to a locked state (from an un-locked state).
19 —	Reserved
18-16 DLL_CTRL_SLV_DLY_TARGET1	DLL slave delay target1 See DLL_CTRL_SLV_DLY_TARGET0 below.
15-9 DLL_CTRL_SLV_OVERRIDE_VAL	DLL slave override val When SLV_OVERRIDE = 1, this field is used to select 1 of 128 physical taps manually. A value of 0 selects tap 1, and a value of 0x7f selects tap 128.
8 DLL_CTRL_SLV_OVERRIDE	DLL slave override Set this field to 1 to Enable manual override for slave delay chain using SLV_OVERRIDE_VAL; to set 0 to disable manual override. This feature does not require the DLL to be enabled using the ENABLE field. In fact to reduce power, if SLV_OVERRIDE is used, it is recommended to disable the DLL with ENABLE = 0
7	DLL gate update

Table continues on the next page...

Table continued from the previous page...

Field	Function
DLL_CTRL_GATE_UPDATE	Set this field to 1 to prevent the DLL from updating (because when clock_in exists, glitches might appear during DLL updates). This field might be used by software if such a condition occurs. Clear the bit to 0 to allow the DLL to update automatically.
6-3 DLL_CTRL_SLV_DLY_TARGET0	DLL slave delay target0 The delay target for uSDHC loopback read clock can be programmed in 1/16th increments of an ref_clock half-period. The delay is (({DLL_CTRL_SLV_DLY_TARGET1,DLL_CTRL_SLV_DLY_TARGET0} +1)* REF_CLOCK / 2) / 16 So the input read-clock can be delayed relative input data from (REF_CLOCK / 2) / 16 to REF_CLOCK * 4. **NOTE** For the restrictions of delay cell implementation, the delay target must be set between REF_CLOCK/16 and REF_CLOCK*2 when REF_CLOCK is running at 200 MHz. When REF_CLOCK frequency is slower than 100 MHz, the maximum delay target might not reach REF_CLOCK*2.
2 DLL_CTRL_SLV_FORCE_UPD	DLL slave delay line Setting this field to 1, forces the slave delay line to update to the DLL calibrated value immediately. The slave delay line updates automatically based on the SLV_UPDATE_INT interval or when a DLL lock condition is sensed. Subsequent forcing of the slave-line update can only occur if SLV_FORCE_UP is set back to 0 and then asserted again (edge triggered). Be sure to use it when uSDHC is idle. This function might not work when uSDHC is working on data / cmd / response.
1 DLL_CTRL_RESET	DLL reset Setting this field to 1 force a reset on DLL. This causes the DLL to lose lock and re-calibrate to detect an REF_CLOCK half period phase shift. This signal is used by the DLL as edge-sensitive, so to create a subsequent reset, RESET must be taken low and then asserted again.
0 DLL_CTRL_ENABLE	DLL and delay chain Set this field to 1 to enable the DLL and delay chain; otherwise; set to 0 to bypasses DLL. Note that using the slave delay line override feature with SLV_OVERRIDE and SLV_OVERRIDE VAL, the DLL does not need to be enabled.

39.4.1.25 DLL Status (DLL_STATUS)

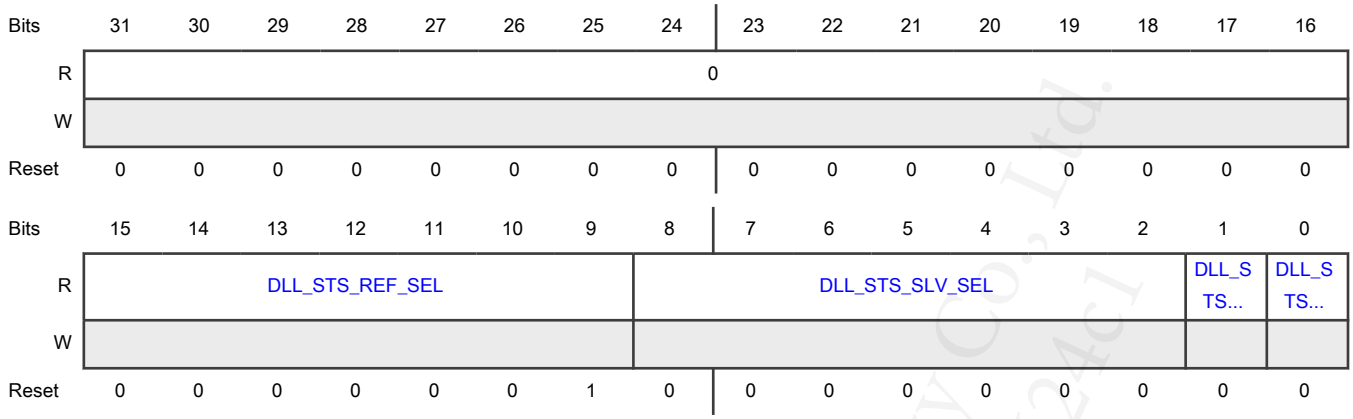
Offset

Register	Offset
DLL_STATUS	64h

Function

This register contains the DLL status information. All fields are read only and reads the same as the power-reset value.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-9 DLL_STS_REF_SEL	Reference delay line select taps This is encoded by 7 fields for 127 taps.
8-2 DLL_STS_SLV_SEL	Slave delay line select status This is the instant value generated from reference chain. Because the reference chain can only be updated when REF_CLOCK is detected, this value should be the right value to be updated when the reference is locked.
1 DLL_STS_REF_LOCK	Reference DLL lock status This signifies that the DLL has detected and locked to a half-phase ref_clock shift, allowing the slave delay-line to perform programmed clock delays
0 DLL_STS_SLV_LOCK	Slave delay-line lock status This signifies that a valid calibration has been set to the slave-delay line and that the slave-delay line is implementing the programmed delay value

39.4.1.26 CLK Tuning Control and Status (CLK_TUNE_CTRL_STATUS)

Offset

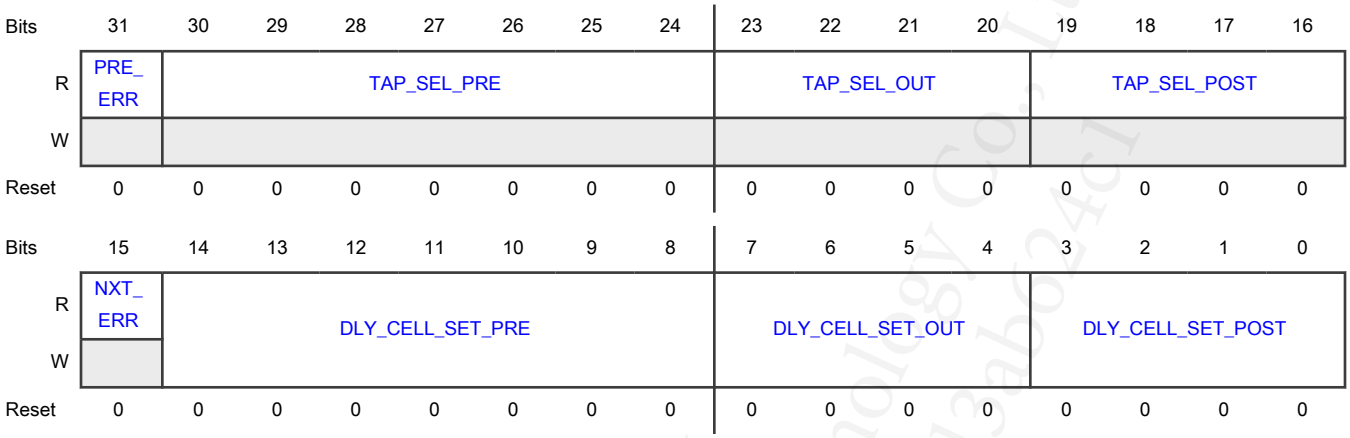
Register	Offset
CLK_TUNE_CTRL_STATUS	68h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEC Automotive Technology Co., Ltd.
c8c462e5-4621-4a8-251-390a3ab624c1

Function

This register contains the Clock Tuning Control status information. All fields are read only and reads the same as the power-reset value. This register is added to support SD3.0 UHS-I SDR104 mode and EMMC HS200 mode.

Diagram



Fields

Field	Function
31 PRE_ERR	PRE error PRE error which means the number of delay cells added on the feedback clock is too small. It is valid only when SMP_CLK_SEL of Mix control register (bit23 of 0x48) is enabled.
30-24 TAP_SEL_PRE	TAP_SEL_PRE Reflects the number of delay cells added on the feedback clock between the feedback clock and CLK_PRE. When AUTO_TUNE_EN (bit24 of 0x48) is disabled, TAP_SEL_PRE is always equal to DLY_CELL_SET_PRE. When AUTO_TUNE_EN (bit24 of 0x48) is enabled, TAP_SEL_PRE is updated automatically according to the status of the auto tuning circuit to adjust the sample clock phase.
23-20 TAP_SEL_OUT	Delay cells added on the feedback clock between CLK_PRE and CLK_OUT Reflects the number of delay cells added on the feedback clock between CLK_PRE and CLK_OUT.
19-16 TAP_SEL_POST	Delay cells added on the feedback clock between CLK_OUT and CLK_POST Reflects the number of delay cells added on the feedback clock between CLK_OUT and CLK_POST.
15 NXT_ERR	NXT error NXT error which means the number of delay cells added on the feedback clock is too large. It's valid only when SMP_CLK_SEL of Mix control register (bit23 of 0x48) is enabled.
14-8	delay cells on the feedback clock between the feedback clock and CLK_PRE Set the number of delay cells on the feedback clock between the feedback clock and CLK_PRE.

Table continues on the next page...

Table continued from the previous page...

Field	Function
DLY_CELL_SET_PRE	
7-4 DLY_CELL_SET_OUT	Delay cells on the feedback clock between CLK_PRE and CLK_OUT Set the number of delay cells on the feedback clock between CLK_PRE and CLK_OUT.
3-0 DLY_CELL_SET_POST	Delay cells on the feedback clock between CLK_OUT and CLK_POST Set the number of delay cells on the feedback clock between CLK_OUT and CLK_POST.

39.4.1.27 Strobe DLL control (STROBE_DLL_CTRL)

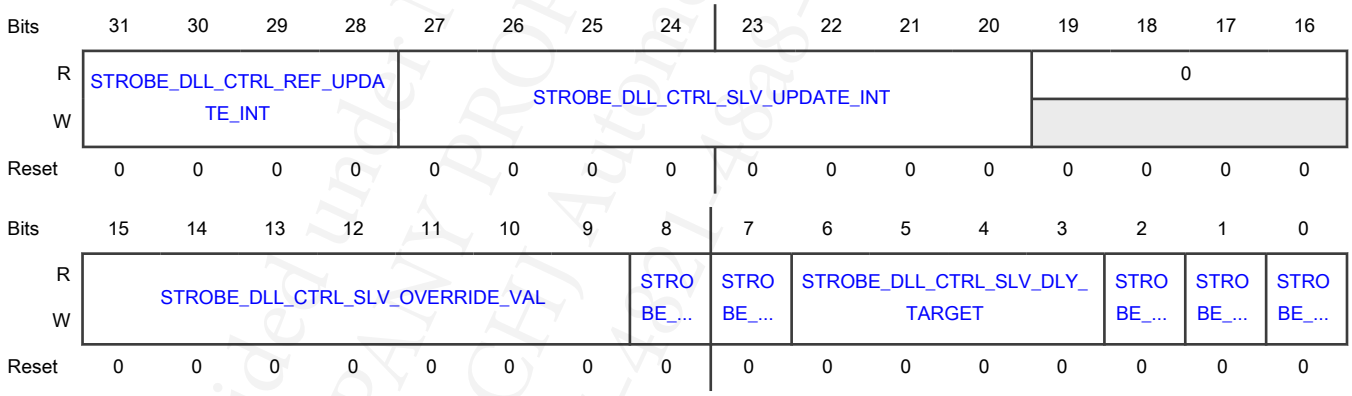
Offset

Register	Offset
STROBE_DLL_CTRL	70h

Function

This register contains the strobe DLL control information.

Diagram



Fields

Field	Function
31-28	Strobe DLL control reference update interval

Table continues on the next page...

Table continued from the previous page...

Field	Function
STROBE_DLL_CTRL_REF_UPDATE_INT	The interval cycle is (2 + STROBE_REF_UPDATE_INT) * STROBE_REF_CLOCK. By default, the DLL control loop updates every two STROBE_REF_CLOCK cycles. **NOTE** Increasing the reference delay-line update interval reduces the ability of the DLL to adjust to fast changes in conditions that might effect the delay (such as voltage and temperature)
27-20 STROBE_DLL_CTRL_SLV_UPDATE_INT	Strobe DLL control slave update interval Slave delay line update interval. If default 0 is used, it means 256 cycles of STROBE_REF_CLOCK. A value of 0x0F results in 15 cycles and so on. **NOTE** software can always cause an update of the slave-delay line using the STROBE_SLV_FORCE_UPDATE register. The slave delay line also updates automatically when the reference DLL transitions to a locked state (from an un-locked state).
19-16 —	This field is reserved. This read-only field is reserved and always has the value 0.
15-9 STROBE_DLL_CTRL_SLV_OVERRIDE_VAL	Strobe DLL control slave Override value When STROBE_SLV_OVERRIDE = 1, this field is used to manually select one of 128 physical taps. A value of 0 selects tap 1, and a value of 0x7F selects tap 128.
8 STROBE_DLL_CTRL_SLV_OVERRIDE	Strobe DLL control slave override Set this field to 1 to Enable manual override for slave delay chain using STROBE_SLV_OVERRIDE_VAL; set this field to 0 to disable manual override. This feature does not require the DLL to be enabled using the ENABLE field. In fact to reduce power, if STROBE_SLV_OVERRIDE is used, it is recommended to disable the DLL with ENABLE = 0.
7 STROBE_DLL_CTRL_GATE_UPDATE	Strobe DLL control gate update Set this field to 1 to prevent the DLL from updating (because when STROBE_CLOCK_IN exists, glitches might appear during DLL updates). This field can be used by software if such a condition occurs. Clear the field to 0 to allow the DLL to update automatically.
6-3 STROBE_DLL_CTRL_SLV_DLY_TARGET	Strobe DLL Control Slave Delay Target The delay target for uSDHC loopback read clock can be programmed in 1/16th increments of an STROBE_REF_CLOCK half-period. The delay is ((STROBE_DLL_CTRL_SLV_DLY_TARGET + 1) * STROBE_REF_CLOCK / 2) / 16, So the input read-clock can be delayed relative input data from (STROBE_REF_CLOCK / 2) / 16 to (STROBE_REF_CLOCK *4)/16. **NOTE** For the restrictions of delay cell implementation, the minimum delay target that can be set is REF_CLOCK/16.
2	Strobe DLL control slave force updated

Table continues on the next page...

Table continued from the previous page...

Field	Function
STROBE_DLL_CTRL_SLV_FORCE_UPD	Setting this field to 1, forces the slave delay line to update to the DLL calibrated value immediately. The slave delay line should automatically update the STROBE_SLV_UPDATE_INT interval or when a DLL lock condition is sensed. Subsequent forcing of the slave-line update can only occur if STROBE_SLV_FORCE_UP is set back to 0 and then asserted again (edge triggered). Be sure to use it when uSDHC is idle. This function might not work when uSDHC is working on data / cmd / response.
1 STROBE_DLL_CTRL_RESET	Strobe DLL control reset Setting this field to 1 to force a reset on DLL. This causes the DLL to lose lock and re-calibrate to detect an REF_CLOCK half period phase shift. This signal is used by the DLL as edge-sensitive, to create a subsequent reset, RESET must be taken low and then asserted again.
0 STROBE_DLL_CTRL_ENABLE	Strobe DLL control enable Set this field to 1 to enable the DLL and delay chain; otherwise, set to 0 to bypasses DLL. **NOTE** Using the slave delay line override feature with STROBE_SLV_OVERRIDE and STROBE_SLV_OVERRIDE VAL, the DLL does not need to be enabled.

39.4.1.28 Strobe DLL status (STROBE_DLL_STATUS)

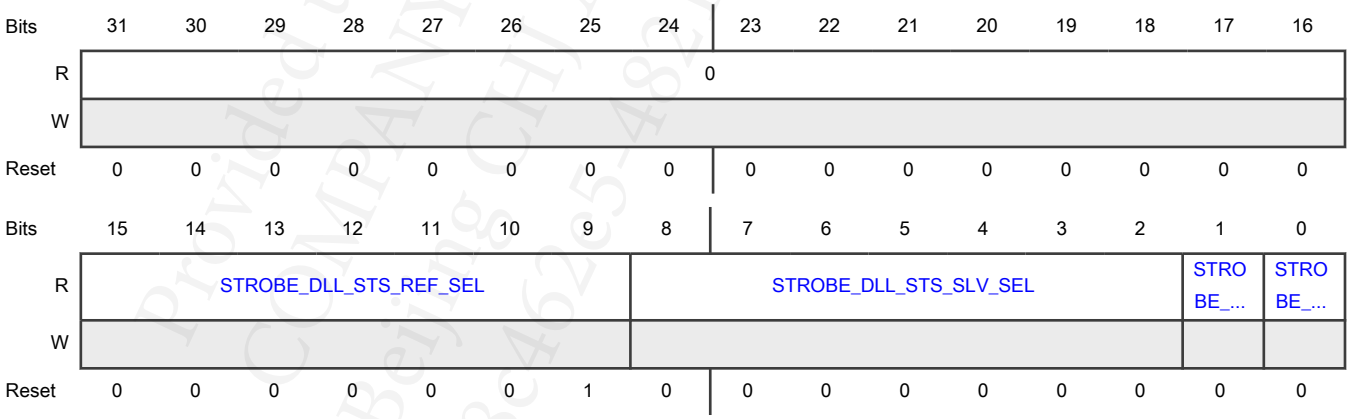
Offset

Register	Offset
STROBE_DLL_STATUS	74h

Function

This register contains the strobe DLL status information. All fields are read only and read the same as the power-reset value.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462525-4821-48a8-b251-39033b62fc1

Fields

Field	Function
31-16 —	This field is reserved. This read-only field is reserved and always has the value 0.
15-9 STROBE_DLL_STS_REF_SEL	Strobe DLL status reference select Reference delay line select taps. This is encoded by 7 fields for 127 taps.
8-2 STROBE_DLL_STS_SLV_SEL	Strobe DLL status slave select Slave delay line select status. This is the instant value generated from reference chain. Because the reference chain can only be updated when STROBE_REF_CLOCK is detected, this value can be updated with the right value when the reference is locked.
1 STROBE_DLL_STS_REF_LOCK	Strobe DLL status reference lock This signifies that the DLL has detected and locked to a half-phase REF_CLOCK shift, it allows the slave delay-line to perform programmed clock delays.
0 STROBE_DLL_STS_SLV_LOCK	Strobe DLL status slave lock Slave delay-line lock status. This signifies that a valid calibration has been set to the slave-delay line, and the slave-delay line is implementing the programmed delay value.

39.4.1.29 Vendor Specific Register (VEND_SPEC)

Offset

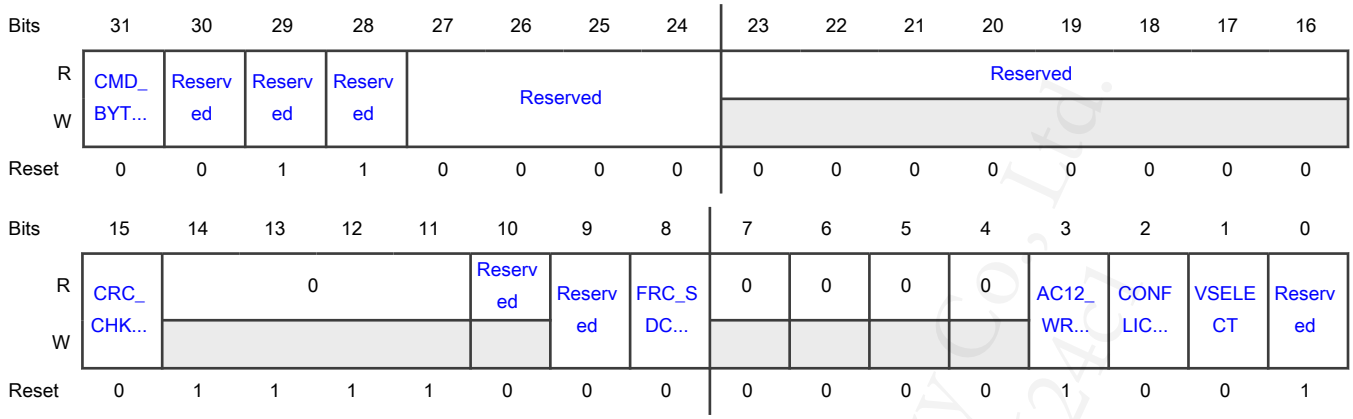
Register	Offset
VEND_SPEC	C0h

Function

This register contains the vendor specific control/status register.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-3903ab624c1

Diagram



Fields

Field	Function
31 CMD_BYTE_EN	Byte access This field controls the byte access. 0b - Disable 1b - Enable
30 —	Reserved Always write as 0.
29 —	Reserved Always write as 1
28 —	Reserved Always write as 0.
27-24 —	Reserved Always write as 4'b0000.
23-16 —	Reserved Always write as 8'h00.
15 CRC_CHK_DIS	CRC Check Disable 0b - Check CRC16 for every read data packet and check CRC fields for every write data packet 1b - Ignore CRC16 check for every read data packet and ignore CRC fields check for every write data packet
14-11 —	Reserved Reserved
10	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Always write as 0.
9 —	Reserved Always write as 0.
8 FRC_SDCLK_ON	Force CLK Force CLK output active Do not set this bit to 1 unless it is necessary. Also, make sure that this bit is cleared when uSDHC's clock is about to be changed (frequency change, clock source change, or delay chain tuning). 0b - CLK active or inactive is fully controlled by the hardware. 1b - Force CLK active
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 AC12_WR_CHKBUSY_EN	Check busy enable Check busy enable after auto CMD12 for write data packet 0b - Do not check busy after auto CMD12 for write data packet 1b - Check busy after auto CMD12 for write data packet
2 CONFLICT_CHK_EN	Conflict check enable It is not implemented in uSDHC IP. 0b - Conflict check disable 1b - Conflict check enable
1 VSELECT	Voltage selection Change the value of output signal VSELECT to control the voltage on pads for external card. There must be a control circuit out of uSDHC to change the voltage on pads. 0b - Change the voltage to high voltage range, around 3.0 V 1b - Change the voltage to low voltage range, around 1.8 V
0	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Always write as 0.

39.4.1.30 MMC Boot (MMC_BOOT)

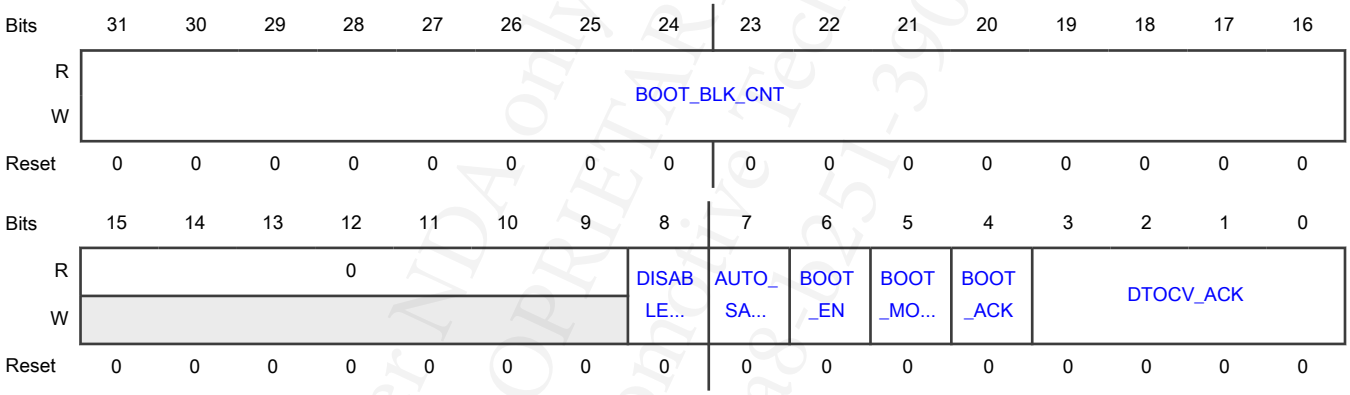
Offset

Register	Offset
MMC_BOOT	C4h

Function

This register contains the MMC Fast Boot control register.

Diagram



Fields

Field	Function
31-16 BOOT_BLK_CNT	Stop At Block Gap value of automatic mode The value defines the Stop At Block Gap value of automatic mode. When received, card block cnt is equal to (BLK_CNT - BOOT_BLK_CNT) and AUTO_SABG_EN is 1, then Stop At Block Gap. Here, BLK_CNT is defined in the Block Attributes Register, field 31 - 16 of 0x04.
15-9 —	Reserved
8 DISABLE_TIME_OUT	Time out **NOTE** When this field is set, there is no timeout check no matter whether BOOT_EN is set or not.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Enable time out 1b - Disable time out
7 AUTO_SABG_EN	Auto stop at block gap During boot, enable auto stop at block gap function. This function is triggered, and host stops at block gap when received card block cnt is equal to (BLK_CNT - BOOT_BLK_CNT).
6 BOOT_EN	Boot enable Boot mode enable 0b - Fast boot disable 1b - Fast boot enable
5 BOOT_MODE	Boot mode Boot mode select 0b - Normal boot 1b - Alternative boot
4 BOOT_ACK	BOOT ACK Boot ACK mode select 0b - No ack 1b - Ack
3-0 DTOCV_ACK	DTOCV_ACK Boot ACK time out counter value 0000b - SDCLK x 2^32 0001b - SDCLK x 2^33 0010b - SDCLK x 2^18 0011b - SDCLK x 2^19 0100b - SDCLK x 2^20 0101b - SDCLK x 2^21 0110b - SDCLK x 2^22 0111b - SDCLK x 2^23 1110b - SDCLK x 2^30 1111b - SDCLK x 2^31

Provided under NDA only
COPYRIGHT PROPRIETARY
Beijing CCAT Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

39.4.1.31 Vendor Specific 2 Register (VEND_SPEC2)

Offset

Register	Offset
VEND_SPEC2	C8h

Function

This register contains the vendor specific control 2 register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FBCLK_TAP_SEL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EN_32K_...	Reserved	Reserved	ACMD23_...	HS400_R...	HS400_W...	0	0		TUNING_...	TUNING_...	TUNING_...	CARD_IN...	0		
W																
Reset	1	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-16 FBCLK_TAP_SEL	Enable extra delay on internal feedback clock
15 EN_32K_CLK	Enable 32khz clock for card detection Use 32khz clock for card detection
14 —	Reserved
13 —	Reserved
12 ACMD23_ARGU2_EN	Argument2 register enable for ACMD23 0b - Disable 1b - Argument2 register enable for ACMD23 sharing with SDMA system address register. Default is enabled.
11	HS400 read clock stop enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
HS400_RD_CLK_STOP_EN	Only stop clock at read block gap.
10 HS400_WR_CLK_STOP_EN	HS400 write clock stop enable Only stop clock at write block gap.
9 —	Reserved
8-7 —	Reserved
6 TUNING_CMD_EN	Tuning command enable Enable the auto tuning circuit to check the CMD line. 0b - Auto tuning circuit does not check the CMD line. 1b - Auto tuning circuit checks the CMD line.
5 TUNING_1bit_EN	Tuning 1bit enable Enable the auto tuning circuit to check the DATA0 only. It is used with the TUNING_8bit_EN together.
4 TUNING_8bit_EN	Tuning 8bit enable Enable the auto tuning circuit to check the DATA[7:0]. It is used with the TUNING_1bit_EN together. 0b00 - Tuning circuit only checks the DATA[3:0] 0b01 - Tuning circuit only checks the DATA0 0b10 - Tuning circuit checks the whole DATA[7:0] 0b11 - Invalid **NOTE** The format of these two fields are [TUNNING_8bit_EN:TUNNING_1bit_EN].
3 CARD_INT_D3_TEST	Card interrupt detection test This field is used only for debugging. 0b - Check the card interrupt only when DATA3 is high. 1b - Check the card interrupt by ignoring the status of DATA3.
2-0 —	Reserved

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46235-4891-48a8-b251-390d3ab624c1

39.4.1.32 Tuning Control (TUNING_CTRL)

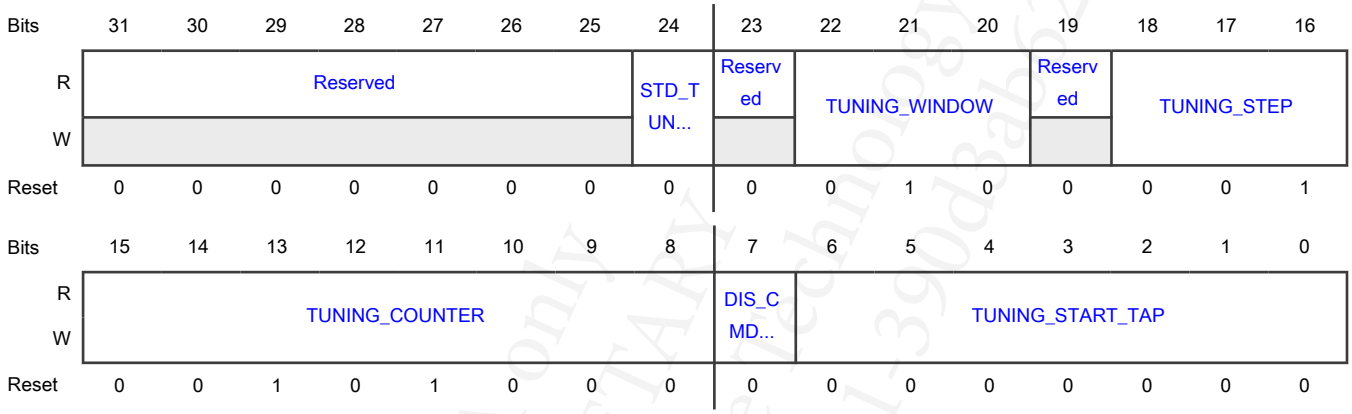
Offset

Register	Offset
TUNING_CTRL	CCh

Function

The register contains configuration of tuning circuit.

Diagram



Fields

Field	Function
31-25 —	Reserved
24 STD_TUNING_EN	Standard tuning circuit and procedure enable This field is used to enable standard tuning circuit and procedure.
23 —	Reserved
22-20 TUNING_WINDOW	Data window Select data window value for auto tuning
19 —	Reserved
18-16	TUNING_STEP

Table continues on the next page...

Table continued from the previous page...

Field	Function
TUNING_STEP	The increasing delay cell steps in tuning procedure.
15-8 TUNING_COUNTER	Tuning counter The MAX repeat CMD19 times in tuning procedure.
7 DIS_CMD_CHK_FOR_STD_TUNING	Disable command check for standard tuning Writing 1 to this field disables command check (command CRC, CMD end bit error, and CMD index error or CMD timeout error) for standard tuning flow after each tuning command is sent.
6-0 TUNING_START_TAP	Tuning start The start delay cell point when send first CMD19 in tuning procedure.

39.4.1.33 Command Queuing Version (CQVER)

Offset

Register	Offset
CQVER	100h

Function

This register provides information about the version of the e•MMC CQ standard that is implemented by CQE, in BCD format. The current version is 5.1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				MAJOR_VN				MINOR_VN				VERSION_SUFFIX			
W																
Reset	0	0	0	0	0	1	0	1	0	0	0	1	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46215-482a-b251-390d3ab624c1

Fields

Field	Function
31-12 —	Reserved
11-8 MAJOR_VN	e •MMC major version number Digit right of decimal point, in BCD format
7-4 MINOR_VN	e •MMC minor version number Digit right of decimal point, in BCD format
3-0 VERSION_SUFFIX	e •MMC version suffix The second digit right of decimal point, in BCD format

39.4.1.34 Command Queuing Capabilities (CQCAP)

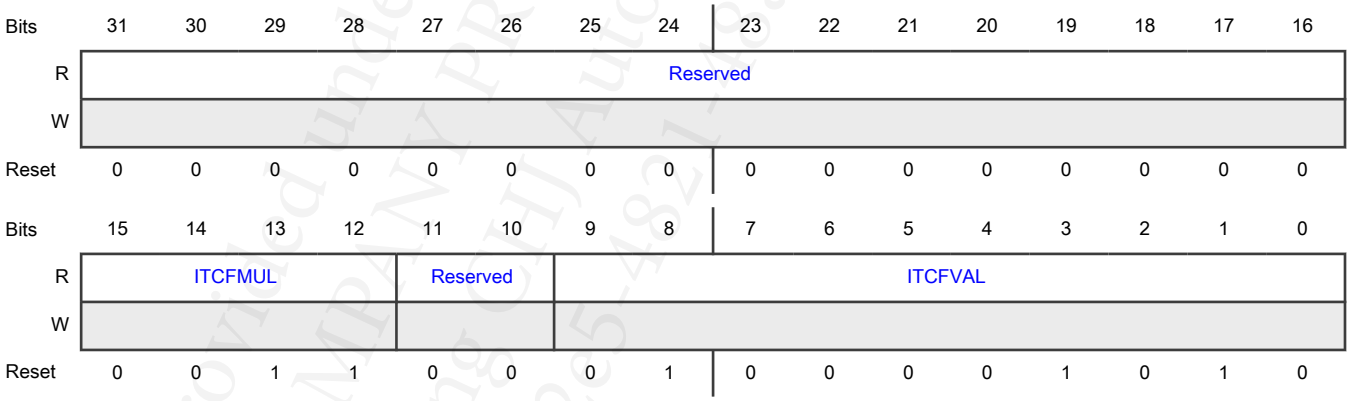
Offset

Register	Offset
CQCAP	104h

Function

This register indicates hardware capabilities.

Diagram



Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-4ca8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
15-12 ITCFMUL	Internal timer clock frequency multiplier ITCFMUL and ITCFVAL indicate the frequency of the clock used for interrupt coalescing timer and for determining the SQS polling period. See ITCFVAL definition for details. 0001b - 0.001 MHz 0010b - 0.01 MHz 0011b - 0.1 MHz 0100b - 1 MHz 0101b - 10 MHz 0110-1001b - Reserved
11-10 —	Reserved
9-0 ITCFVAL	Internal timer clock frequency value ITCFMUL and ITCFVAL indicate the frequency of the clock used for interrupt coalescing timer and for determining the polling period when using periodic SEND_QUEUE_ STATUS (CMD13) polling. The clock frequency is calculated as ITCFVAL* ITCFMUL. For example, to encode 19.2 MHz, ITCFVAL is C0h (= 192 decimal) and ITCFMUL is 2h (0.1 MHz): 192 * 0.1 MHz = 19.2 MHz.

39.4.1.35 Command Queuing Configuration (CQCFG)

Offset

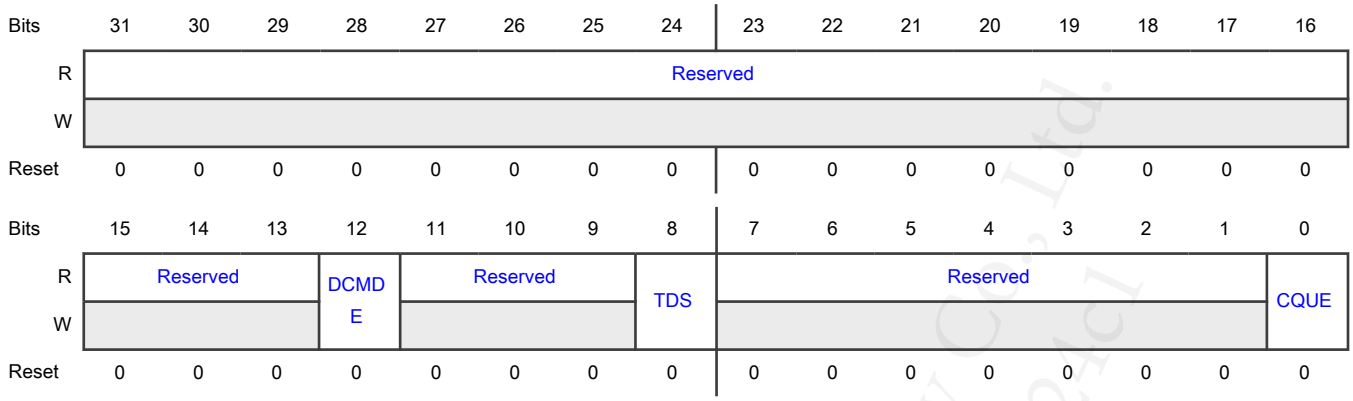
Register	Offset
CQCFG	108h

Function

This register controls CQE behavior affecting the general operation of command queuing module or operation of multiple tasks in the same time.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-13 —	Reserved
12 DCMDE	Direct command (DCMD) enable This bit indicates to the hardware whether the Task Descriptor in slot #31 of the TDL is a Data Transfer Task Descriptor, or a Direct Command Task Descriptor. CQE uses this bit when a task is issued in slot #31, to determine how to decode the Task Descriptor. 0b - Task descriptor in slot #31 is a Data Transfer Task Descriptor 1b - Task descriptor in slot #31 is a DCMD Task Descriptor
11-9 —	Reserved
8 TDS	Task descriptor size This bit indicates whether the task descriptor size is 128 bits or 64 bits as detailed in Data Structures section. This bit can only be configured when Command Queuing Enable bit is ‘0’ (command queuing is disabled) 0b - Task descriptor size is 64 bits 1b - Task descriptor size is 128 bits
7-1 —	Reserved
0 CQUE	Command queuing enable Software shall write '1' this bit when in order to enable command queuing mode (for example, enable CQE). When this bit is 0, CQE is disabled and software controls the e•MMC bus using the legacy e•MMC host controller. Before software writes '1' to this bit, software shall verify that the e•MMC host controller is in idle state and there are no commands or data transfers ongoing.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When software wants to exit command queuing mode, it shall clear all previous tasks if such exist before setting this bit to 0.

39.4.1.36 Command Queuing Control (CQCTL)

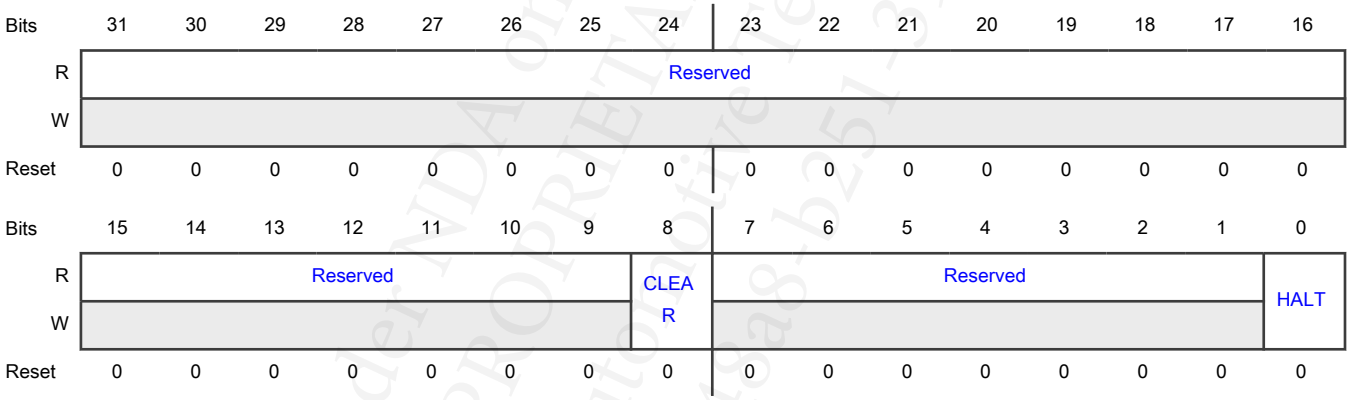
Offset

Register	Offset
CQCTL	10Ch

Function

This register controls CQE behavior affecting the general operation of command queuing module or operation of multiple tasks in the same time.

Diagram



Fields

Field	Function
31-9 —	Reserved
8 CLEAR	Clear all tasks Software shall write '1' this bit when it wants to clear all the tasks sent to the chip. This bit can only be written when CQE is in halt state (for example, Halt bit is 1). When software writes 1, the value of the register is updated to '1', and CQE shall reset CQTDBR register and all other context information for all unfinished tasks. Then CQE will clear this bit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Software should poll on this bit until it is set to back 0 and may then resume normal operation, by clearing the Halt bit. CQE does not communicate to the chip that the tasks were cleared. It is software's responsibility to order the chip to discard the tasks in its queue using CMDQ_TASK_MGMT command. Writing '0' to this register shall have no effect.
7-1 —	Reserved
0 HALT	Halt Host software shall write '1' to the bit when it wants to acquire software control over the e•MMC bus and disable CQE from issuing commands on the bus. For example, issuing a Discard Task command (CMDQ_TASK_MGMT) When software writes '1', CQE shall complete the ongoing task if such a task is in progress. Once the task is completed and CQE is in idle state, CQE shall not issue new commands and shall indicate so to software by setting this bit to 1. Software may poll on this bit until it is set to 1, and may only then send commands on the e•MMC bus. In order to exit halt state (for example, resume CQE activity), software shall clear this bit (write '0'). Writing '0' when the value is already '0' shall have no effect.

39.4.1.37 Command Queuing Interrupt Status (CQIS)

Offset

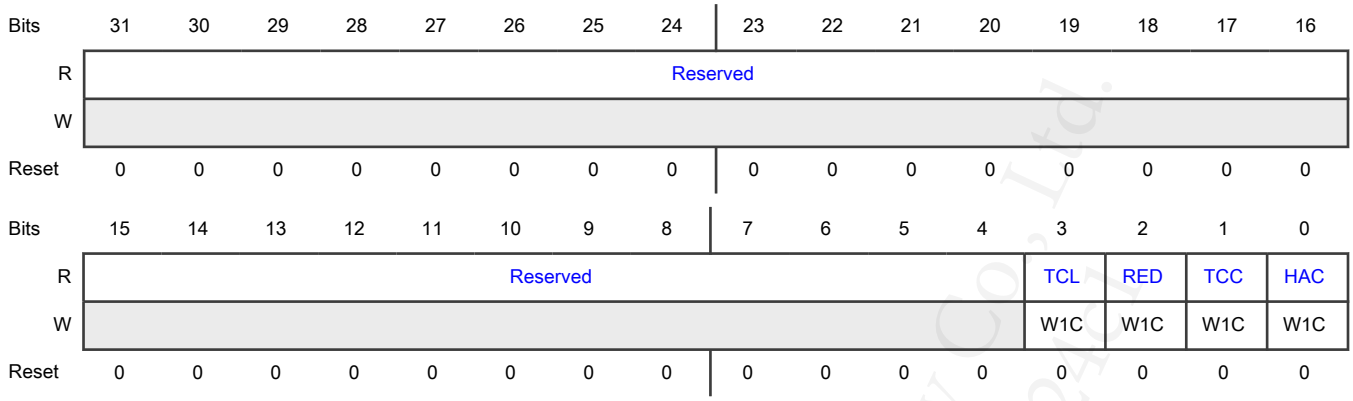
Register	Offset
CQIS	110h

Function

This register indicates pending interrupts that require service. Each bit in this registers is asserted in response a specific event, only if the respective bit is set in CQISTE register.

Provided under NDA
COMPANY PROPRIETARY
Beijing CTI Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-390f3a6624c1

Diagram



Fields

Field	Function
31-4 —	Reserved
3 TCL	Task cleared This status bit is asserted (if CQISTE[TCL_STE]=1) when a task clear operation is completed by CQE. The completed task clear operation is either an individual task clear (CQTCLR) or clearing of all tasks (CQCTL).
2 RED	Response error detected interrupt This status bit is asserted (if CQISTE[RED_STE]=1) when a response is received with an error bit set in the device status field. The contents of the device status field are listed in 6.13. Software uses CQRMEM register to configure which device status bit fields may trigger an interrupt, and which are masked.
1 TCC	Task complete interrupt This status bit is asserted (if CQISTE[TCC_STE]=1) when at least one of the following two conditions are met: • A task is completed and the INT bit is set in its Task Descriptor • Interrupt caused by Interrupt Coalescing logic (see 0)
0 HAC	Halt complete interrupt This status bit is asserted (if CQISTE[HAC_STE]=1) when halt bit in CQCTL register transitions from 0 to 1 indicating that host controller has completed its current ongoing task and has entered halt state.

39.4.1.38 Command Queuing Interrupt Status Enable (CQISTE)

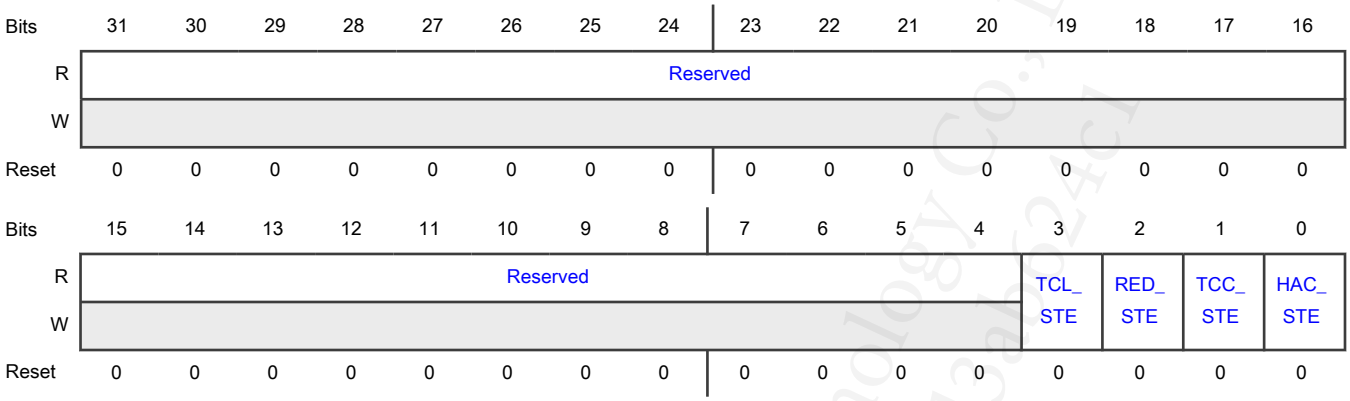
Offset

Register	Offset
CQISTE	114h

Function

This register enables and disables the reporting of the corresponding interrupt to host software in CQIS register. When a bit is set ('1') and the corresponding interrupt condition is active, then the bit is CQIS is asserted. Interrupt sources that are disabled (‘0’) are not indicated in the CQIS register. This register is bit-index matched to CQIS register.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 TCL_STE	Task cleared status enable 0b - CQIS[TCL] is disabled 1b - CQIS[TCL] is set when its interrupt condition is active
2 RED_STE	Response error detected status enable 0b - CQIS[RED] is disabled 1b - CQIS[RED] is set when its interrupt condition is active
1 TCC_STE	Task complete status enable 0b - CQIS[TCC] is disabled 1b - CQIS[TCC] is set when its interrupt condition is active
0 HAC_STE	Halt complete status enable 0b - CQIS[HAC] is disabled 1b - CQIS[HAC] is set when its interrupt condition is active

Provided under NDA only
COMPANY PROPRIETARY
Beijing BCTC Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab2241

39.4.1.39 Command Queuing Interrupt Signal Enable (CQISGE)

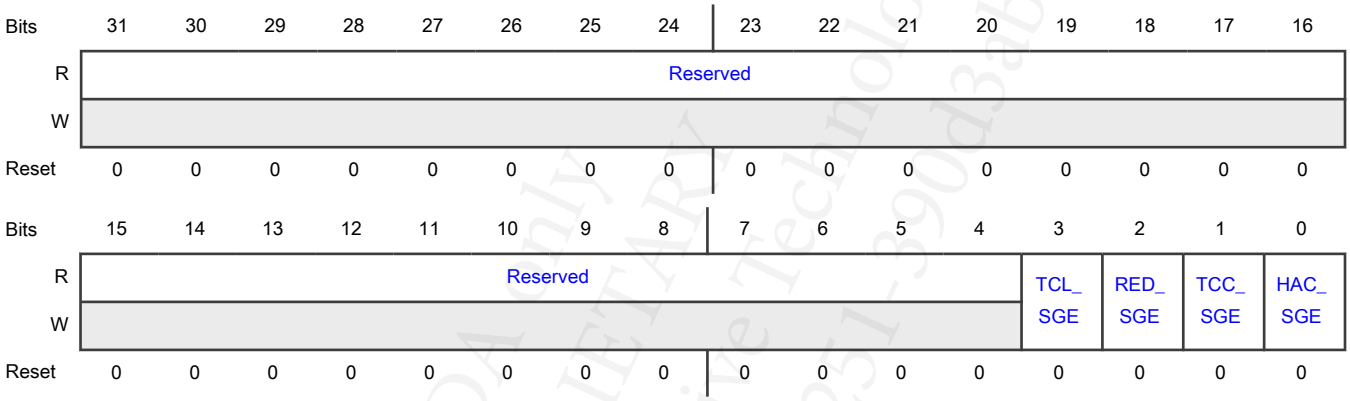
Offset

Register	Offset
CQISGE	118h

Function

This register enables and disables the generation of interrupts to host software. When a bit is set ('1') and the corresponding bit in CQIS is set, then an interrupt is generated. Interrupt sources that are disabled (‘0’) are still indicated in the CQIS register. This register is bit-index matched to CQIS register.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 TCL_SGE	Task cleared signal enable When set and CQIS[TCL] is asserted, the CQE generates an interrupt.
2 RED_SGE	Response error detected signal enable When set and CQIS[RED] is asserted, the CQE generates an interrupt.
1 TCC_SGE	Task complete signal enable When set and CQIS[TCC] is asserted, the CQE generates an interrupt.
0 HAC_SGE	Halt complete signal enable When set and CQIS[HAC] is asserted, the CQE generates an interrupt.

Provided under NDA only
CONTAINS PROPRIETARY
Bittium C-D Automotive Technology Co., Ltd.
c8462e54-8f81-48a8-b251-590d3ab62411

39.4.1.40 Command Queuing Interrupt Coalescing (CQIC)

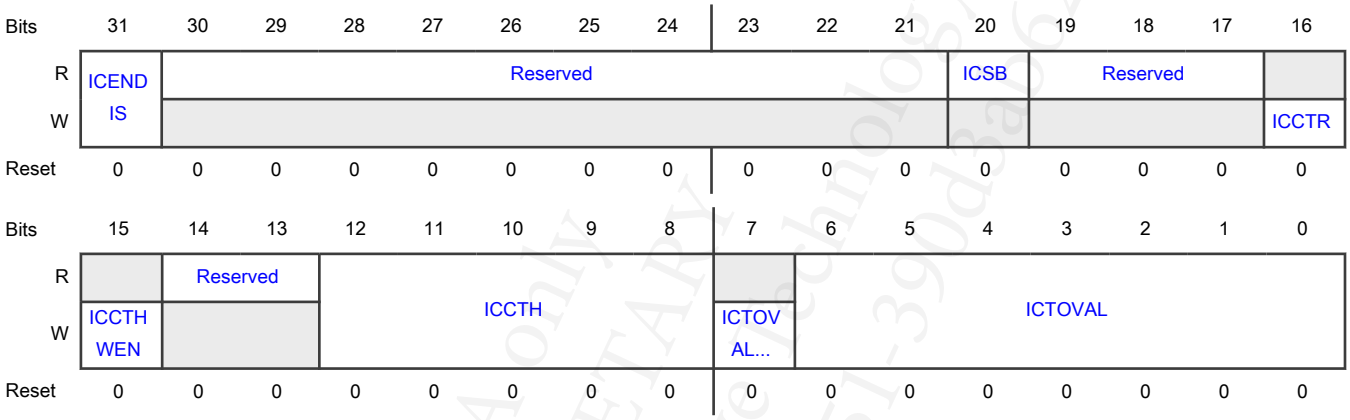
Offset

Register	Offset
CQIC	11Ch

Function

This register controls the interrupt coalescing feature.

Diagram



Fields

Field	Function
31 ICENDIS	Interrupt coalescing enable/disable When host driver writes ‘1’, the interrupt coalescing timer and counter are reset.
30-21 —	Reserved
20 ICSB	Interrupt coalescing status This bit indicates to software whether any tasks (with INT=0) have completed and counted towards interrupt coalescing (for example, ICSB is set if and only if IC counter > 0). 0b - No task completions have occurred since last counter reset (IC counter =0) 1b - At least one task completion has been counted (IC counter >0)
19-17 —	Reserved
16 ICCTR	Counter and timer reset When host driver writes ‘1’, the interrupt coalescing timer and counter are reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 ICCTHWEN	Interrupt coalescing counter threshold write enable When software writes ‘1’, the value ICCTH is updated with the contents written at the same cycle. When software writes ‘0’, the value in ICCTH is not updated. **NOTE** Write operations to ICCTH are only allowed when the task queue is empty.
14-13 —	Reserved
12-8 ICCTH	Interrupt coalescing counter threshold Software uses this field to configure the number of task completions (only tasks with INT=0 in the Task Descriptor) which are required in order to generate an interrupt. Counter Operation: As data transfer tasks with INT=0 complete, they are counted by CQE. The counter is reset by software during the interrupt service routine. The counter stops counting when it reaches the value configured in ICCTH. The maximum allowed value is 31. **NOTE** When ICCTH is 0, task completions are not counted, and counting-based interrupts are not generated. In order to write to this field, the ICCTHWEN field must be set at the same write operation.
7 ICTOVALWEN	Interrupt coalescing timeout value write enable When software writes ‘1’, the value ICTOVAL is updated with the contents written at the same cycle. When software writes ‘0’, the value in ICTOVAL is not updated. **NOTE** Write operations to ICTOVAL are only allowed when the task queue is empty.
6-0 ICTOVAL	Interrupt coalescing timeout value Software uses this field to configure the maximum time allowed between the completion of a task on the bus and the generation of an interrupt. Timer Operation: The timer is reset by software during the interrupt service routine. It starts running when a data transfer task with INT=0 is completed, after the timer was reset. When the timer reaches the value configured in ICTOVAL field it generates an interrupt and stops. The timer’s unit is equal to 1024 clock periods of the clock whose frequency is specified in the Internal Timer Clock Frequency field CQCAP register. The minimum value is 01h (1024 clock periods) and the maximum value is 7Fh (127*1024 clock periods).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	For example, a CQCAP field value of 0 indicates a 19.2 MHz clock frequency (period = 52.08 ns). If the setting in ICTOVAL is 10h, the calculated polling period is 16*1024*52.08 ns= 853.33 us **NOTE** When ICTOVAL is 0, the timer is not running, and timer-based interrupts are not generated. In order to write to this field, the ICTOVALWEN bit must be set at the same write operation.

39.4.1.41 Command Queuing Task Descriptor List Base Address (CQTDLBA)

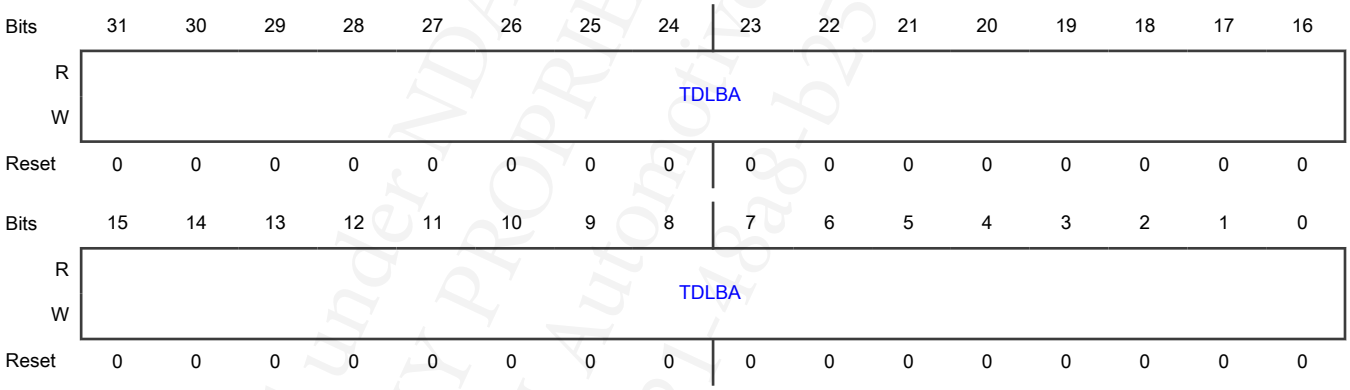
Offset

Register	Offset
CQTDLBA	120h

Function

This register is used for configuring the lower 32 bits of the byte address of the head of the Task Descriptor List in the host memory.

Diagram



Fields

Field	Function
31-0 TDLBA	Task descriptor list base address This register stores the LSB bits (bits 31:0) of the byte address of the head of the Task Descriptor List in system memory. The size of the task descriptor list is 32 * (Task Descriptor size + Transfer Descriptor size) as configured by Host driver. This address shall be set on 1 KByte boundary: The lower 10 bits of this register shall be set to 0 by software and shall be ignored by CQE.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ec46e5-4821-b251-90d3ab624c

39.4.1.42 Command Queuing Task Descriptor List Base Address Upper 32 Bits (CQTDLBAU)

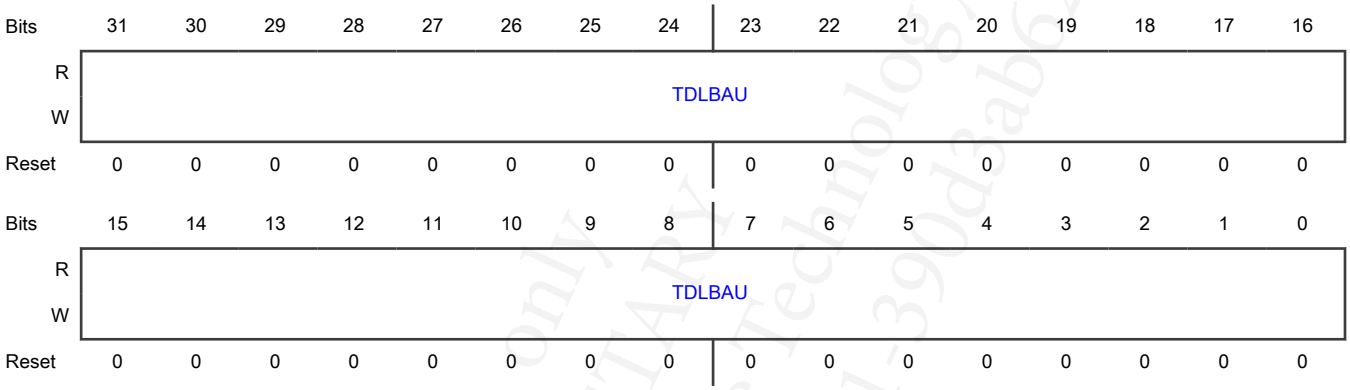
Offset

Register	Offset
CQTDLBAU	124h

Function

This register is used for configuring the upper 32 bits of the byte address of the head of the Task Descriptor List in the host memory.

Diagram



Fields

Field	Function
31-0 TDLBAU	Task descriptor list base address This register stores the MSB bits (bits 63:32) of the byte address of the head of the Task Descriptor List in system memory. The size of the task descriptor list is 32 * (Task Descriptor size + Transfer Descriptor size) as configured by Host driver. This register is reserved when using 32-bit addressing mode.

39.4.1.43 Command Queuing Task Doorbell (CQTDBR)

Offset

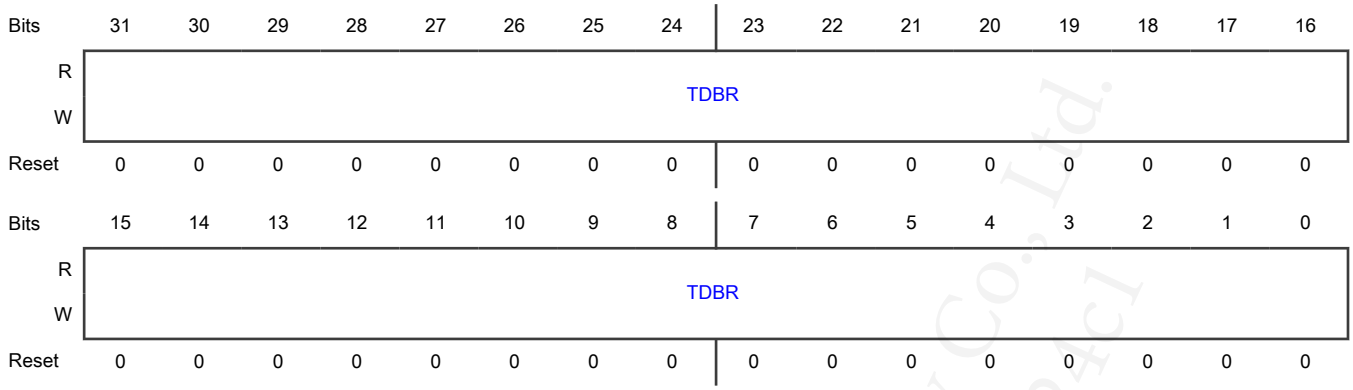
Register	Offset
CQTDBR	128h

Function

Using this register, software triggers CQE to process a new task.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-482-4888-b251-390d3ab6240d

Diagram



Fields

Field	Function
31-0 TDBR	Task doorbell Software configures TDLBA and TDLBAU, and enable CQE in CQCFG before using this register. Writing 1 to bit n of this register triggers CQE to start processing the task encoded in slot n of the TDL. CQE always processes tasks in-order according to the order submitted to the list by CQTDBR write transactions. CQE processes Data Transfer tasks by reading the Task Descriptor and sending QUEUED_TASK_PARAMS (CMD44) and QUEUED_TASK_ADDRESS (CMD45) commands to the device. CQE processes DCMD tasks (in slot #31, when enabled) by reading the Task Descriptor, and generating the command encoded by its index and argument. The corresponding bit is cleared to '0' by CQE in one of the following events: 1. When a task execution is completed (with success or error) 2. The task is cleared using CQTCLR register 3. All tasks are cleared using CQCTL register 4. CQE is disabled using CQCFG register Software may initiate multiple tasks at the same time (batch submission) by writing 1 to multiple bits of this register in the same transaction. In the case of batch submission: CQE shall process the tasks in order of the task index, starting with the lowest index. If one or more tasks in the batch are marked with QBR, the ordering of execution will be based on said processing order. Writing 0 by software shall have no impact on the hardware, and will not change the value of the register bit.

Preliminary Information
COMPANY CONFIDENTIAL
Beijing CINA Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-b2...-90d3ab624c1

39.4.1.44 Command Queuing Task Completion Notification (CQTCN)

Offset

Register	Offset
CQTCN	12Ch

Function

This register is used by CQE to notify software about completed tasks.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TCN															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TCN															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TCN	Task complete notification CQE shall set bit n of this register (at the same time it clears bit n of CQTDBR) when a task execution is completed (with success or error). When receiving interrupt for task completion, software may read this register to know which tasks have finished. After reading this register, software may clear the relevant bit fields by writing 1 to the corresponding bits.

39.4.1.45 Command Queuing Device Queue Status (CQDQS)

Offset

Register	Offset
CQDQS	130h

Function

This register stores the most recent value of the device's queue status.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DQS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DQS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DQS	Device queue status Every time the Host controller receives a queue status register (QSR) from the device, it updates this register with the response of status command (For example, the chip's queue status).

39.4.1.46 Command Queuing Device Pending Tasks (CQDPT)

Offset

Register	Offset
CQDPT	134h

Function

This register indicates to software which tasks are queued in the device, awaiting execution.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DPT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DPT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e54-821-48a8-b251-39d3ab624c1

Fields

Field	Function
31-0 DPT	Device pending tasks Bit n of this register is set if and only if QUEUED_TASK_PARAMS (CMD44) and QUEUED_TASK_ADDRESS (CMD45) were sent for this specific task and if this task hasn't been executed yet. CQE shall set this bit after receiving a successful response for CMD45. CQE shall clear this bit after the task has completed execution. Software needs to read this register in the task-discard procedure, when the controller is halted, to determine if the task is queued in the device. If the task is queued, the driver sends a CMDQ_TASK_MGMT (CMD48) to the device ordering it to discard the task. Then software clears the task in the CQE. Only then the software orders CQE to resume its operation using CQCTL register.

39.4.1.47 Command Queuing Task Clear (CQTCLR)

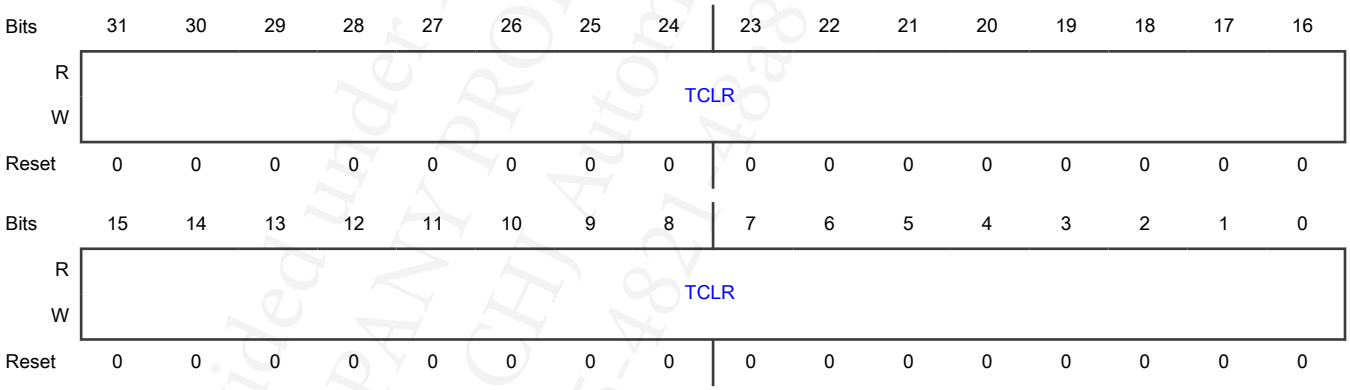
Offset

Register	Offset
CQTCLR	138h

Function

This register is used for removing an outstanding task in the CQE. The register should be used only when CQE is in Halt state.

Diagram



Fields

Field	Function
31-0 TCLR	Task clear Writing 1 to bit n of this register orders CQE to clear a task which software has previously issued. This bit can only be written when CQE is in Halt state as indicated in CQCFG register Halt bit.

Table continues on the next page...

Field	Function
	When software writes '1' to a bit in this register, CQE updates the value to '1', and starts clearing the data structures related to the task. CQE clears the bit fields (sets a value of 0) in CQTCLR and in CQTDBR once clear operation is complete. Software should poll on the CQTCLR until it is cleared to verify clear operation was complete. Writing to this register only clears the task in the CQE and does not have impact on the device. In order to discard the task in the device, host software shall send CMDQ_TASK _MGMT while CQE is still in Halt state. Host driver is not allowed to use this register to clear multiple tasks at the same time. Clearing multiple tasks can be done using CQCTL register. Writing 0 to a register bit shall have no impact.

39.4.1.48 Command Queuing Send Status Configuration 1 (CQSSC1)

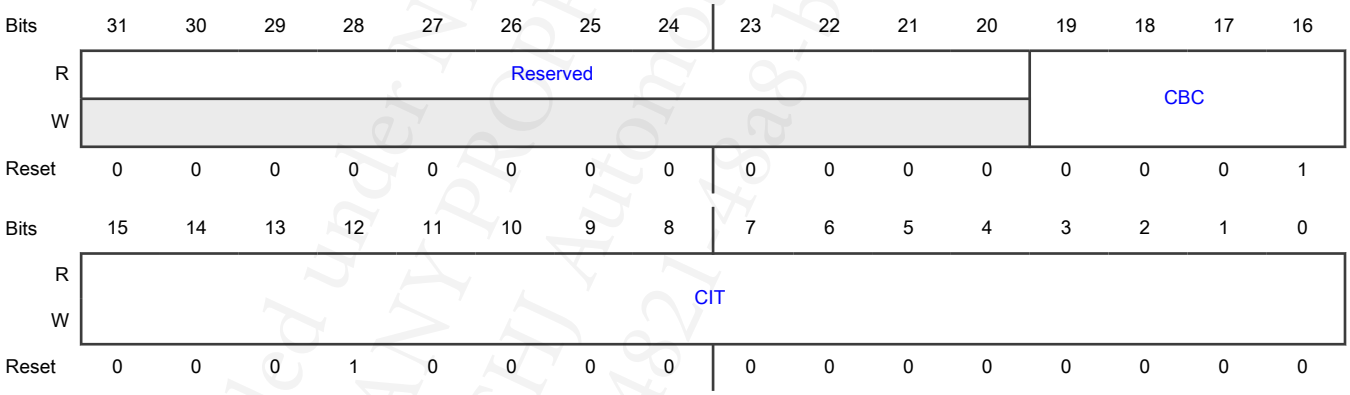
Offset

Register	Offset
CQSSC1	140h

Function

The register controls the when SEND_QUEUE_STATUS commands are sent.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16	Send status command block counter

Table continues on the next page...

Table continued from the previous page...

Field	Function
CBC	This field indicates to CQE when to send SEND_QUEUE_STATUS (CMD13) command to inquire the status of the device's task queue. A value of *n* means CQE sends status command on the CMD line, during the transfer of data block BLOCK_CNT-n, on the data lines, where BLOCK_CNT is the number of blocks in the current transaction. A value of 0 means that SEND_QUEUE_STATUS (CMD13) command is not sent during the transaction. Instead it will be sent only when the data lines are idle. A value of 1 means that STATUS command is to be sent during the last block of the transaction.
15-0 CIT	Send status command idle timer This field indicates to CQE the polling period to use when using periodic SEND_QUEUE_STATUS (CMD13) polling. Periodic polling is used when tasks are pending in the device, but no data transfer is in progress. When a SEND_QUEUE_STATUS response indicating that no task is ready for execution, CQE counts the configured time until it issues the next SEND_QUEUE_STATUS. Timer units are clock periods of the clock whose frequency is specified in the Internal Timer Clock Frequency field CQCAP register. The minimum value is 0001h (1 clock period) and the maximum value is FFFFh (65535 clock periods). Default interval is: 4096 clock periods. For example, a CQCAP field value of 0 indicates a 19.2 MHz clock frequency (period = 52.08 ns). If the setting in CQSSC1.CIT is 1000h, the calculated polling period is 4096*52.08 ns= 213.33 us

39.4.1.49 Command Queuing Send Status Configuration 2 (CQSSC2)

Offset

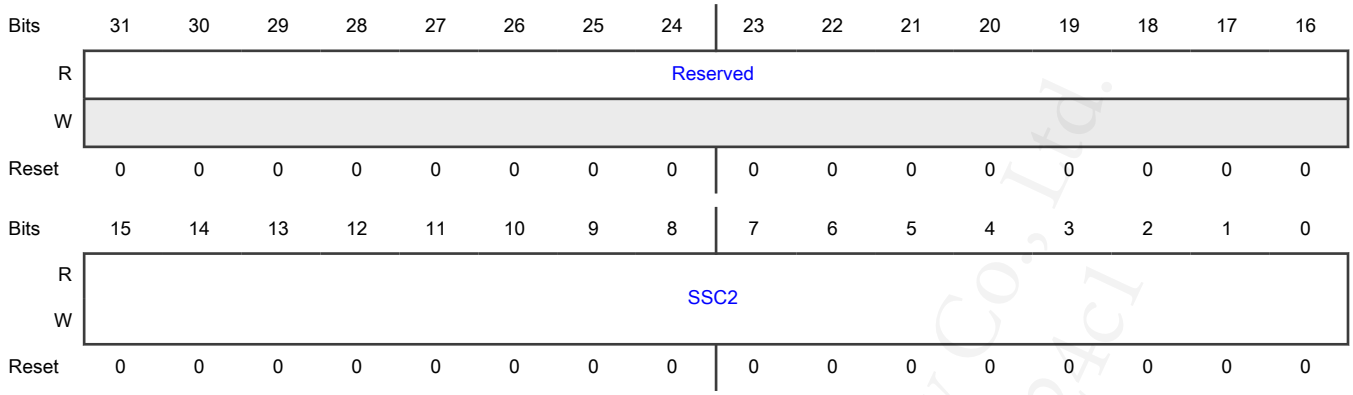
Register	Offset
CQSSC2	144h

Function

This register is used for configuring RCA field in SEND_QUEUE_STATUS command argument.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4621-48a8-92...-590a3b624c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 SSC2	Send queue status RCA This field provides CQE with the contents of the 16-bit RCA field in SEND_QUEUE_ STATUS (CMD13) command argument. CQE copies this field to bits 31:16 of the argument when transmitting SEND_ QUEUE_STATUS (CMD13) command.

39.4.1.50 Command Queuing Command Response for Direct-Command Task (CQCRDCT)

Offset

Register	Offset
CQCRDCT	148h

Function

This register is used for passing the response of a DCMD task to software.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEIU Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CRDCT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CRDCT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CRDCT	Direct command last response This register contains the response of the command generated by the last direct-command (DCMD) task which was sent. CQE updates this register when it receives the response for a DCMD task. This register is considered valid only after bit 31 of CQTDBR register is cleared by CQE.

39.4.1.51 Command Queuing Response Mode Error Mask (CQRMEM)

Offset

Register	Offset
CQRMEM	150h

Function

This register controls the generation of Response Error Detection (RED) interrupt.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RMEM															
W																
Reset	1	1	1	1	1	1	0	1	1	1	1	1	1	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RMEM															
W																
Reset	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e54-821-488-b251-90d3ab624c1

Fields

Field	Function
31-0 RMEM	Response mode error mask This bit is used as in interrupt mask on the device status filed which is received in R1/R1b responses. The reset value of this register is set to trigger an interrupt on all “Error” type bits in the device status, as defined in 6.13. **NOTE** Responses to CMD13 (SQS) encode the QSR, so they are ignored by this logic. 00000000000000000000000000000000b - When a R1/R1b response is received, bit i in the device status is ignored 00000000000000000000000000000001b - When a R1/R1b response is received, with bit i in the device status set, a RED interrupt is generated

39.4.1.52 Command Queuing Task Error Information (CQTERRI)

Offset

Register	Offset
CQTERRI	154h

Function

This register is updated by CQE when an error occurs on data or command related to a task activity.

When such error is detected by CQE or indicated by the e•MMC controller CQE stores in CQTERRI the task IDs and the command indices of the commands which were executed on the command line and data lines when the error occurred.

Software is expected to use this information in the error recovery procedure. See B.2.8 for more details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DTEFV	Reserved		DTETID					Reserved		DTECI					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RMEFV	Reserved		RMETID					Reserved		RMECI					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHD Automotive Technology Co., Ltd.
c8c162e5-4821-48a2-9251-390d3ab624a1

Fields

Field	Function
31 DTEFV	Data transfer error fields valid This bit is updated when an error is detected by CQE, or indicated by e•MMC controller. If a data transfer is in progress when the error is detected/indicated, the bit is set to 1. If a no data transfer is in progress when the error is detected/indicated, the bit is cleared to 0.
30-29 —	Reserved
28-24 DTETID	Data transfer error task ID This field indicates the ID of the task which was executed on the data lines when an error occurred. The field is updated if a data transfer is in progress when an error is detected by CQE, or indicated by e•MMC controller.
23-22 —	Reserved
21-16 DTECI	Data transfer error command index This field indicates the index of the command which was executed on the data lines when an error occurred. The index shall be set to EXECUTE_READ_TASK (CMD46) or EXECUTE_WRITE_TASK (CMD47) according to the data direction. The field is updated if a data transfer is in progress when an error is detected by CQE, or indicated by e•MMC controller.
15 RMEFV	Response mode error fields valid This bit is updated when an error is detected by CQE, or indicated by e•MMC controller. If a command transaction is in progress when the error is detected/indicated, the bit is set to 1. If a no command transaction is in progress when the error is detected/indicated, the bit is cleared to 0.
14-13 —	Reserved
12-8 RMETID	Response mode error task ID This field indicates the index of the command which was executed on the command line when an error occurred. The field is updated if a command transaction is in progress when an error is detected by CQE, or indicated by e•MMC controller.
7-6 —	Reserved
5-0	Response mode error command index

Table continues on the next page...

Table continued from the previous page...

Field	Function
RMECI	This field indicates the index of the command which was executed on the command line when an error occurred. The field is updated if a command transaction is in progress when an error is detected by CQE, or indicated by e•MMC controller.

39.4.1.53 Command Queuing Command Response Index (CQCRI)

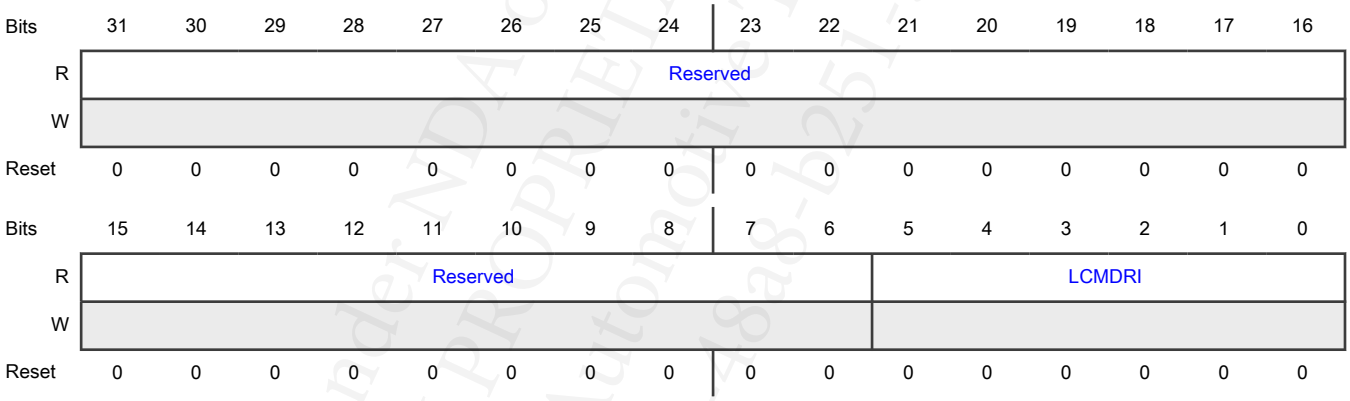
Offset

Register	Offset
CQCRI	158h

Function

This register stores the index of the last received command response.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 LCMDRI	Last command response index This field stores the index of the last received command response. CQE shall update the value every time a command response is received..

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c62e5-4821-18a8-b251-390d3ab624c1

39.4.1.54 Command Queuing Command Response Argument (CQCRA)

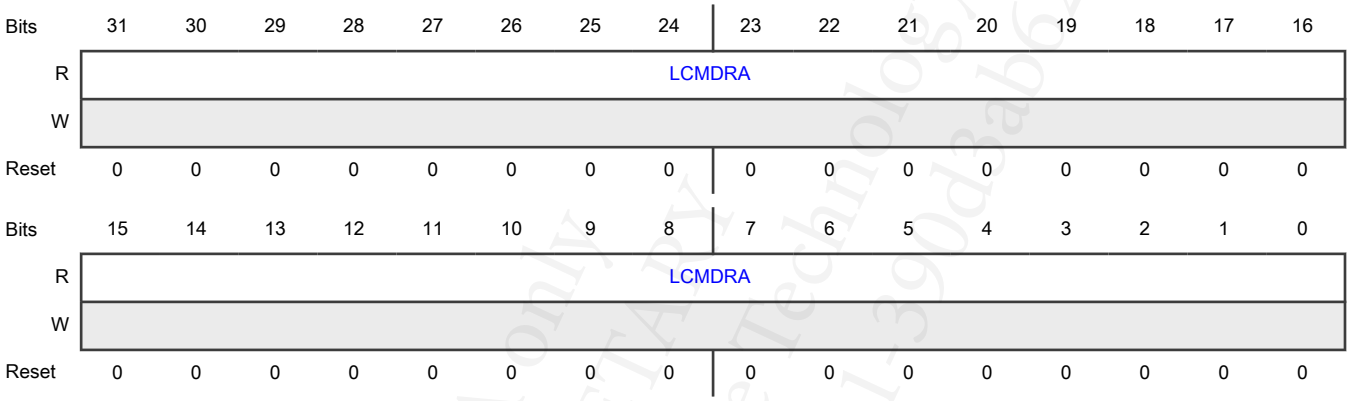
Offset

Register	Offset
CQCRA	15Ch

Function

This register stores the argument of the last received command response.

Diagram



Fields

Field	Function
31-0 LCMDRA	Last command response argument This field stores the argument of the last received command. CQE shall update the value every time a command response is received.

39.5 Functional description

The following sections provide a brief functional description of the major system blocks, including the data buffer, DMA AXI interface, register bank as well as IP Bus interface, dual-port memory wrapper, data/command controller, clock and reset manager, and clock generator.

39.5.1 Data buffer

The uSDHC module uses one configurable data buffer to transfer data between the system bus (IP bus or advanced extensible interface (AXI) bus) and the SD card in an optimized manner, maximizing throughput between the two clock domains (IP peripheral clock and the master clock). The buffer is used as a temporary storage for transferring data between the host system and the card. The watermark levels for read and write are both configurable and can range between 1 to 128 words. When using internal DMA mode, The watermark level for read should be configured to 16 and watermark level for write should be configured to integral multiple of 16 to get max AXI bus throughput. The next figure provides the uSDHC buffer scheme.

NXP Semiconductors NDA only
BLUE CHIP COMPANY PROPRIETARY
c8c6e5-4a21-4848-b251-390da3ab624c1
Beijing Automotive Technology Co., Ltd.

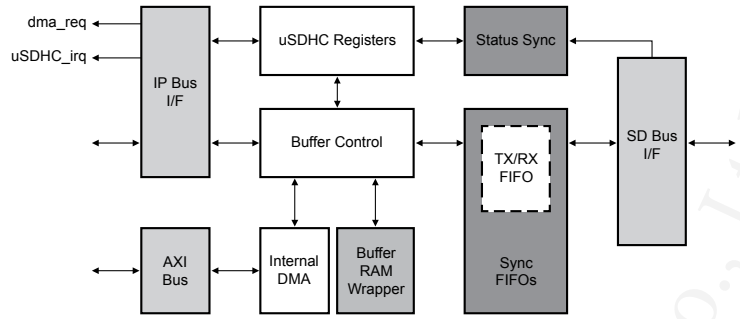


Figure 209. uSDHC buffer scheme

Here are 2 transfer modes to access the data buffer:

- CPU polling mode:
  - For a host-read operation, when the number of words received in the buffer meets or exceeds the RD_WML watermark value, by polling the BRR bit, the host driver can read the Buffer Data Port register to fetch the amount of words set in the RD_WML register from the buffer. The write operation is similar. For more information on the process of writing operation, see Write operation sequence.
- Internal DMA mode (includes simple and advanced DMA accesses):
  - The internal DMA access, either by simple or advanced DMA, is over the AXI bus.

For a read operation, when there are more words in the buffer than the amount set in the RD_WML register, the internal DMA starts fetching data over the AXI bus.

The Write operation functions in a similar manner—sequential and contiguous access is necessary to ensure that the pointer address value is correct. Random or skipped access is not possible. The byte order, by reset, is little endian mode. The actual byte order is swapped inside the buffer, according to the endian mode configured by software (see the following figures). For a host write operation, the byte order is swapped after the data is fetched from the buffer and ready to send to the SD Bus. For a host read operation, the byte order is swapped before the data is stored in the buffer.

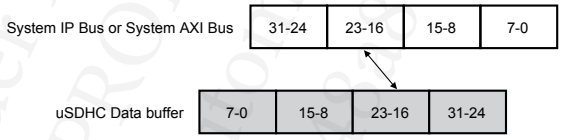


Figure 210. Data swap between system bus and uSDHC data buffer in the byte little endian mode

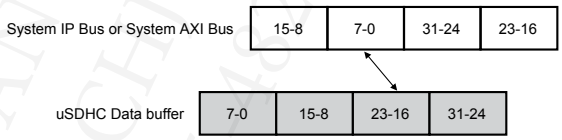


Figure 211. Data swap between system bus and uSDHC data buffer in half word big endian mode

39.5.1.1 Write operation sequence

There are 2 ways to write data into the buffer when the user transfers data to the card:

- Processor core polling through the BWR bit in the Interrupt Status register (interrupt or polling)
- Internal DMA

Provided under NDA ONLY
COMPANY PROPRIETARY
Leijin CHEN (leijin.chen@nxp.com) - 5d3e624c1
c8-162540-1 Thursday, 1 October 2020 Technology Co., Ltd.

When the internal DMA is not used, the DMAEN bit in the Transfer Type register is not set when the command is sent, uSDHC asserts a DMA request when the amount of buffer space exceeds the value set in the WR_WML register and is ready for receiving new data. At the same time, uSDHC sets the BWR bit. The buffer write ready interrupt is generated if it is enabled by software.

When internal DMA is used, uSDHC does not inform the system before all the required number of bytes are transferred (if no error is encountered). When an error occurs during the data transfer, uSDHC aborts the data transfer and abandons the current block. The host driver should read the contents of the DMA System Address register to obtain the starting address of the abandoned data block. If the current data transfer is in multi-block mode, uSDHC does not automatically send CMD12, even though the AC12EN bit in the Transfer Type register is set. The host driver sends CMD12 in this scenario and restarts the write operation from that address. It is recommended that a software reset for Data be applied before the transfer is restarted.

The uSDHC module does not start data transmission until the number of words set in the WR_WML register can be held in the buffer. If the buffer is empty and the host system does not write data in time, uSDHC stops the CLK to avoid the data buffer underrun situation.

39.5.1.2 Read operation sequence

There are 2 ways to read data from the buffer when the user transfers data to the card:

- Processor core polling through the BRR bit in the Interrupt Status register (interrupt or polling)
- Internal DMA

When internal DMA is not used (DMAEN bit in Transfer Type register is not set when the command is sent), uSDHC asserts a DMA request when the amount of data exceeds the value set in the RD_WML register, which is available and ready for system fetching data. At the same time, uSDHC sets the BRR bit. The buffer read ready interrupt is generated if it is enabled by the software.

When internal DMA is used, uSDHC does not inform the system before all the required number of bytes are transferred (if no error is encountered). When an error occurs during the data transfer, uSDHC aborts the data transfer and abandons the current block. The host driver should read the content of the DMA System Address register to get the starting address of the abandoned data block. If the current data transfer is in multi-block mode, uSDHC does not automatically send CMD12, even though the AC12EN bit in the Transfer Type register is set. The host driver sends CMD12 in this scenario and restarts the read operation from that address. It is recommended that a software reset for data be applied before the transfer is restarted.

For any write transfer mode, uSDHC does not start data transmission until the number of words set in the RD_WML register are in buffer. If the buffer is full and the host system does not read data in time, uSDHC stops the CLK to avoid the data buffer overrun situation.

39.5.1.3 Data buffer and block size

The user needs to know the buffer size for the buffer operation during a data transfer to utilize it in the most optimized way. In the uSDHC module, the only data buffer can hold up to 256 words (32-bit) and the watermark levels for write and read can be configured accordingly. The host driver may configure the value according to the system situation and requirement.

During a multi-block data transfer, the block length can be set to any value between 1 and 4096 bytes, satisfying the requirements of the external card. The only restriction is from the external card, which can be limited in size or support of a partial block access (which is not the integer times of 512 bytes).

As uSDHC treats each block individually, for block sizes which are not multiples of four (not word-aligned), stuffed bytes are required at the end of each block. For example, if the block size is 7 bytes and there are 12 blocks to write, the system side must write twice for each block. For each block, the ending byte is abandoned by uSDHC because it only sends 7 bytes to the card and picks data from the following system write, resulting in 24 beats of write access in total.

39.5.1.4 Dividing large data transfer

This SDIO command CMD53 definition limits the maximum data size of data transfers according to the following formula:

Max data size = Block size x Block count

The length of a multiple block transfer needs to be in block size units. If the total data length cannot be divided evenly into a multiple of the block size, then there are two ways to transfer the data. These two ways depend on the function and the card design. Option

1 is for the host driver to split the transaction. The remainder of the block size data is then transferred by using a single block command at the end. Option 2 is to add dummy data in the last block to fill the block size. For option 2, the card must manage the removal of the dummy data.

See the figure below for an example showing the division of large data transfers, assuming a kind of WLAN SDIO card that only supports a block size of up to 64 bytes. Although uSDHC supports a block size of up to 4096 bytes, the SDIO can only accept a block size of less than 64 bytes, so the data must be divided (see the example below).

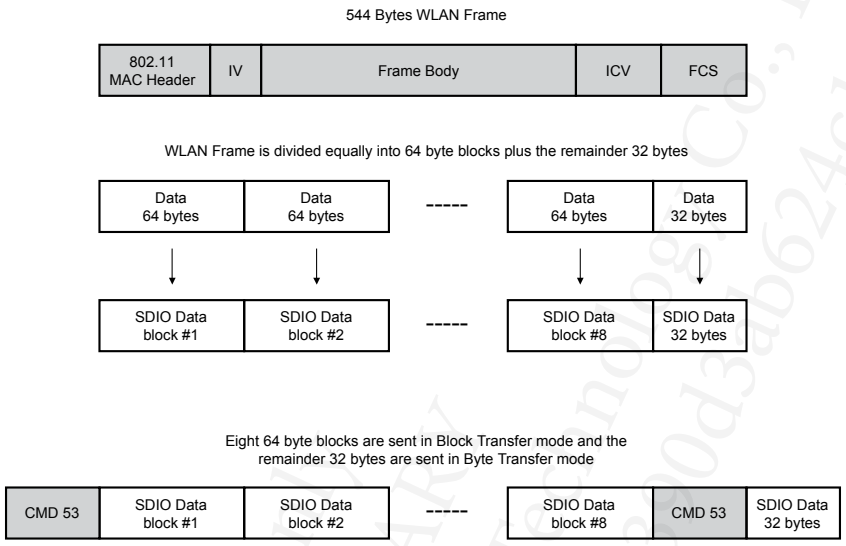


Figure 212. Example for dividing large data transfers

39.5.2 DMA AXI interface

The internal DMA implements a DMA engine and the AXI master. When the internal DMA is enabled, the uSDHC_dreq_b is not asserted during the transfer, but the BWR and BRR bits are set if the BWRSEN and BRRSEN bits have been set in the Interrupt Status Enable register. See the figure below for an illustration of the DMA AXI interface block.

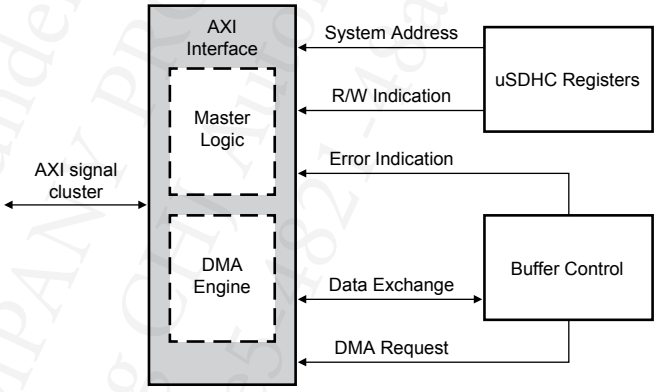


Figure 213. DMA AXI interface block

39.5.2.1 Internal DMA request

If the watermark level requirement is met in data transfer or if the last data of current block is ready in the data buffer, and the Internal DMA is enabled, the data buffer block sends a DMA request to AXI interface. The delay in response from the internal DMA engine depends on the system AXI bus loading and the priority assigned to uSDHC. The DMA engine does not respond to the request during its burst transfer, but is ready to serve as soon as the burst is over. The data buffer de-asserts

the request if the data buffer space (for write) or bytes in data buffer is smaller than the watermark level. Upon access to the buffer by internal DMA, the data buffer updates its internal buffer pointer, and when the watermark level is satisfied or the last data of the current block is ready in the data buffer, another DMA request is sent.

The data transfer is in the block unit, and the subsequent watermark level is always set as the remaining number of words. For instance, for a multi block data read with each block size of 31 bytes, and the watermark level set to six words. After the first burst transfer, if there are more than two words in the buffer, which might contain some data of the next block), another DMA request is sent. This is because the remaining number of words to send for the current block is (31 - 6 * 4) / 4 = 2. The uSDHC module reads two words out of the buffer, with seven valid bytes and one stuffed byte.

39.5.2.2 AXI master interface

It is possible that the internal AXI DMA engine could fail during the data transfer. Upon detection of an AXI bus error during DMA transfer, the DMA engine stops the transfer and goes to the idle state. At that point, the internal data buffer stops receiving incoming data and sending out data. The DMAE bit in the Interrupt Status register is generated to host CPU to report a bus error condition.

After the DMAE interrupt is received, the software sends a CMD12 to abort the current transfer and read the DS_ADDR bits of the DMA System Address register to get the starting address of the corrupted block. After the DMA error is fixed, the software should apply a data reset and restart the transfer from this address to recover the corrupted block. DMA operation resumes when the interrupt is serviced by the software.

39.5.2.3 ADMA engine

In the SD host controller standard, a new DMA transfer algorithm called the Advanced DMA (ADMA) is defined. For simple DMA, after the page boundary is reached, a DMA interrupt is generated and the new system address is programmed by the host driver. The ADMA defines the programmable descriptor table in the system memory. The host driver can calculate the system address at the page boundary and program the descriptor table before executing ADMA. It reduces the frequency of interrupts to the host system. Therefore, higher speed DMA transfers could be realized because the host MCU intervention is not be needed during long DMA-based data transfers.

There are two types of ADMA in host controller: ADMA1 and ADMA2. ADMA1 can support data transfer of 4KB aligned data in system memory, and ADMA2 improves the restriction so that the data of any location and any size can be transferred in system memory. Their formats of Descriptor Table are different.

ADMA can recognize all kinds of descriptors defined in SD host controller Standard, and if the "End" flag is detected in the descriptor, ADMA stops after this descriptor is processed.

39.5.2.3.1 ADMA concept and descriptor format

ADMA1 includes the following descriptors:

- Valid/invalid descriptor
- Nop descriptor
- Set data length descriptor
- Set data address descriptor
- Link descriptor
- Interrupt flag and end flag in descriptor

ADMA2 includes the following descriptors:

- Valid/invalid descriptor
- Nop descriptor
- Rsv descriptor
- Set data length and address descriptor
- Link descriptor

- Interrupt flag and end flag in descriptor

ADMA starts read/write operation after it reaches the tran state, using the data length and data address analyzed from most recent descriptor(s).

For ADMA1, the valid data length descriptor is the last set type descriptor before the tran type descriptor. Every tran type triggers a transfer, and the transfer data length is extracted from the most recent set type descriptor. If there is no set type descriptor after the previous Trans descriptor, the data length is the value for previous transfer, or 0 if no set descriptor is ever met.

For ADMA2, the tran type descriptor contains both data length and transfer data address, so only a tran type descriptor can start a data transfer.

See the figure below for the format of the descriptor table for ADMA1.

Address/ Page Field		Address/ Page Field		Attribute Field					
31	12	11	6	5	4	3	2	1	0
Address or Data Length		000000		Act 2	Act 1	0	Int	End	Valid

Act 2	Act1	Symbol	Comment	31- 28	27- 12
0	0	Nop	No Operation	Don't Care	
0	1	Set	Set Data Length	0000	Data Length
1	0	Tran	Transfer Data	Data Address	
1	1	Link	Link Descriptor	Descriptor Address	

Valid	Valid = 1 indicates this line of descriptor is effective. If Valid = 0 generate ADMA Error Interrupt and stop ADMA.
End	End = 1 indicates current descriptor is the ending one.
Int	Int = 1 generates DMA Interrupt when this descriptor is processed.

Figure 214. Format of the ADMA1 descriptor table

See the figure below for the concept and access method of the descriptor table for ADMA1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a9-b251-29903a2b24c1

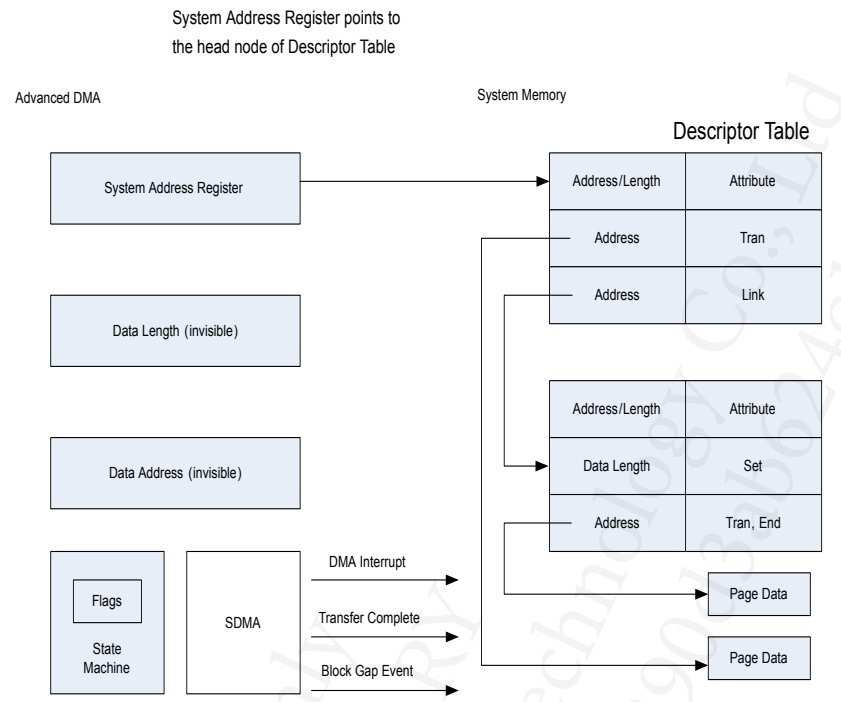


Figure 215. Concept and access method of the ADMA1 descriptor table

The figure below explains the ADMA2 format. ADMA2 deals with the lower 32-bit first, and then the higher 32-bit. If the 'Valid' flag of descriptor is 0, it ignores the higher 32-bit. The Address field should be set to word aligned (lower 2-bit is always set to 0). Data length is in byte unit.

Address Field		Length		Reserved		Attribute Field					
63	32	31	16	15	06	05	04	03	02	01	00
32-bit Address		16-bit length		0000000000		Act 2	Act 1	0	Int	End	Valid

Act 2	Act1	Symbol	Comment	Operation
0	0	Nop	No Operation	Don't Care
0	1	Rsv	Reserved	Same as Nop. Read this line and go to next one
1	0	Tran	Transfer Data	Transfer data with address and length set in this descriptor line
1	1	Link	Link Descriptor	Link to another descriptor

Valid	Valid = 1 indicates this line of descriptor is effective. If Valid = 0 generate ADMA Error Interrupt and stop ADMA.
End	End = 1 indicates current descriptor is the ending one.
Int	Int = 1 generates DMA Interrupt when this descriptor is processed.

Figure 216. Format of the ADMA2 descriptor table

See the figure below for the concept and access method of the descriptor table for ADMA2.

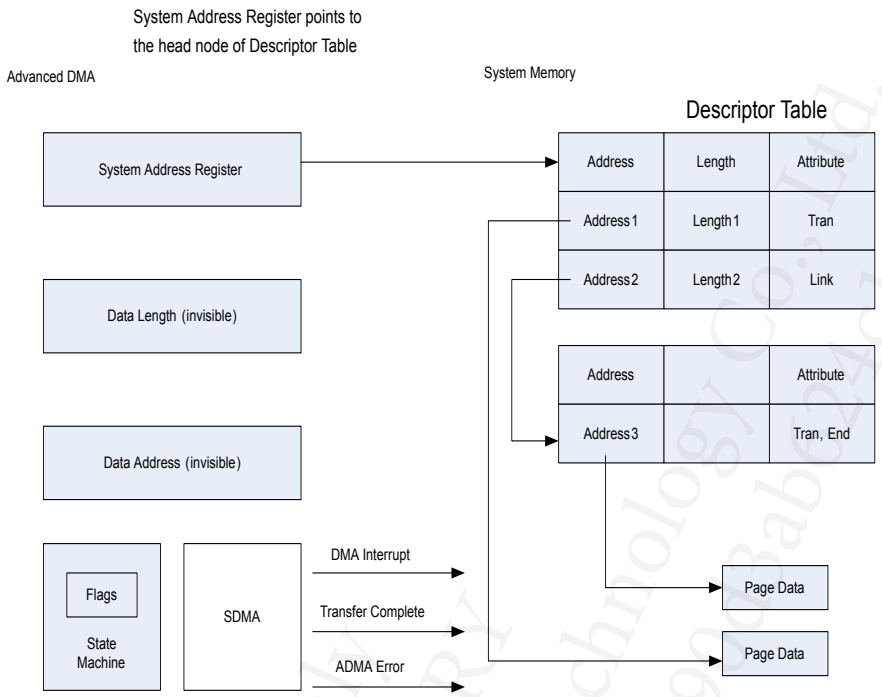


Figure 217. Concept and access method of the ADMA2 descriptor table

39.5.2.3.2 ADMA interrupt

If the interrupt flag descriptor is set, ADMA generates an interrupt according to various types of descriptors.

For ADMA1:

- Set type of descriptor: interrupt is generated when data length is set.
- Tran type descriptor: interrupt is generated when this transfer is complete.
- Link type of descriptor: interrupt is generated when new descriptor address is set.
- Nop type of descriptor: interrupt is generated just after this descriptor is fetched.

For ADMA2:

- Tran type of descriptor: interrupt is generated when this transfer is complete.
- Link type of descriptor: interrupt is generated when new descriptor address is set.
- Nop/Rsv type of descriptor: interrupt is generated just after this descriptor is fetched.

39.5.2.3.3 ADMA error

The ADMA stops whenever an error is encountered. These errors include:

- Fetching descriptor error
- AXI response error
- Data length mismatch error

An ADMA descriptor error is generated when it fails to detect a "valid" flag in the descriptor. If an ADMA descriptor error occurs, the interrupt is not generated even if the "Interrupt" flag of this descriptor is set.

When BLKCNTEN bit is set, data length set in buffer must be equal to the whole data length set in descriptor nodes, otherwise data length mismatch error is generated.

When BLKCNTEN bit is not set, then the whole data length set in descriptor is a multiple of block lengths; otherwise, when data set in the descriptor nodes are not performed at block boundaries, then data mismatch errors occur.

39.5.3 Register bank with IP bus interface

Register accesses through the IP bus interface are on the register bank. See the figure below for the block diagram.

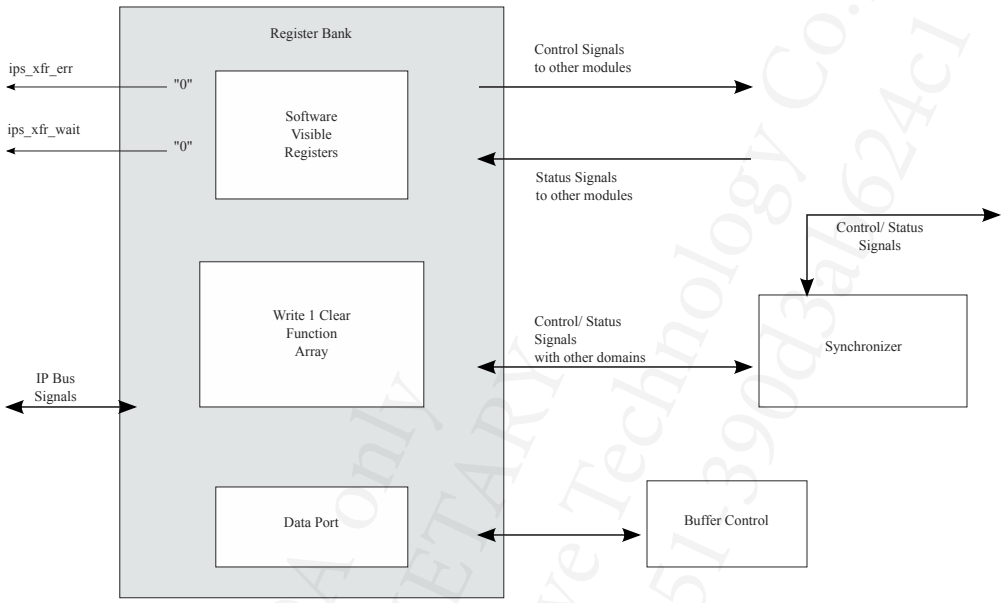


Figure 218. Register bank diagram

Only a 32-bit access is allowed, and no partial read/write is supported; therefore, all accesses are word aligned.

Access to an unimplemented address within the register memory map space does not generate a transfer error.

39.5.3.1 SD protocol unit

The SD protocol unit deals with all SD protocol affairs.

The SD protocol unit performs the following functions:

- Acts as the bridge between the internal buffer and the SD bus
- Sends the command data as well as its argument serially
- Stores the serial response bit stream into corresponding registers
- Detects the bus state on the CMD/DAT lines
- Monitors the interrupt from the SDIO card
- Asserts the read wait signal
- Gates off the SD clock when buffer is announcing danger status
- Detects the write protect state

The SD protocol unit consists of four submodules:

- SD control misc
- Command control

- Data control
- Clock control

39.5.3.2 SD control misc

In the SD control misc unit:

- The card detection (including CD_B and DATA3 for card detection) and card interrupt are implemented.
- The module monitors the signal level on all the eight data lines and the command lines. It directly routes the level values into the register bank.
- The module also detects the Write Protect (WP) line. If WP is active, writes to the register bank are ignored.

39.5.3.3 SD clock control

If the internal data buffer is near full (for read) or near empty (for write), the SD clock must be gated off to avoid buffer over/under-run. This module asserts the gate of the output SD clock to shut the clock off. After the buffer has space (for read) or has data (for write), the clock gate of this module opens, and the SD clock is active again.

39.5.3.4 Command control

The Command Control module deals with the transactions on the CMD line.

See the figure below for an illustration of the structure for the Command CRC shift register.

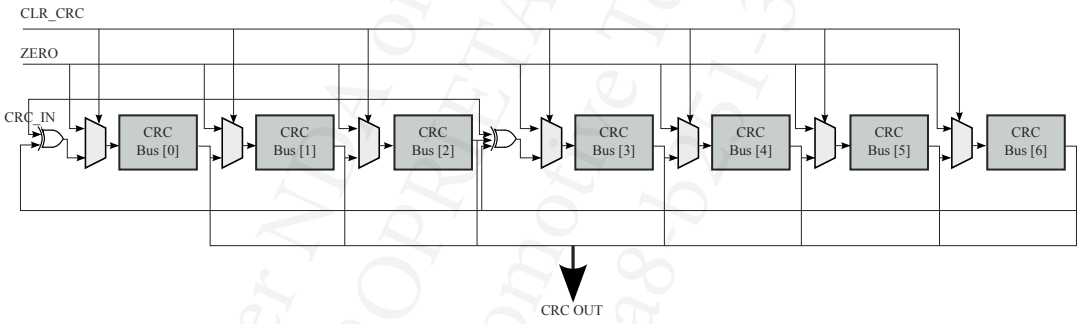


Figure 219. Command CRC shift register

The CRC polynomials for the CMD are as follows:

```
Generator polynomial: G(x) = x^7 + x^3 + 1
M(x) = (first bit) * x^n + (second bit) * x^n-1 +...+ (last bit) * x^0
CRC[6:0] = Remainder [(M(x) * x^7) / G(x)]
```

39.5.3.5 Data control

The data agent deals with the transactions on the eight data lines. Moreover, this module also detects the busy state on the DATA0 line and generates the read wait state by the request from the transceiver.

The CRC polynomials for the data are as follows:

```
Generator polynomial: G(x) = x^16 + x^12 + x^5 +1

M(x) = (first bit) * x^n + (second bit) * x^n-1 +...+ (last bit) * x^0

CRC[15:0] = Remainder [(M(x) * x^16) / G(x)]
```

39.5.4 Clock and reset manager

This module controls all four kinds reset signals within uSDHC:

- Hardware reset
- Software reset for all logic
- Software reset for the data logic
- Software reset for the command logic

All these signals are fed into this module and stable signals are generated inside the module to reset all other modules. The module also gates off all the inside signals.

39.5.5 Clock generator

The clock generator generates the card CLK by peripheral source clock in two stages. The clock divisor can be configured through register SYS_CTRL[SDCLKFS] is for prescaler configuration while the [DVS] is for divisor configuration. Details can be found in the register function description. See the following figure for the structure of the divider. The term "Base" represents the frequency of peripheral source clock.

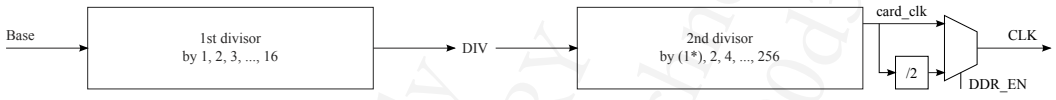


Figure 220. Two stages of the clock divider

The first stage outputs an intermediate clock (DIV) that can be Base, Base/2, Base/3, ..., or Base/16.

The second stage is a prescaler and outputs the actual internal working clock (card_clk). This clock is the driving clock for all the sub modules of the SD protocol unit, and helps in syncing FIFOs (see Figure 209) to synchronize with the data rate from the internal data buffer. The frequency of the clock output from this stage can be DIV, DIV/2, DIV/4,..., or DIV/256. Therefore, the highest frequency of the card_clk is base, and the next highest is Base/2, while the lowest frequency is Base/4096. If the duty cycle of the base clock is 50%, the duty cycle of card_clk is also 50%, even when the compound divisor is an odd value.

NOTE

CLK is different for the SDR and DDR modes.

- In the SDR mode, CLK is equal to the internal working clock (card_clk).

- In the DDR mode, CLK is equal to card_clk/2.

39.5.6 SDIO card interrupt

Information on interrupts in 1-bit mode, interrupts in 4-bit mode, and card interrupt handling are detailed in the sections below.

39.5.6.1 Interrupts in 1-bit mode

In this case, the DATA1 pin provides the interrupt function. An interrupt is asserted by pulling the DATA1 low from the SDIO card, until the interrupt service is finished to clear the interrupt.

39.5.6.2 Interrupt in 4-bit mode

As the interrupt and data line 1 share pin 8 in a 4-bit mode, an interrupt is only sent by the card and recognized by the host during a specific time. This is known as the interrupt period. The uSDHC module only provides sample the level on pin 8 during the interrupt period. At all other times, the host ignores the level on pin 8 and treats it as the data signal. The definition of the interrupt period is different for operations with single block and multiple block data transfers.

In the case of normal single data block transmissions, the interrupt period becomes active two clock cycles after the completion of a data packet. This interrupt period lasts until after the card receives the end bit of the next command that has a data block transfer associated with it.

For multiple block data transfers in a 4-bit mode, there is only a limited period of time that the interrupt period can be active because of the limited period of data line availability between the multiple blocks of data. This requires stricter definition of the interrupt period. For this case, the interrupt period is limited to two clock cycles. This begins two clocks after the end bit of the previous data block. During this 2-clock cycle interrupt period, if an interrupt is pending, the DATA1 line holds low for one clock cycle with the last clock cycle pulling DATA1 high. On completion of the interrupt period, the card releases the DATA1 line into the high Z state. The uSDHC module provides sample of the DATA1 during the interrupt period when the IABG bit in the Protocol Control register is set.

Refer to SDIO Card Specification v1.10 for further information about the SDIO card interrupt.

39.5.6.3 Card interrupt handling

When the CINTIEN bit in the Interrupt Signal Enable Register is set to 0, uSDHC clears the interrupt request to the host system. The host driver should clear this bit before servicing the SDIO interrupt and should set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts.

The SDIO Card Interrupt Status can be cleared by writing 1 to this bit. But as the interrupt source from the SDIO card does not clear, this bit is set again. To clear this bit, it is required to reset the interrupt source from the external card followed by writing 1 to this bit. In a 1-bit mode, uSDHC detects the SDIO interrupt with or without the SD clock (to support wakeup). In a 4-bit mode, the interrupt signal is sampled during the interrupt period, so there are some sample delays between the interrupt signal from the SDIO card and the interrupt to the Host System Interrupt Controller. When the SDIO status is set and the host driver needs to service this interrupt, the SDIO bit in the Interrupt Control Register of SDIO card is cleared. This is required to clear the SDIO interrupt status latched in uSDHC and to stop driving the interrupt signal to the System Interrupt Controller. The host driver must issue a CMD52 to clear the card interrupt. After completion of the card interrupt service, the SDIO Interrupt Status Enable bit is set to 1 and uSDHC starts sampling the interrupt signal again.

See the figure below for an illustration of the SDIO card interrupt scheme and for the sequences of software and hardware events that take place during a card interrupt handling procedure.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.
c8c462e5-4821-48a8-b251-[illegible]b4c1

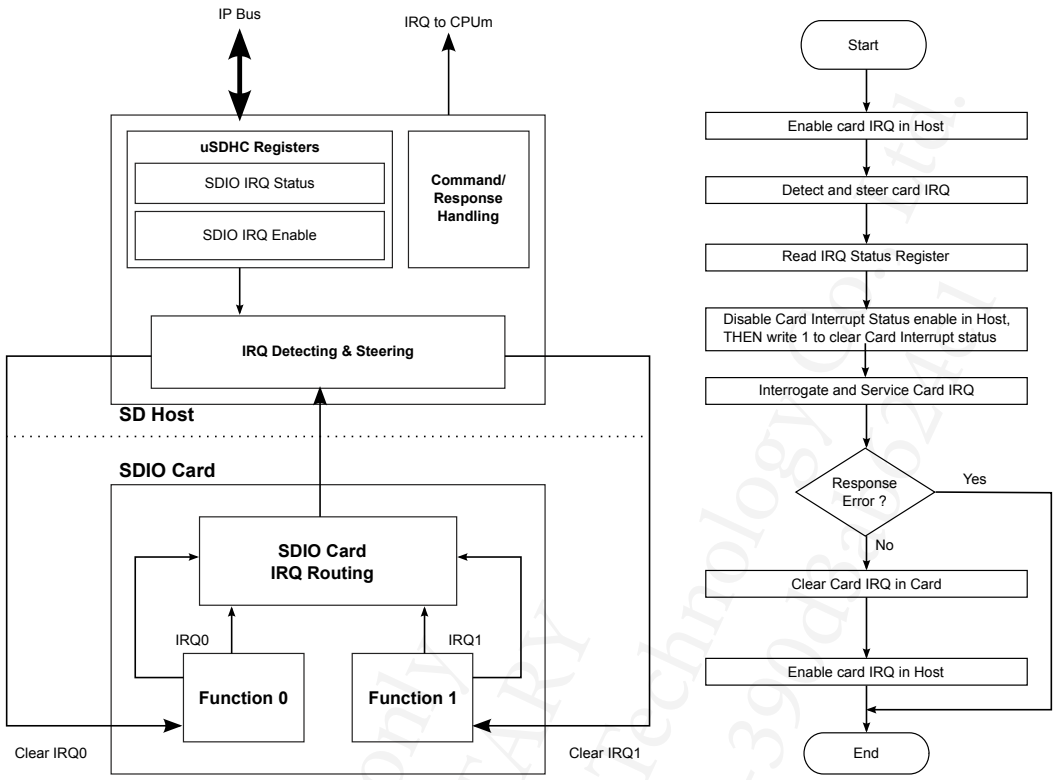


Figure 221. Card interrupt scheme, card interrupt detection, and handling procedure

39.5.7 Card insertion and removal detection

The uSDHC module uses either the DATA3 pin or the CD_B pin to detect card insertion or removal. When there is no card on the MMC/SD bus, the DATA3 is pulled to a low voltage level by default.

When any card is inserted to or removed from the socket, uSDHC detects the logic value changes on the DATA3 pin and generates an interrupt. When the DATA3 pin is not used for card detection (for example, it is implemented in GPIO), the CD_B pin must be connected for card detection. Whether DATA3 is configured for card detection or not, the CD_B pin is always a reference for card detection. Whether the DATA3 pin or the CD_B pin is used to detect card insertion, uSDHC sends an interrupt (if enabled) to inform the Host system that a card is inserted.

39.5.8 Power management and wakeup events

When there is no operation between uSDHC and the card through the SD bus, the user can completely disable the peripheral clock and base clock in the chip-level clock control module to save power. When the user needs to use uSDHC to communicate with the card, it can enable the clock and start the operation.

In some circumstances, when the clocks to uSDHC are disabled, for instance, when the system is in low-power mode, there are some events for which the user needs to enable the clock and handle the event. These events are called wakeup interrupts. The uSDHC module can generate these interrupts even when there are no clocks enabled. The three interrupts that can be used as wakeup events are these:

- Card removal interrupt
- Card insertion interrupt
- Interrupt from SDIO card

The uSDHC module offers a power management feature. By clearing the clock enabled bits in the System Control register, the clocks are gated in the low position to uSDHC. For maximum power saving, the user can disable all the clocks to uSDHC when there is no operation in progress.

These three wakeup events (or wakeup interrupts) can also be used to wakeup the system from low-power modes.

NOTE

To make the interrupt a wakeup event, when all the clocks to uSDHC are disabled or when the entire system is in low power mode, the corresponding wakeup enabled bit needs to be set. Refer to Protocol Control (PROT_CTRL) for more information on the uSDHC Protocol Control register.

39.5.8.1 Setting wakeup events

For uSDHC to respond to a wakeup event, the software must set the respective wakeup enable bit before the CPU enters the sleep mode. Before the software disables the host clock, it should ensure that all the following conditions have been met:

- No read or write transfer is active
- Data and command lines are not active
- No interrupts are pending
- Internal data buffer is empty

39.5.9 MMC fast boot

The Embedded MultiMediaCard (eMMC4.3) specification adds a fast boot feature that requires hardware support. There are two types of fast boot modes in the eMMC4.3 specification: boot operation and alternative boot operation in the eMMC4.3 specification. Each type also has with-acknowledge and without-acknowledge modes.

In the boot operation mode, the master (MultiMediaCard host) can read boot data from the slave (MMC device) by keeping CMD line low after power-on, or sending CMD0 with argument + 0xFFFFFFFA (optional for slave), before issuing CMD1.

NOTE

For the eMMC4.3 card setting, please see the eMMC4.3 specification.

39.5.9.1 Boot operation

If the CMD line is held low for 74 clock cycles and more after power-up before the first command is issued, the slave recognizes that boot mode is being initiated and starts preparing boot data internally.

Within one second after the CMD line goes low, the slave starts to send the first boot data to the master on the DATA line(s). The master must keep the CMD line low to read all of the boot data.

NOTE

For the purposes of this documentation, fast boot is called "normal fast boot mode".

If boot acknowledge is enabled, the slave has to send acknowledge pattern '010' to the master within 50ms after the CMD line goes low. If boot acknowledge is disabled, the slave does not send out acknowledge pattern '010'.

The master can terminate the boot mode with the CMD line high.

The boot operation is terminated when all the contents of the enabled boot data are sent to the master. After the boot operation is executed, the slave gets ready for the CMD1 operation and the master needs to start a normal MMC initialization sequence by sending CMD1.

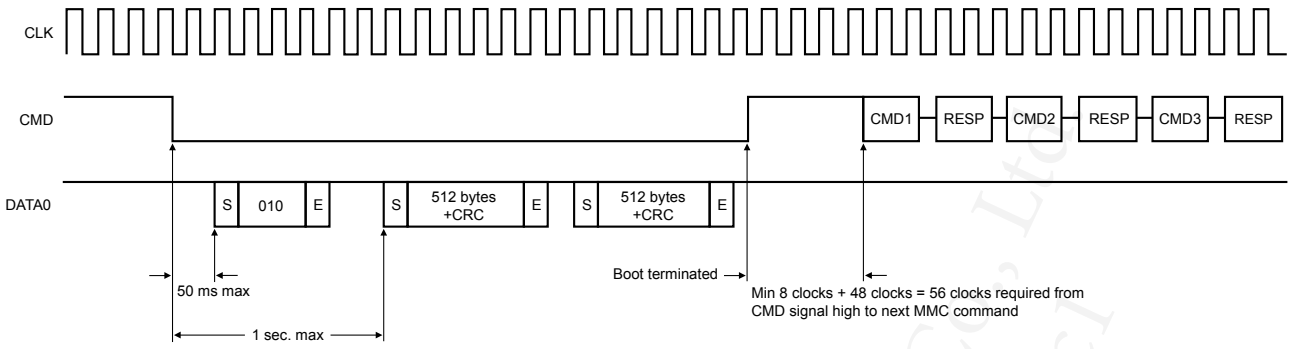


Figure 222. MultiMediaCard state diagram (normal boot mode)

39.5.9.2 Alternative boot operation

This boot function is optional for the device. If bit 0 in the extended CSD byte[228] is set to '1', the device supports the alternative boot operation.

After power-up, if the host issues CMD0 with the argument of 0xFFFFFFFA after 74 clock cycles, before CMD1 is issued or the CMD line goes low, the slave recognizes that boot mode is being initiated and starts preparing boot data internally.

Within one second after CMD0 with the argument of 0xFFFFFFFA is issued, the slave starts to send the first boot data to the master on the DATA line(s).

If boot acknowledge is enabled, the slave must send the acknowledge pattern '010' to the master within 50ms after the CMD0 with the argument of 0xFFFFFFFA is received. If boot acknowledge is disabled, the slave does not send out acknowledge pattern '010'.

The master can terminate the boot mode by issuing CMD0 (Reset).

Boot operation is terminated when all the contents of the enabled boot data are sent to the master. After the boot operation is executed, the slave gets ready for the CMD1 operation and the master needs to start a normal MMC initialization sequence by sending CMD1.

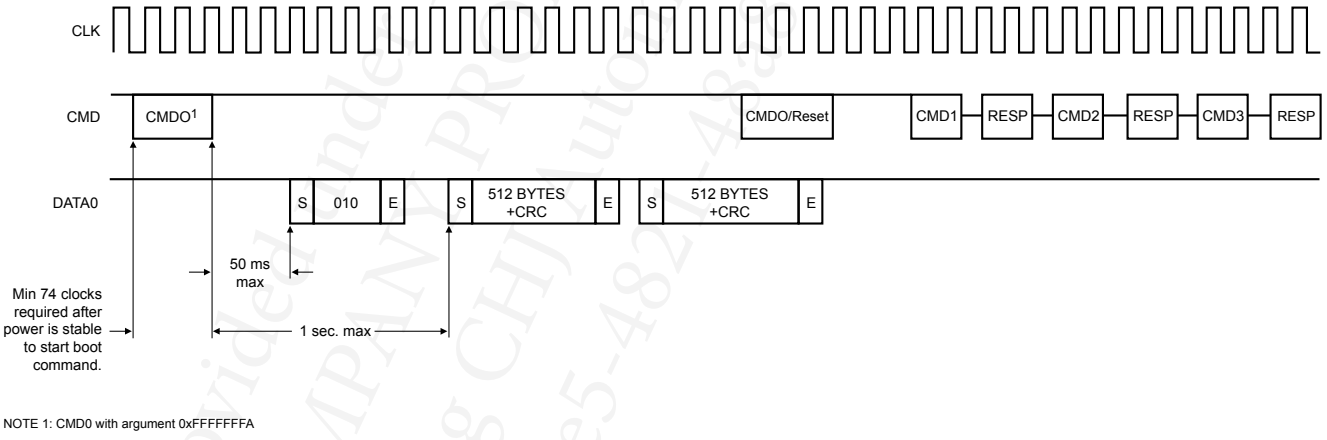


Figure 223. MultiMediaCard state diagram (alternative boot mode)

39.6 Initialization/application of uSDHC

All communication between the system and cards are controlled by the host. The host sends commands of two types: broadcast and addressed (point-to-point).

Broadcast commands are intended for all cards, such as GO_IDLE_STATE, SEND_OP_COND, and ALL_SEND_CID. In the Broadcast mode, all cards are in the open-drain mode to avoid bus contention. Refer to Commands for MMC/SD/SDIO for the commands of bc and bcr categories.

After the broadcast command CMD3 is issued, the cards enter the standby mode. Addressed type commands are used from this point. In this mode, the CMD/DATA I/O pads turns to the push-pull mode to have the driving capability for maximum frequency operation. Refer to Commands for MMC/SD/SDIO for the commands of ac and adtc categories.

A uSDHC FIFO control problem exists with a software reset for a single block read. See MIX_CTRL[DTDSEL] for more information.

39.6.1 Command send & response receive basic operation

Assuming that the data type WORD is an unsigned 32-bit integer, the flow indicated below presents a guideline for sending a command to the card(s):

```
send_command(cmd_index, cmd_arg, other requirements)
{
WORD wCmd; // 32-bit integer to make up the data to write into Transfer Type register, it is
recommended to implement in a bit-field manner
wCmd = (<cmd_index> & 0x3f) << 24; // set the first 8 bits as '00'+<cmd_index>
set CMDTYP, DPSEL, CICEN, CCCEN, RSTTYP, DTDSEL accorind to the command index;
if (internal DMA is used) wCmd |= 0x1;
if (multi-block transfer) {
          set MSBSEL bit;
          if (finite block number) {
                    set BCEN bit;
                    if (auto12 command is to use) set AC12EN bit;
          }
}
write_reg(CMDARG, <cmd_arg>); // configure the command argument
write_reg(XFERTYP, wCmd); // set Transfer Type register as wCmd value to issue the command
}
wait_for_response(cmd_index)
{
while (CC bit in IRQ Status register is not set); // wait until Command Complete bit is set
read IRQ Status register and check if any error bits about Command are set
if (any error bits are set) report error;
write 1 to clear CC bit and all Command Error bits;
}
```

For the sake of simplicity, the function wait_for_response is implemented here by means of polling. For an effective and formal way, the response is usually checked after the Command Complete Interrupt is received. When doing this, make sure that the corresponding interrupt status bits are enabled.

For some scenarios, the response time-out is expected. For instance, after all cards respond to CMD3 and move to the Standby state no response to the Host when CMD2 is sent. The host driver deals with "fake" errors like this with caution.

39.6.2 Card identification mode

When a card is inserted to the socket or the card is reset by the host, the host needs to validate the operation voltage range, identify the cards, request the cards to publish the Relative Card Address (RCA) or to set the RCA for MMC cards.

39.6.2.1 Card detect

See the figure below for a flow diagram showing the detection of MMC, SD, and SDIO cards using uSDHC.

Beijing CU Automotive Technology Co., Ltd
c84625-832c-4848-b251-390daa6241c1
Preliminary Information
COMPANY CONFIDENTIAL

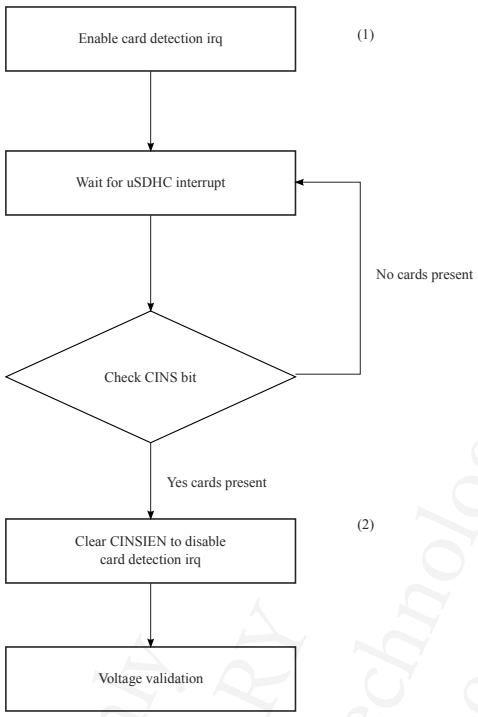


Figure 224. Flow diagram for card detection

Here is the card detect sequence:

- Set the CINSIEN bit to enable card detection interrupt.
- When an interrupt from uSDHC is received, check the CINS bit in the Interrupt Status register to see if it was caused by card insertion.
- Clear the CINSIEN bit to disable the card detection interrupt and ignore all card insertion interrupts afterwards.

39.6.2.2 Reset

The host consists of three types of resets:

- Hardware reset (Card and Host) that is driven by Power On Reset (POR).
- Software reset (Host only) is initiated by the write operation on the RSTD, RSTC, or RSTA bits of the System Control register to reset the data part, command part, or all parts of the host controller, respectively.
- Card reset (Card only): The command, "Go_Idle_State" (CMD0), is the software reset command for all types of MMC cards and SD memory cards. This command sets each card into the idle state regardless of the current card state. For an SD I/O card, CMD52 is used to write an I/O reset in CCCR. The cards are initialized with a default relative card address (RCA=0x0000) and with a default driver stage register setting (lowest speed, highest driving current capability).

After the card is reset, the host needs to validate the voltage range of the card. See the figure below for the software flow to reset both uSDHC and the card.

Provided under NDA only
COMPANY PROPRIETARY
Beijing Futurehive Technology Co., Ltd.
c8c462...1-148a-1251-390d3ab624c1

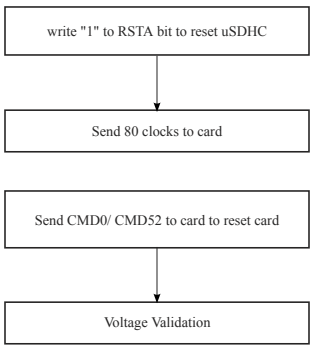


Figure 225. Flow chart for resetting uSDHC and SD I/O card

```
software_reset()
{
set_bit(SYSCTRL, RSTA); // software reset the Host
set DTOCV and SDCLKFS bit fields to get the CLK of frequency around 400kHz
configure IO pad to set the power voltage of external card to around 3.0V
poll bits CIHB and CDIHB bits of PRSSTAT to wait both bits are cleared
set_bit(SYSCTRL, INTIA); // send 80 clock ticks for card to power up
send_command(CMD_GO_IDLE_STATE, <other parameters>); // reset the card with CMD0
or send_command(CMD_IO_RW_DIRECT, <other parameters>);
}
```

39.6.2.3 Voltage validation

All cards should be able to establish communication with the host using any operation voltage in the maximum allowed voltage range specified in the card specification. However, the supported minimum and maximum values for VDD are defined in the Operation Conditions Register (OCR) and may not cover the whole range. Cards that store the CID and CSD data in the preload memory are only able to communicate this information under data transfer VDD conditions. This means that if the host and card have non-common VDD ranges, the card is neither able to complete the identification cycle nor able to send CSD data.

Therefore, special commands Send_Op_Cont (CMD1 for MMC), SD_Send_Op_Cont (ACMD41 for SD Memory), and IO_Send_Op_Cont (CMD5 for SD I/O) are used. The voltage validation procedure is designed to provide a mechanism to identify and reject cards that do not match the VDD range(s) desired by the host. This is accomplished when the host sends the desired VDD voltage window as the operand of this command. Cards that cannot perform the data transfer in the specified range must discard themselves from further bus operations and go into the Inactive state. By omitting the voltage range in the command, the host can query each card and determine the common voltage range before sending out-of-range cards into the Inactive state. This query should be used if the host is able to select a common voltage range or if a notification is sent to the system when a non-usable card in the stack is detected.

The following steps show how to perform voltage validation when a card is inserted:

```
voltage_validation(voltage_range_arguement)
{
label the card as UNKNOWN;
send_command(IO_SEND_OP_COND, 0x0, <other parameters are omitted>); // CMD5, check SDIO
operation voltage, command argument is zero
if (RESP_TIMEOUT != wait_for_response(IO_SEND_OP_COND)) { // SDIO command is accepted
          if (0 < number of IO functions) {
                    label the card as SDIO;
                    IORDY = 0;
                    while (!(IORDY in IO OCR response)) { // set voltage range for each IO function
                              send_command(IO_SEND_OP_COND, <voltage range>, <other parameter>);
```

```
                              wait_for_response(IO_SEND_OP_COND);
                    } // end of while ...
          } // end of if (0 < ...
          if (memory part is present inside SDIO card) Label the card as SDCombo; // this is an
SD-Combo card
} // end of if (RESP_TIMEOUT ...
if (the card is labelled as SDIO card) return; // card type is identified and voltage range is
set, so exit the function;
send_command(APP_CMD, 0x0, <other parameters are omitted>); // CMD55, Application specific CMD
prefix
if (no error calling wait_for_response(APP_CMD, <...>) { // CMD55 is accepted
          send_command(SD_APP_OP_COND, <voltage range>, <...>); // ACMD41, to set voltage range
for memory part or SD card
          wait_for_response(SD_APP_OP_COND); // voltage range is set
          if (card type is UNKNOWN) label the card as SD;
          return; //
} // end of if (no error ...
else if (errors other than time-out occur) { // command/response pair is corrupted
          deal with it by program specific manner;
} // of else if (response time-out
else { // CMD55 is refuse, it must be MMC card  if (card is already labelled as SDCombo) { //
change label
                    re-label the card as SDIO;
                    ignore the error or report it;
                    return; // card is identified as SDIO card
          } // of if (card is ...
          send_command(SEND_OP_COND, <voltage range>, <...>);
          if (RESP_TIMEOUT == wait_for_response(SEND_OP_COND)) { // CMD1 is not accepted, either
                    label the card as UNKNOWN;
                    return;
          } // of if (RESP_TIMEOUT ...
} // of else
}
```

39.6.2.4 Card registry

Card registry for the MMC and SD/SDIO/SD combo cards are different. For the SD card, the identification process starts at a clock rate lower than 400 kHz and the power voltage higher than 2.7 V (as defined by the card spec). Currently, the CMD line output drives are push-pull drivers instead of open-drain. After the bus is activated, the host requests the card to send their valid operation conditions. The response to ACMD41 is the operation condition register of the card. The same command is sent to all the new cards in the system. Incompatible cards are put into the Inactive state. The host then issues the command, All_Send_CID (CMD2), to each card to get its unique card identification (CID) number. Cards that are currently unidentified (in the Ready state), send their CID number as the response. After the CID is sent by the card, the card goes into the Identification state.

The host then issues Send_Relative_Addr (CMD3), requesting the card to publish a new relative card address (RCA) that is shorter than the CID. This RCA is used to address the card for future data transfer operations. After the RCA is received, the card changes its state to the Standby state. At this point, if the host wants the card to have an alternative RCA number, it may ask the card to publish a new number by sending another Send_Relative_Addr command to the card. The last published RCA is the actual RCA of the card.

The host repeats the identification process with CMD2 and CMD3 for each card in the system until the last CMD2 gets no response from any of the cards in the system.

For MMC operation, the host starts the card identification process in the open-drain mode with the identification clock rate lower than 400 kHz and the power voltage higher than 2.7 V. The open drain driver stages on the CMD line to allow parallel card operation during card identification. After the bus is activated, the host requests the cards to send their valid operation conditions (CMD1). The response to CMD1 is the "wired OR" operation on the condition restrictions of all cards in the system. Incompatible cards are sent into the Inactive state. The host then issues the broadcast command All_Send_CID (CMD2), asking all cards for

their unique card identification (CID) number. All unidentified cards (the cards in the Ready state) simultaneously start sending their CID numbers serially, while bit-wise monitoring their outgoing bit stream. Those cards, whose outgoing CID bits do not match the corresponding bits on the command line in any one of the bit periods, stop sending their CID immediately and must wait for the next identification cycle. As the CID is unique for each card, only one card can be successfully sent its full CID to the host. This card then goes into the Identification state. Thereafter, the host issues Set_Relative_Addr (CMD3) to assign a relative card address (RCA) to the card. After the RCA is received, the state of the card changes to standby, and the card does not react in further identification cycles. Also, its output driver switches from open-drain to push-pull. The host repeats the process, mainly CMD2 and CMD3, until the host receives a time-out condition to recognize the completion of the identification process.

The following steps show how to perform an operation using MMC cards:

```
card_registry()
{
do { // decide RCA for each card until response time-out
          if(card is labelled as SDCombo or SDIO) { // for SDIO card like device
                    send_command(SET_RELATIVE_ADDR, 0x00, <...>); // ask SDIO card to publish its
RCA
                    retrieve RCA from response;
          } // end if (card is labelled as SDCombo ...
          else if (card is labelled as SD) { // for SD card
                    send_command(ALL_SEND_CID, <...>);
                    if (RESP_TIMEOUT == wait_for_response(ALL_SEND_CID)) break;
                    send_command(SET_RELATIVE_ADDR, <...>);
                    retrieve RCA from response;
          } // else if (card is labelled as SD ...
          else if (card is labelled as MMC) {
                    send_command(ALL_SEND_CID, <...>);
                    rca = 0x1; // arbitrarily set RCA, 1 here for example
                    send_command(SET_RELATIVE_ADDR, 0x1 << 16, <...>); // send RCA at upper 16
bits
          } // end of else if (card is labelled as MMC ...
} while (response is not time-out);
}
```

39.6.3 Card access

Information about Block Write, Block Read, Suspense Resume, ADMA Usage, Transfer Error, and Card Interrupt are detailed in the sections below.

39.6.3.1 Block write

Information on Normal Write, DDR Write, and Write with Pause are detailed in the sections below.

39.6.3.1.1 Normal Write

During a block write (CMD24 - 27, CMD60, CMD61), one or more blocks of data are transferred from the host to the card with a CRC appended to the end of each block by the host. If the CRC fails, the card indicates that the failure on the DATA line. The transferred data is discarded and not written, and all further transmitted blocks (in multiple block write mode) are ignored.

If the host uses partial blocks whose accumulated length is not block aligned and block misalignment is not allowed (CSD parameter WRITE_BLK_MISALIGN is not set), the card detects the block misalignment error and aborts the programming before the beginning of the first misaligned block. The card sets the ADDRESS_ERROR error bit in the status register, and while ignoring all further data transfer, waits in the Receive-data-State for a stop command. The write operation is also aborted if the host tries to write over a write-protected area.

For MMC and SD cards, programming of the CID and CSD registers does not require a previous block length setting. The transferred data is also CRC protected. If a part of the CSD or CID register is stored in ROM, then this unchangeable part must match the corresponding part of the receive buffer. If this match fails, then the card reports an error and does not change any register contents.

For all types of cards, some may require long and unpredictable periods of time to write a block of data. After receiving a block of data and completing the CRC check, the card begins writing and holds the DATA line low if its write buffer is full and unable to accept new data from a new WRITE_BLOCK command. The host may poll the status of the card with a SEND_STATUS command (CMD13) or other means for SDIO cards at any time, and the card responds with its status. The responded status indicates whether the card can accept new data or whether the write process is still in progress. The host may deselect the card by issuing a CMD7 (to select a different card) to place the card into the Standby state and release the DATA line without interrupting the write operation. When re-selecting the card, it reactivates the busy indication by pulling data to low if the programming is still in progress and the write buffer is unavailable.

The software flow to write to a card incorporates the internal DMA and the write operation is a multi-block write with the Auto CMD12 enabled. For the other methods (CPU polling status) with different transfer methods, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to write to a card is described below:

1. Check the card status, wait until the card is ready for data.
2. Set the card block length/size:
   - For SD/MMC cards, use SET_BLOCKLEN (CMD16).
   - For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT (CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred, or another error that occurred during the auto12 command sending and response receiving.

39.6.3.1.2 DDR write

uSDHC supports the dual data rate mode.

The software flow to write to a card in the DDR mode is described as below:

1. Check the card status and wait until the card is ready for data.

NOTE

For eMMC card, block length only can be set to 512byte.

2. Set the uSDHC number block register (NOB), where nob is 5, for instance.
3. Set the eMMC card to high-speed mode and use SWITCH(CMD6).
4. Set the eMMC card bus with 4-bit/8-bit DDR mode and use SWITCH(CMD6).
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The DDR_EN and AC12EN bits should be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred or another error that occurred during the auto12 command sending and response receiving.

39.6.3.1.3 Write with Pause

The write operation can be paused during the transfer. Instead of stopping the CLK at any time to pause all the operations, which is also inaccessible to the host driver, the driver can set the Stop At Block Gap Request (SABGREQ) bit in the Protocol

Control register to pause the transfer between the data blocks. As there is no time-out condition in a write operation during the data blocks, a write to all types of cards can be paused in this way, and if the DATA0 line is not required to de-assert to release the busy state, no suspend command is needed.

Similar to the flow described in Normal Write, the write with pause is shown with the same kind of write operation:

1. Check the card status and wait until the card is ready for data.
2. Set the card block length/size:
   - For SD/MMC, use SET_BLOCKLEN (CMD16).
   - For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Set the SABGREQ bit.
7. Wait for the Transfer Complete interrupt.
8. Clear the SABGREQ bit.
9. Check the status bit to see if a write CRC error occurred.
10. Set the CREQ bit to continue the write operation.
11. Wait for the Transfer Complete interrupt.
12. Check the status bit to see if a write CRC error occurred or some another error that occurred during the auto12 command sending and response receiving.

The number of blocks left during the data transfer is accessible by reading the contents of the BLKCNT field in the Block Attribute register. As the data transfer and the setting of the SABGREQ bit are concurrent, and the delay of register read and the register setting, the actual number of blocks left may not be exactly the value read earlier. The host driver reads the value of BLKCNT after the transfer is paused and the Transfer Complete interrupt is received.

It is also possible that the last block has begun when the Stop At Block Gap Request is sent to the buffer. In this case, the next block gap is the end of the transfer. These types of requests are ignored, the driver should treat these as a non-pause transfer, and deal with it as a common write operation.

When the write operation is paused, the data transfer inside the host system is not stopped, and the transfer is active until the data buffer is full. Because of this, it is recommended to avoid using the Suspend command for the SDIO card. This is because when such a command is sent, uSDHC interprets the system that switches to another function on the SDIO card and flush the data buffer. uSDHC takes the Resume command as a normal command with data transfer, and it is left for the host driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the Transfer Type register are set as well as the AC12EN bit. However, the uSDHC module automatically sends a CMD12 to mark the end of the multi-block transfer.

39.6.3.2 Block read

Information about Normal read, DDR read, Read with Pause, and Delay Line (DLL) in Read Path are detailed in the sections below.

39.6.3.2.1 Normal read

For block reads, the basic unit of data transfer is a block whose maximum size is stored in areas defined by the corresponding card specification. A CRC is appended to the end of each block, ensuring data transfer integrity. The CMD17, CMD18, CMD53, CMD60, CMD61, and so on, can initiate a block read. After completing the transfer, the card returns to the Transfer state. For multi blocks read, data blocks are continuously transferred until a stop command is issued.

The software flow to read from a card incorporates the internal DMA and the read operation is a multi-block read with the Auto CMD12 enabled. For the other methods (CPU polling status) with different transfer methods, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to read from a card is described below:

1. Check the card status and wait until card is ready for data.
2. Set the card block length/size:
   - For SD/MMC, use SET_BLOCKLEN (CMD16).
   - For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
3. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
4. Set the uSDHC number block register (NOB), where nob is 5, for instance.
5. Disable the buffer read ready interrupt, configure the DMA settings, and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a read CRC error occurred, or another error occurred during the auto12 command sending and response receiving.

39.6.3.2.2 DDR read

The uSDHC module supports dual data rate mode.

The software flow to write to a card in the DDR mode is described below:

1. Check the card status ans wait until the card is ready for data.

NOTE

For eMMC card, block length can be set to only 512 bytes.

2. Set the uSDHC number block register (NOB) where nob is 5, for instance.
3. Set the eMMC card to high-speed mode and use SWITCH (CMD6).
4. Set the eMMC card bus with 4-bit /8-bit DDR mode and use SWITCH(CMD6).
5. Disable the buffer write ready interrupt, configure the DMA settings, and enable the uSDHC DMA when sending the command with data transfer. The DDR_EN and AC12EN bits should be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred, or another error that occurred during the auto12 command sending and response receiving.

39.6.3.2.3 Read with Pause

The read operation is not generally able to pause. Only the SDIO card (and SDCombo card working under I/O mode) supporting the Read Wait feature can pause during the read operation. If the SDIO card supports Read Wait (SRW bit in CCCR register is 1), the host driver can set the SABGREQ bit in the Protocol Control register to pause the transfer between the data blocks. Before setting the SABGREQ bit, ensure that the RWCTL bit in the Protocol Control register is set, otherwise uSDHC does not assert the Read Wait signal during the block gap and data corruption occurs. It is recommended to set the RWCTL bit after the Read Wait capability of the SDIO card is recognized.

Similar to the flow described in Normal read, the read with pause is shown with the same kind of read operation:

1. Check the SRW bit in the CCR register on the SDIO card to confirm that the card supports the Read Wait mode.
2. Set the RWCTL bit.

3. Check the card status and wait until the card is ready for data.
4. Set the card block length/size:
   - For SD/MMC, use SET_BLOCKLEN (CMD16).
   - For SDIO cards or the I/O portion of SDCombo cards, use IO_RW_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7).
5. Set the uSDHC block length register to be the same as the block length set for the card in step 2.
6. Set the uSDHC number block register (NOB), where nob is 5, for instance.
7. Disable the buffer read ready interrupt, configure the DMA setting, and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
8. Set the SABGREQ bit.
9. Wait for the Transfer Complete interrupt.
10. Clear the SABGREQ bit.
11. Check the status bit to see if read CRC error occurred.
12. Set the CREQ bit to continue the read operation.
13. Wait for the Transfer Complete interrupt.
14. Check the status bit to see if a read CRC error occurred, or another error occurred during the auto12 command sending and response receiving.

Similar to the Write operation, it is possible to meet the ending block of the transfer when paused. In this case, uSDHC ignores the Stop At Block Gap Request and treats it as a command read operation.

Unlike the write operation, there is no remaining data inside the buffer when the transfer is paused. All data received before the pause is transferred to the host system. No matter if the Suspend command is sent or not, the internal data buffer is not flushed.

If the Suspend command is sent and the transfer is later resumed by means of a Resume command, uSDHC takes the command as a normal one accompanied with data transfer. It is left for the host driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the Transfer Type register are set, as well as the AC12EN bit. However, the uSDHC automatically sends CMD12 to mark the end of multi-block transfer.

39.6.3.2.4 Delay Line (DLL) in read path

The DLL is newly added to assist in sampling read data. The DLL provides the ability to programmatically select a quantized delay (in fractions of the clock period) regardless of on-chip variations such as process, voltage, and temperature (PVT).

The reasons why DLL is needed for uSDHC are these:

- The path of read data traveling from card to host varies.
- In the SD/MMC DDR mode, the minimum input setup and hold time are both at 2.5 ns.

The data sampling window is so small that the delay of loopback clock needs to be accurate and consistent regardless of PVT. The DLL takes the divided card_clk as the reference clock and loopback clock as the input clock. It then generates a delayed version of the input clock according to the programmed target delay.

The DLL can be disabled or bypassed, and it can also be manually set for a fixed delay in the override mode. The override value set is the number of delay cells. In the override mode, there is no need to set the DLL_enable. Another DLL mode is target value mode. In this mode, the DLL automatically adjusts the number of delay cells according to the target value set by the user and PVT changes. Be aware that the target value is in units of 1/32 of the clock reference period. If the card_clk is 100Mhz, then the reference clock period is 10ns; setting target value of 16 means 5ns =(16/32)*10ns. The software can disable automatic update by the setting dll_gate_update bit.

As the user may change the frequency of card_clk from time to time by changing SDCLKFS[7:0]/DVS[3:0], the software must adjust the delay value to ensure it works correctly when the reference clock (card_clk) is changed. The following is the correct flow, which should be ignored if DLL_CTRL_ENABLE is not set.

Step 1: Set the DLL_CTRL_RESET bit

Step 2: Configure the SDCLKFS[7:0] and DVS[3:0]

Step 3: Wait until SDSTB is asserted

Step 4: Clear the DLL_CTRL_RESET bit

Step 5: Wait until both the DLL_STS_SLV_LOCK and DLL_STS_REF_LOCK are asserted

Step 6: Set the DLL_CTRL_SLV_FORCE_UPD

Step 7: Clear the DLL_CTRL_SLV_FORCE_UPD

NOTE

The software should make sure that the DLL_CTRL_SLV_FORCE_UPD lasts for at least one card_clk. So, the software may need to add some delay between step 6 and step 7.

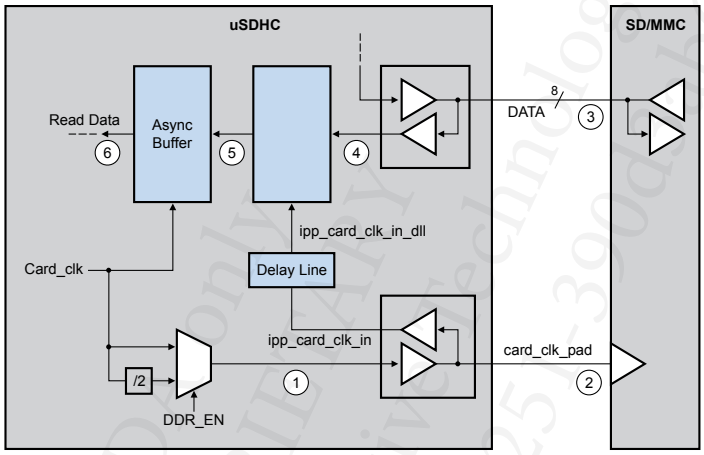


Figure 226. DLL in read path

39.6.3.3 Suspend Resume

The uSDHC module supports the Suspend Resume operations of SDIO cards, although slightly different than the suggested implementation of Suspend in the SDIO card specification.

39.6.3.3.1 Suspend

After setting the SABGREQ bit, the host driver may send a Suspend command to switch to another function of the SDIO card. The uSDHC module does not monitor the content of the response, so it does not know if the Suspend command succeeded or not. Accordingly, it does not de-assert Read Wait for read pause. To solve this problem, the host driver does not mark the Suspend command as "Suspend", (that is, setting the CMDTYP bits to 01). Instead, the driver sends this command as if it were a normal command, and only when the command succeeds, and the BS bit is set in the response, can the Driver send another command marked as "Suspend" to inform uSDHC that the current transfer is suspended. Here is the sequence for the Suspend operation:

1. Set the SABREQ bit to pause the current data transfer at block gap.
2. After the BGE bit is set, send the Suspend command to suspend the active function. The CMDTYP bit field must be 2'b00.
3. Check the BS bit of the CCCR in the response. If it is 1, repeat this step until the BS bit is cleared or abandon the suspend operation according to the Driver strategy.
4. Send another normal I/O command to the suspended function. The CMDTYP of this command must be 2'b01, so uSDHC can detect this special setting and be informed that the paused operation has successfully suspended. If the paused transfer is a read operation, uSDHC stops driving DATA2 and goes to the idle state.

5. Save the context registers in the system memory for later use, including the DMA System Address Register (for internal DMA operation), and the Block Attribute Register.
6. Begin operation for another function on the SDIO card.

39.6.3.3.2 Resume

To resume the data transfer, a Resume command is issued:

1. To resume the suspended function, restore the context register with the saved value in step #5 of the Suspend operation.
2. Send the Resume command: In the Transfer Type register, all the bit fields are set to the value as if this were another ordinary data transfer instead of a transfer resume (except the CMDTYP is set to 2'b10).
3. If the Resume command has responded, the data transfer is resumed.

39.6.3.4 ADMA usage

To use the ADMA in a data transfer, the host driver must prepare the correct descriptor chain prior to sending the read/write command. The steps to prepare the correct descriptor chain are these:

1. Create a descriptor to set the data length that the current descriptor group is about to transfer. The data length should be even numbers of the block size.
2. Create another descriptor to transfer the data from the address setting in this descriptor. The data address must be at a page boundary (4KB address aligned).
3. If necessary, create a Link descriptor containing the address of the next descriptor. The descriptor group is created in steps 1 ~ 3.
4. Repeat steps 1 ~ 3 until all descriptors are created.
5. In the last descriptor, set the End flag to 1 and make sure the total length of all descriptors matches the product of the block size and block number configured in the Block Attribute Register.
6. Set the ADMA System Address Register to the address of the first descriptor and set the DMAS field in the Protocol Control Register to 01 to select the ADMA.
7. Issue a write or read command with the DMAEN bit set to 1 in the Transfer Type Register.

Steps 1 ~ 5 are independent of step 6, so step 6 can finish before steps 1 ~ 5. Regarding the descriptor configuration, it is recommended not to use the Link descriptor as it requires extra system memory access.

39.6.3.5 Transfer error

Information about CRC, Internal DMA, Transfer ADMA, and Auto CMD12 errors are detailed in the sections below.

39.6.3.5.1 CRC error

It is possible at the end of a block transfer that a write CRC status error or read CRC error occurs. For this type of error, the latest block received is discarded. This is because the integrity of the data block is not guaranteed. It is recommended to discard the following data blocks and re-transfer the block from the corrupted one. For a multi-block transfer, the host driver issues a CMD12 to abort the current process and start the transfer by a new data command. In this scenario, even when the AC12EN and BCEND bits are set, uSDHC does not automatically send a CMD12 because the last block is not transferred. On the other hand, if it is within the last block that the CRC error occurs, an Auto CMD12 is sent by uSDHC. In this case, the host driver re-sends or re-obtains the last block with a single block transfer.

39.6.3.5.2 Internal DMA error

During the data transfer with internal Simple DMA, if the DMA engine encounters an error on the AXI bus, the DMA operation is aborted, and the DMA error interrupt is sent to the host system. When acknowledged by such an interrupt, the host driver calculates the start address of data block in which the error occurs. The start address can be calculated by either:

- Reading the DMA System Address register: The error occurs during the previous burst. Considering the block size, and the start address of the next burst transfer, it is straight forward to obtain the start address of the corrupted block.
- Reading the BLKCNT field of the Block Attribute register: Considering the number of blocks left, the total number to transfer, the start address of transfer, and the size of each block, the start address of corrupted block can be determined. When the BCEN bit is not set, the contents of the Block Attribute register do not change, so this method does not work.

When a DMA error occurs, it is recommended to abort the current transfer by means of a CMD12 (for multi block transfer), apply a reset for data, and restart the transfer from the corrupted block to recover from the error.

39.6.3.5.3 Transfer ADMA error

There are three kinds of possible ADMA errors: The AXI transfer, invalid descriptor, and data-length mismatch. Whenever these errors occur, the DMA transfer stops and the corresponding error status bit is set. For acknowledging the status, the host driver should recover the error as shown below and re-transfer from the place of interruption.

- AXI transfer error: Such errors may occur during data transfer or descriptor fetch. For either scenario, it is recommended to retrieve the transfer context, reset for the data part and re-transfer the block that was corrupted, or to the next block if no block is corrupted.
- Invalid descriptor error: For such errors, it is recommended to retrieve the transfer context, reset for the data part and recreate the descriptor chain from the invalid descriptor and issue a new transfer. As the data to transfer now may be less than the previous setting, the data length configured in the new descriptor chain should match the new value.
- Data-length mismatch error: It is similar to recover from this error. The Host Driver polls relating registers to retrieve the transfer context, apply a reset for the data part, configure a new descriptor chain and make another transfer if there is data left. Like the previous scenario of the invalid descriptor error, the data length must match the new transfer.

39.6.3.5.4 Auto CMD12 error

After the last block of the multi-block transfer is sent or received, and the AC12EN bit is set when the data transfer is initiated by the data command, uSDHC automatically sends a CMD12 to the card to stop the transfer. When errors with this command occur, it is recommended to the host driver to deal with the situations in the following manner:

- Auto CMD12 response time-out: It is not certain whether the command is accepted by the card or not. The host driver clears the auto CMD12 error status bits and re-send CMD12 until it is accepted by the card.
- Auto CMD12 response CRC error: As the card responds to CMD12, it aborts the transfer. The host driver may ignore the error and clear the error status bit.
- Auto CMD12 conflict error or not sent: The command is not sent; therefore, the host driver sends a CMD12 manually.

39.6.3.6 Card interrupt

The external cards can inform the host controller by means of some special signals. For the SDIO card, it can be the low-level on the DATA1 line during some special period. The uSDHC module only monitors the DATA1 line and supports the SDIO interrupt.

When the SDIO interrupt is captured by uSDHC, and the host system is informed by uSDHC asserting the uSDHC interrupt line, the interrupt service from the host driver is called.

As the interrupt source is controlled by the external card, the interrupt from the SDIO card must be serviced before the CINT bit is cleared by written. Refer to Card interrupt handling for the card interrupt handling flow.

39.6.4 Switch function

A switch command is issued by the host driver to enable new features added to the SD/MMC spec. The SD/MMC cards can transfer data at bus widths other than 1-bit. Different speed modes are also defined. To enable these features, a switch command is issued by the host driver.

For SDIO cards, the high-speed mode/DDR50/SDR50/SDR104 are enabled by writing the EHS bit in the CCCR register after the SHS bit is confirmed. For SD cards, the high-speed mode/DDR50/SDR50/SDR104 are queried and enabled by a CMD6 (with

the mnemonic symbol as SWITCH_FUNC). For MMC cards, the high-speed mode/HS200/HS400 are queried by a CMD8 and enabled by a CMD6 (with the mnemonic symbol as SWITCH).

The SDR4-bit, SDR8-bit, DDR4-bit, and DDR8-bit width of MMC is also enabled by the SWITCH command, but with a different argument.

These new functions can also be disabled by a software reset. For SDIO cards, it can be done by setting the RES bit in the CCCR register. For other cards, it can be accomplished by issuing a CMD0. This method of restoring to the normal mode is not recommended because a complete identification process is needed before the card is ready for data transfer.

For the sake of simplicity, the following pseudocode examples do not show current capability check, which is recommended in the function switch process.

39.6.4.1 Query, enable, and disable SDIO high-speed mode

```
enable_sdio_high_speed_mode(void)
{
send CMD52 to query bit SHS at address 0x13;
if (SHS bit is '0') report the SDIO card does not support high speed mode and return;
send CMD52 to set bit EHS at address 0x13 and read after write to confirm EHS bit is set;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of around 50MHz;
(data transactions like normal peers)
}
disable_sdio_high_speed_mode(void)
{
send CMD52 to clear bit EHS at address 0x13 and read after write to confirm EHS bit is cleared;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}
```

39.6.4.2 Query, enable, and disable SD high-speed mode/DDR50/SDR50/SDR104

```
enable_sd_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0xFFFFFx and read 64 bytes of data accompanying the R1 response;(high speed
mode,x=1; SDR50,x=2; SDR104,x=3; DDR50,x=4;)
wait data transfer done bit is set;
check if the bit x of received 512 bits is set;
if (bit 401 is '0') report the SD card does not support high speed mode and return;
if (bit 402 is '0') report the SD card does not support SDR50 mode and return;
if (bit 403 is '0') report the SD card does not support SDR104 mode and return;
if (bit 404 is '0') report the SD card does not support DDR50 mode and return;
send CMD6, with argument 0x80FFFFFx and read 64 bytes of data accompanying the R1 response;(high
speed mode,x=1; SDR50,x=2; SDR104 x=3; DDR50 x=4;)
check if the bit field 379~376 is 0xF;
if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the card_clk
of around 50MHz for high speed mode, 100MHz for SDR50, 200MHz for SDR104, 50MHz for DDR50;
(data transactions like normal peers)
}
disable_sd_speed_mode(void)
{
set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
send CMD6, with argument 0x80FFFFF0 and read 64 bytes of data accompanying the R1 response;
check if the bit field 379~376 is 0xF;
if (the bit field is 0xF) report the function switch failed and return;
```

```
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}
```

39.6.4.3 Query, enable, and disable MMC high-speed mode

```
enable_mmc_high_speed_mode(void)
{
send CMD9 to get CSD value of MMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the MMC does not support high speed mode and return;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of MMC;
extract the value of CARD_TYPE field to check the 'high speed mode' in this MMC is 26MHz or
52MHz;
send CMD6 with argument 0x1B90100;
send CMD13 to wait card ready (busy line released);
send CMD8 to get EXT_CSD value of MMC;
check if HS_TIMING byte (byte number 185) is 1;
if (HS_TIMING is not 1) report MMC switching to high speed mode failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of around 26MHz or 52MHz according to the CARD_TYPE;
(data transactions like normal peers)
}
disable_mmc_high_speed_mode(void)
{
send CMD6 with argument 0x2B90100;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of MMC;
check if HS_TIMING byte (byte number 185) is 0;
if (HS_TIMING is not 0) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 20MHz;
(data transactions like normal peers)
}
```

39.6.4.4 Set MMC bus width

```
change_mmc_bus_width(void)
{
send CMD9 to get CSD value of MMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the MMC does not support multiple bit width and return;
send CMD6 with argument 0x3B70x00; (8-bit(dual data rate), x=6; 4-bit(dual data rate), x=5;8-bit,
x=2; 4-bit, x=1; 1-bit, x=0)
send CMD13 to wait card ready (busy line released);
(data transactions like normal peers)
}
```

39.6.5 ADMA operation

Here are the codes for the ADMA1 and ADMA2 operations.

Provided under NDA only
COMPANY PROPRIETARY
Beijing eSOL Automotive Technology Co., Ltd.
c8462e5-3821-48a8-8251-3903ab6241

39.6.5.1 ADMA1 operation

```
Set_adma1_descriptor
{
if (to start data transfer) {
// Make sure the address is 4KB align.
Set 'Set' type descriptor;
{
Set Act bits to 01;
Set [31:12] bits data length (byte unit);
}
Set 'Tran' type descriptor;
{
Set Act bits to 10;
Set [31:12] bits address (4KB align);
}
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to 11;
Set [31:12] bits the next descriptor address (4KB aligned);
}
else { // other types of descriptor
Set Act bits accordingly
}
if (this descriptor is the last one) {
Set End bit to 1;
}
if (to generate interrupt for this descriptor) {
Set Int bit to 1;
}
Set Valid bit to 1;
}
```

39.6.5.2 ADMA2 operation

```
Set_adma2_descriptor
{
if (to start data transfer) {
// Make sure the address is a 32-bit boundary (lower 2-bit are always '00').
Set higher 32-bit of descriptor for this data transfer initial address;
Set [31:16] bits data length (byte unit);
Set Act bits to '10';
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to '11';
// Make sure the address is 32-bit boundary (lower 2-bit are always set to '00').
Set higher 32-bit of descriptor for the next descriptor address;
}
else { // other types of descriptor
Set Act bits accordingly
}
if (this descriptor is the last one) {
Set 'End' bit '1';
}
if (to generate interrupt for this descriptor) {
Set 'Int' bit '1';
}
```

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

```
Set the 'Valid' bit to '1';
}
```

39.6.6 Fast boot operation

39.6.6.1 Normal fast boot flow

Here are the steps for normal fast boot flow:

1. Software must configure the init_active bit (system control register bit 27) to make sure that 74 card clocks are finished.
2. Software must configure the MMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot), and bit 5 to 0 (normal fast boot), and bit 4 to select the ack mode. If the data is sent through the DMA mode, the software should configure bit 7 to enable the automatic stop at block gap feature, and configure bit 3-bit 0 to select the ack timeout value according to the SD CLK frequency.
3. Software then needs to configure the Block Attributes Register to set the block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes.
4. Software must configure the Protocol control register to set Data Transfer Width (DTW). If in the DDR fast boot mode, DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software needs to configure the Command Argument Register to set argument if needed (no need in normal fast boot).
6. Software must configure the Transfer Type Register to start the boot process. In normal boot mode, CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN and DMAEN retain the default value, where DPSEL bit is set to 1, DTDSEL is set to 1 and MSBSEL is set to 1.
7. DMAEN should be configured as 0 in the polling mode and if BCEN is configured as 1, it is recommended to configure the number of blocks in the Block Attributes Register to the maximum value. If in DDR fast boot mode, DDR_EN needs to be set to 1.
8. When step 6 is configured, the boot process begins. The software needs to poll the data buffer ready status to read the data from the buffer in time. If a boot timeout happens (ack times out or the first data read times out), an interrupt is triggered, and the software must configure MMC Boot Register to bit 6 to 0 to disable boot. This makes CMD high, then after at least 56 clocks, it is ready to begin a normal initialization process.
9. If there is no timeout, software needs to determine when the data read is finished and then configure MMC Boot Register bit 6 to 0 to disable boot. This render CMD line high and command completed asserted. After at least 56 clocks, it is ready to begin the normal initialization process.
10. You must reset the host and begin the normal process.

39.6.6.2 Alternative fast boot flow

Here are the steps for alternative fast boot flow:

1. Software needs to configure init_active bit (system control register bit 27) to make sure 74 card clocks are finished.
2. Software needs to configure MMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot), and bit 5 to 1 (alternative boot), and bit 4 to select the ack mode or not. If data needs to be sent through the DMA mode, then configure bit 7 to enable the automatic stop at block gap feature. Software should also configure bit 3-bit 0 to select the ack timeout value according to the SD clock frequency.
3. Software then needs to configure the Block Attributes Register to set the block size and count. If in the DDR fast boot mode, the block size only can be configured to 512 bytes.
4. Software needs to configure the Protocol control register to set the data transfer width (DTW). If in the DDR fast boot mode, DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software needs to configure Command Argument Register to set argument to 0xFFFFFFFA.
6. Software needs to configure the Transfer Type Register to start the boot process by CMD0 with the 0xFFFFFFFA argument. In alternative boot, CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN, and DMAEN retain the

default value. DPSEL bit is set to 1, DTDSEL is set to 1, and MSBSEL is set to 1. Note DMAEN should be configured as 0 in the polling mode, and if BCEN is configured as 1 in polling mode, it is recommended to configure the block count in the Block Attributes Register to the maximum value. If in the DDR fast boot mode, DDR_EN needs to be set to 1.

7. When step 6 is configured, the boot process begins. Software needs to poll the data buffer ready status to read the data from the buffer in time. If there is a boot timeout (ack data timeout in 50ms or data timeout in 1s), the host sends out the interrupt and the software needs to send CMD0 with reset and then configure the boot enable bit to 0 to stop this process.
8. If there is no time out, the software needs to decide when to stop the boot process, and send out the CMD0 with reset and then after the command is completed, configure the MMC Boot Register bit 6 to stop the process. After 8 clocks from the command completion, the slave (card) is ready for the identification step.
9. You must reset the host and begin the normal process.

39.6.6.3 Fast boot application case (in DMA mode)

In the boot application case, because the image destination and the image size are contained in the beginning of the image, it is necessary to switch DMA parameters on the fly during MMC fast boot.

In fast boot, the host can use Advanced DMA2 (ADMA2) with two destinations.

The detailed flow is described below:

1. The software needs to configure INIT_ACTIVE bit (system control register bit 27) to make sure that 74 card clocks are finished.
2. The software needs to configure the MMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot); and bit 5 to 0 (normal fast boot) or 1 (alternative boot); and bit 4 to select the ack mode. In DMA mode, configure bit 7 to 1 to enable the automatic stop at block gap feature. Also configure bits[31-16] to set the (BLK_CNT - VALUE1). Here VALUE1 is the value of the block count that needs to transfer the first time, so that the host stops at the block gap when the uSDHC controller gets VAULE1 blocks from the device. Also, configure bits[3-0] to select the ack timeout value according to the SD clock frequency.
3. The software then needs to configure the Block Attributes Register to set block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes. In DMA mode, it is recommended to set the block count (BLK_CNT) to the max value (16'hffff).
4. The software needs to configure Protocol Control Register to set DTW (data transfer width). If in DDR fast boot mode, the DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software enable ADMA2 by configuring Protocol Control Register bits [9-8].
6. The software needs to set at least three pairs of ADMA2 descriptor in boot memory (that is, in IRAM, at least six words). The first pair descriptor defines the start address (that is, IRAM) and data length (that is,512byte*VALUE1) of the first part boot code. The software also needs to set the second pair descriptor, the second start address (any value that is writable), and data length is suggested to set 1~2word (record as VALUE2). Note that the second couple desc also transfers useful data even at lease 1 word, because our ADMA2 cannot support 0 data_length data transfer descriptor.
7. The software needs to configure Command Argument Register to set argument to 0xFFFFFFFA in alternative fast boot and do not need to be set in normal fast boot.
8. The software needs to configure Transfer Type Register to start the boot process. CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN, and DMAEN retain the default value. DPSEL bit is set to 1, DTDSEL is set to 1, and MSBSEL is set to 1. DMAEN is configured as 1 in the DMA mode. And, if BCEN is configured as 1, then configure blk no in Bock Attributes Register to the max value. And, if in the DDR fast boot mode, DDR_EN needs to be set to 1.
9. When step 8 is configured, boot process begins, the first VALUE1 block number data gets transferred. The software needs to poll the TC bit (bit1 in Interrupt Status Register) to determine first transfer is ended. Also, the software needs to polling the BGE bit (bit2 in Interrupt Status Register) to determine if the first transfer stops at the block gap.

10. When TC and BGE bits are set to 1, the software can analyze the first code of VALUE1 block, initializes the new memory device, if required, and sets the third pair of descriptors to define the start address and length of the remaining part of the boot code (VALUE3, the remain boot code block). Remember to set the last descriptor with END.
11. The software needs to configure the MMC Boot Register (offset 0xc4) again. Set bit 6 to 1 (enable boot); and bit 5 to 0 (normal fast boot), to 1 (alternative boot); and bit 4 to select the ack mode or not. In the DMA mode, configure bit 7 to 1 for enabling the automatically stop at block gap feature. Also, configure bit31-bit16 to set the (BLK_CNT - (VALUE1+1+VALUE3)), that host stops at block gap when the uSDHC controller gets (VALUE1+1+VALUE3)) blocks from device totally include the blocks received in step 9. And need to configure bit 3-bit0 to select the ack timeout value according to the sd clk frequency. Note that the software does not need to configure the BLK_CNT again, because it is counted down automatically by the uSDHC controller.
12. The software needs to clear the TC and BGE bits and the software needs to clear SABGREQ (bit 16 in the Protocol control register) and set CREQ (bit17 in the Protocol control register) to 1 to resume the data transfer. Host transfers the VALUE2 and VALUE3 data to the destination that is set by descriptor.
13. The software needs to do poll BGE bit to determine if the fast boot is over.

Note:

- When ADMA boot flow starts, for uSDHC, it is like a normal ADMA read operation. So, set ADMA2 descriptor as the normal ADMA2 transfer.
- Need a few words length memory to keep descriptor.
- For the 1~2-word data in second descriptor setting, it is a useful data, so the software needs to deal the data because of the application case.

39.7 Commands for MMC/SD/SDIO

A table containing the list of commands for the MMC/SD/SDIO cards is provided here.

Refer to the corresponding specifications for more details about the command information.

There are four kinds of commands defined to control the MultiMediaCard:

- Broadcast commands (bc), no response
- Broadcast commands with response (bcr), response from all cards simultaneously
- Addressed (point-to-point) commands (ac), no data transfer on the DATA
- Addressed (point-to-point) data transfer commands (adtc)

Response: A response is a token that is sent from the card to the host as an answer to a previously received command. A response is transferred serially on the CMD line.

Table 262. Commands for MMC/SD/SDIO cards

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD0	bc	[31:0] stuff bits	-	GO_IDLE_STATE	Resets all MMC and SD memory cards to idle state.
CMD1	bcr	[31:0] OCR without busy	R3	SEND_OP_COND	Asks all MMC and SD Memory cards in idle state to send their operation conditions register contents in the response on the CMD line.

Table continues on the next page...

Table 262. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD2	bcr	[31:0] stuff bits	R2	ALL_SEND_CID	Asks all cards to send their CID numbers on the CMD line.
CMD3[1]	ac	[31:6] RCA [15:0] stuff bits	R1 R6 (SDIO)	SET/ SEND_RELATIVE_ADDR	Assigns relative address to the card.
CMD4	bc	[31:0] DSR [15:0] stuff bits	-	SET_DSR	Programs the DSR of all cards.
CMD5	bc	[31:0] OCR without busy	R4	IO_SEND_OP_COND	Asks all SDIO cards in idle state to send them operation conditions register contents in the response on the CMD line.
CMD6[2]	adtc	[31] Mode 0: Check function 1: Switch function [30:8] Reserved for function groups 6 ~ 3 (All 0 or 0xFFFF) [7:4] Function group1 for command system [3:0] Function group2 for access mode	R1	SWITCH_FUNC	Checks switch ability (mode 0) and switch card function (mode 1). Refer to "SD Physical Specification V1.1" for more details.
CMD6[3]	ac	[31:26] Set to 0 [25:24] Access [23:16] Index [15:8] Value [7:3] Set to 0 [2:0] Cmd Set	R1b	SWITCH	Switches the mode of operation of the selected card or modifies the EXT_CSD registers. Refer to "The MultiMediaCard System Specification Version 4.0 Final draft 2" for more details.
CMD7	ac	[31:6] RCA [15:0] stuff bits	R1b	SELECT/ DESELECT_CARD	Toggles a card between the stand-by and transfer states or between the programming and disconnect states. In both cases, the card is selected by its own relative address and gets deselected by any other address. Address 0 deselects all.
CMD8	adtc	[31:0] stuff bits	R1	SEND_EXT_CSD	The card sends its EXT_CSD register as a block of data, with a block size of 512 bytes.

Table continues on the next page...

Table 262. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD9	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CSD	Addressed card sends its card-specific data (CSD) on the CMD line.
CMD10	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CID	Addressed card sends its card-identification (CID) on the CMD line.
CMD11	adtc	[31:0] data address	R1	READ_DAT_UNTIL_STOP	Reads data stream from the card, starting at the given address, until a STOP_TRANSMISSION follows.
CMD12	ac	[31:0] stuff bits	R1b	STOP_TRANSMISSION	Forces the card to stop transmission.
CMD13	ac	[31:6] RCA [15:0] stuff bits	R1	SEND_STATUS	Addressed card sends its status register.
CMD14	Reserved				
CMD15	ac	[31:6] RCA [15:0] stuff bits	-	GO_INACTIVE_STATE	Sets the card to inactive state in order to protect the card stack against communication breakdowns.
CMD16	ac	[31:0] block length	R1	SET_BLOCKLEN	Sets the block length (in bytes) for all following block commands (read and write). Default block length is specified in the CSD.
CMD17	adtc	[31:0] data address	R1	READ_SINGLE_BLOCK	Reads a block of the size selected by the SET_BLOCKLEN command.
CMD18	adtc	[31:0] data address	R1	READ_MULTIPLE_BLOCK	Continuously transfers data blocks from card to host until interrupted by a stop command.
CMD19	adtc	[31:0] reserved bits(all 0)	R1	SEND_TUNING_BLOCK	64 bytes tuning pattern is sent for SDR50 and SDR104.
CMD20	adtc	[31:0] data address	R1	WRITE_DAT_UNTIL_STOP	Writes data stream from the host, starting at the given address, until a STOP_TRANSMISION follows.
CMD21	adtc	[31:0] stuff bits	R1	SEND_TUNING_BLOCK	128 clocks of tuning pattern (64 byte in 4 bit mode or 128 byte in 8 bit mode) is sent for HS200 optimal sampling point detection.

Table continues on the next page...

Table 262. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD22-23	Reserved				
CMD24	adtc	[31:0] data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command.
CMD25	adtc	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	Continuously writes blocks of data until a STOP_TRANSMISSION follows.
CMD26	adtc	[31:0] stuff bits	R1	PROGRAM_CID	Programming of the card identification register. This command is issued only once per card. The card contains hardware to prevent this operation after the first programming. Normally this command is reserved for the manufacturer.
CMD27	adtc	[31:0] stuff bits	R1	PROGRAM_CSD	Programming of the programmable bits of the CSD.
CMD28	ac	[31:0] data address	R1b	SET_WRITE_PROT	If the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).
CMD29	ac	[31:0] data address	R1b	CLR_WRITE_PROT	If the card provides write protection features, this command clears the write protection bit of the addressed group.
CMD30	adtc	[31:0] write protect data address	R1	SEND_WRITE_PROT	If the card provides write protection features, this command asks the card to send the status of the write protection bits.
CMD31	Reserved				
CMD32	ac	[31:0] data address	R1	TAG_SECTOR_START	Sets the address of the first sector of the erase group.
CMD33	ac	[31:0] data address	R1	TAG_SECTOR_END	Sets the address of the last sector in a continuous range within the selection of a single sector to be selected for erase.

Table continues on the next page...

Table 262. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD34	ac	[31:0] data address	R1	UNTAG_SECTOR	Removes one previously selected sector from the erase selection.
CMD35	ac	[31:0] data address	R1	TAG_ERASE_GROUP_START	Sets the address of the first erase group within a range to be selected for erase.
CMD36	ac	[31:0] data address	R1	TAG_ERASE_GROUP_END	Sets the address of the last erase group within a continuous range to be selected for erase.
CMD37	ac	[31:0] data address	R1	UNTAG_ERASE_GROUP	Removes one previously selected erase group from the erase selection.
CMD38	ac	[31:0] stuff bits	R1b	ERASE	Erase all previously selected sectors.
CMD39	ac	[31:0] RCA [15] register write flag [14:8] register address [7:0] register data	R4	FAST_IO	Used to write and read 8-bit (register) data fields. The command addresses a card, and a register, and provides the data for writing if the write flag is set. The R4 response contains data read from the address register. This command accesses application dependent registers which are not defined in the MMC standard.
CMD40	bcr	[31:0] stuff bits	R5	GO_IRQ_STATE	Sets the system into interrupt mode.
CMD41	Reserved				
CDM42	adtc	[31:0] stuff bits	R1b	LOCK_UNLOCK	Used to set/reset the password or lock/unlock the card. The size of the data block is set by the SET_BLOCK_LEN command.
CMD43~51	Reserved				
CMD52	ac	[31:0] stuff bits	R5	IO_RW_DIRECT	Access a single register within the total 128k of register space in any I/O function.
CMD53	ac	[31:0] stuff bits	R5	IO_RW_EXTENDED	Accesses a multiple I/O register with a single command. Allows the reading or writing of a large number of I/O registers.

Table continues on the next page...

Table 262. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD54	Reserved				
CMD55	ac	[31:16] RCA [15:0] stuff bits	R1	APP_CMD	Indicates to the card that the next command is an application specific command rather than a standard command.
CMD56	adtc	[31:1] stuff bits [0]: RD/WR	R1b	GEN_CMD	Used either to transfer a data block to the card or to get a data block from the card for general purpose / application specific commands. The size of the data block is set by the SET_BLOCK_LEN command.
CMD57-59	Reserved				
CMD60	adtc	[31] WR [30:24] stuff bits [23:16] address [15:8] stuff bits [7:0] byte count	R1b	RW_MULTIPLE_REGISTER	These registers are used to control the behavior of the device and to retrieve status information regarding the operation of the device. All Status and Control registers are WORD (32-bit) in size and are WORD aligned. CMD60 is used to read and write these registers.
CMD61	adtc	[31] WR [30:16] stuff bits [15:0] data unit count	R1b	RW_MULTIPLE_BLOCK	The host issues a RW_MULTIPLE_BLOCK (CMD61) to begin the data transfer.
CMD62-63	Reserved				
ACMD6[4]	ac	[31:2] stuff bits [1:0] bus width	R1	SET_BUS_WIDTH	Defines the data bus width ('00'=1bit or '10'=4bit bus) to be used for data transfer. The allowed data bus widths are given in SCR register.
ACMD13[4]	adtc	[31:0] stuff bits	R1	SD_STATUS	Send the SD Memory Card status.
ACMD22[4]	adtc	[31:0] stuff bits	R1	SEND_NUM_WR_SECTORS	Send the number of the written sectors (without errors). Responds with 32-bit plus the CRC data block.
ACMD23[4]	ac	[31:23] stuff bits [22:0] Number of blocks	R1	SET_WR_BLK_ERASE_COUNT	Set the number of write blocks to be pre-erased before writing (to be used for fast Multiple Block

Table continues on the next page...

Table 262. Commands for MMC/SD/SDIO cards (continued)

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
					WR command). "1"=default(one write block).
ACMD41[4]	bcr	[31:0] OCR	R3	SD_APP_OP_COND	Asks the accessed card to send its operating condition register (OCR) contents in the response on the CMD line.
ACMD42[4]	ac	[31:1] stuff bits [0] set_cd	R1	SET_CLR_CARD_DETECT	Connect(1)/Disconnect(0) the 50KOhm pull-up resistor on CD_B/DATA3 of the card.
ACMD51[4]	adtc	[31:0] stuff bits	R1	SEND_SCR	Reads the SD Configuration Register (SCR).

1. CMD3 differs for MMC and SD cards. For MMC cards, it is referred to as SET_RELATIVE_ADDR, with a response type of R1. For SD cards, it is referred to as SEND_RELATIVE_ADDR, with a response type of R6 (with RCA inside).
2. CMD6 differs completely between high speed MMC cards and high-speed SD cards. Command SWITCH_FUNC is for high-speed SD cards.
3. Command SWITCH is for high speed MMC cards. The Index field can contain any value from 0-255, but only values 0-191 are valid. If the Index value is in the 192-255 range the card does not perform any modification and the SWITCH_ERROR status bit in the EXT_CSD register is set. The Access Bits are shown in Table 263.
4. ACMDs is preceded with the APP_CMD command. Commands listed are used for SD only, other SD commands not listed are not supported on this module.

The access bits for the EXT_CSD access modes are listed in the following table.

Table 263. EXT_CSD access modes

Bits	Access name	Operation
00	Command set	The command set is changed according to the Cmd Set field of the argument.
01	Set bits	The bits in the pointed byte are set, according to the bits set to 1 in the Value field.
10	Clear bits	The bits in the pointed byte are cleared, according to the bits set to 1 in the Value field.
11	Write byte	The Value field is written into the pointed byte.

39.8 Software restrictions

39.8.1 Initialization active

The driver cannot set INITA bit in System Control register when any of the command line or data lines are active, so the driver must ensure both CDIHB and CIHB bits are cleared.

39.8.2 Software polling procedure

For polling read or write, after the software begins a buffer read or write, it must access exactly the number of times as the values set in the Watermark Level Register; moreover, if the block size is not a multiple of the value in the Watermark Level Register (read and write respectively), the software must access exactly the remaining number of words at the end of each

block. For example, for a read operation, if the RD_WML is 4, indicating the watermark level is 16 bytes, block size is 40 bytes, and the block number is 2, then the access times for the burst sequence in the whole transfer process must be 4, 4, 2, 4, 4, 2.

39.8.3 Suspend operation

To suspend the data transfer, the software must inform uSDHC that the suspend command is successfully accepted. To achieve this, after the Suspend command is accepted by the SDIO card, software must send another normal command marked as suspend command (CMDTYP bits set as '01') to inform uSDHC that the transfer is suspended.

If software needs to resume the suspended transfer, it should read the value in BLKCNT register to save the remaining number of blocks before sending the normal command marked as suspend, otherwise on sending such a 'suspend' command, uSDHC treats the current transfer is aborted and change the BLKCNT register to its original value, instead of retaining the remaining number of blocks.

39.8.4 Data length setting

For either ADMA (ADMA1 or ADMA2) transfer, the data in the data buffer must be word aligned, so the data length set in the descriptor must be a multiple of 4.

39.8.5 (A)DMA address setting

To configure the ADMA1/ADMA2/DMA address register, when TC bit is set, the register always updates itself with the internal address value to support dynamic address synchronization, so the software must ensure that the TC bit is cleared prior to configuring the ADMA1/ADMA2/DMA address register.

39.8.6 Data port access

Data port does not support parallel access. For example, during an internal DMA access, it is not allowed to write any data to the data port by CPU; or during a CPU read operation, it is also prohibited to write any data to the data port, by either CPU or internal DMA. Otherwise the data is corrupted inside the uSDHC buffer.

39.8.7 Change clock frequency

The uSDHC module does not automatically gate off the card clock when the host driver changes the clock frequency. To prevent possible glitch on the card clock, clear the FRC_SDCLK_ON bit when changing the clock divisor value (SDCLKFS or DVS in System Control Register) or setting the RSTA bit.

Also, before changing the clock divisor value, the host driver should make sure that the SDSTB bit is high.

39.8.8 Multi-block read

For pre-defined multi-block read operation, that is, the number of blocks to read has been defined by previous CMD23 for MMC, or pre-defined number of blocks in CMD53 for SDIO/SDCombo, or whatever multi-block read without abort command at card side, an abort command, either automatic or manual CMD12/CMD52, is still required by uSDHC after the pre-defined number of blocks are done, to drive the internal state machine to idle mode. In this case, the card may not respond to this extra abort command and uSDHC gets response timeout. It is recommended to manually send an abort command with RSPTYP[1:0] both bits cleared.

Chapter 40
System Timer Module (STM)

40.1 Chip-specific STM information

40.1.1 STM instances

This chip implements these STMs. All instances have the same configuration.

- STM_0
- STM_1
- STM_2
- STM_3
- STM_4
- STM_5
- STM_6
- STM_7

40.2 Introduction

The System Timer Module (STM) supports commonly required system and application software timing functions. STM includes a 32-bit count-up timer and four 32-bit compare channels with a separate interrupt source for each channel. The timer is driven by the STM module clock divided by an 8-bit prescale value (1 to 256).

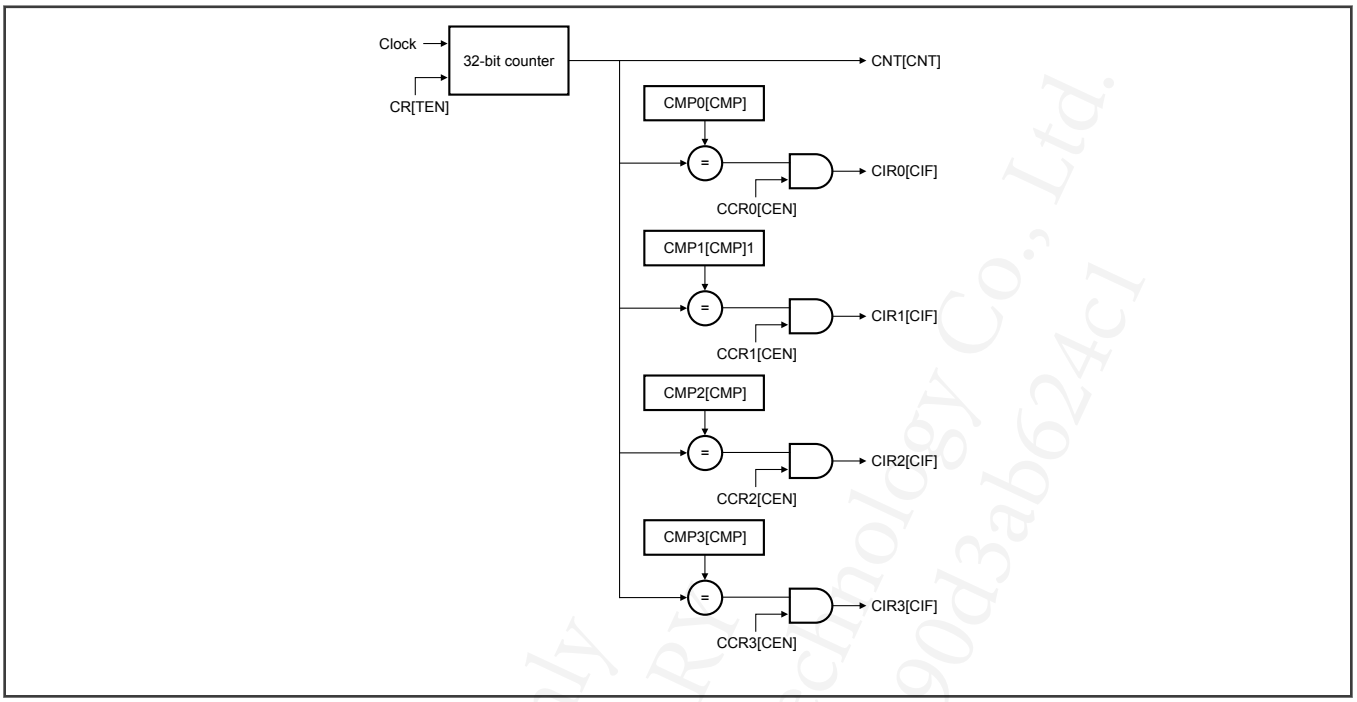
40.3 Features

STM has the following features:

- One 32-bit count-up timer with an 8-bit prescaler
- Four 32-bit compare channels
- An independent interrupt source for each channel
- Ability to stop the timer in Debug mode

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

40.4 STM block diagram



40.5 STM_TS register descriptions

The STM programming model allows only 32-bit (word) accesses. An attempted reference using a different size or to a reserved address generates a bus error termination.

40.5.1 STM_TS memory map

STM_TS base address: 4400_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	0000_0000h
4h	Count (CNT)	32	RW	0000_0000h
10h	Channel Control (CCR0)	32	RW	0000_0000h
14h	Channel Interrupt (CIR0)	32	W1C	0000_0000h
18h	Channel Compare (CMP0)	32	RW	0000_0000h
20h	Channel Control (CCR1)	32	RW	0000_0000h
24h	Channel Interrupt (CIR1)	32	W1C	0000_0000h
28h	Channel Compare (CMP1)	32	RW	0000_0000h
30h	Channel Control (CCR2)	32	RW	0000_0000h
34h	Channel Interrupt (CIR2)	32	W1C	0000_0000h
38h	Channel Compare (CMP2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
40h	Channel Control (CCR3)	32	RW	0000_0000h
44h	Channel Interrupt (CIR3)	32	W1C	0000_0000h
48h	Channel Compare (CMP3)	32	RW	0000_0000h

40.5.2 Control (CR)

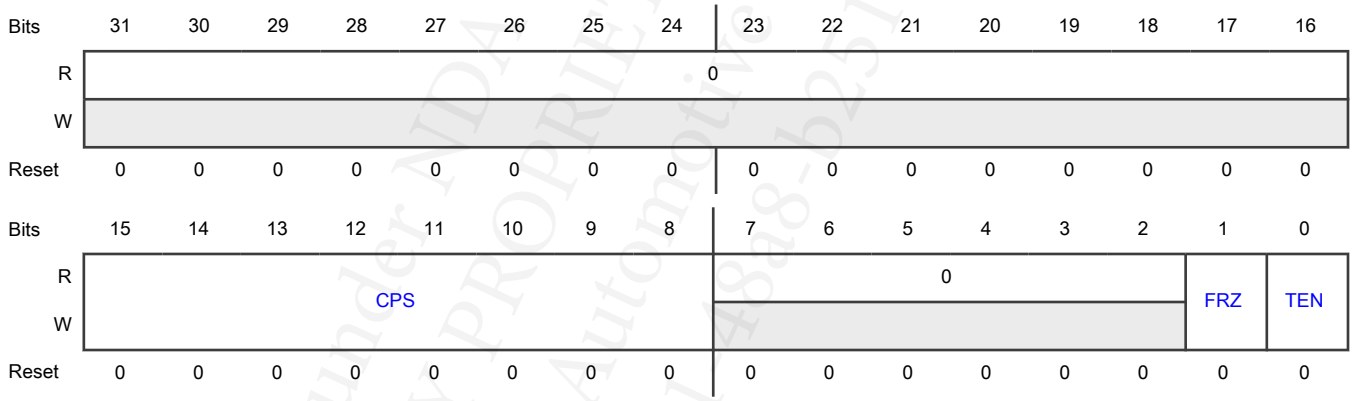
Offset

Register	Offset
CR	0h

Function

Contains fields for the prescale value, freeze control, and timer enable.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 CPS	Counter Prescaler Selects the module clock divide value for the prescaler (1–256). • 00h - Divide module clock by 1

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8316e2e5-4821-48a8-b251-390a3ab624c1

Table continued from the previous page...

Field	Function
	• 01h - Divide module clock by 2 • ... • FFh - Divide module clock by 256
7-2 —	Reserved
1 FRZ	Freeze Stops the timer when the chip enters Debug mode. **NOTE** When the chip enters Debug mode, it notifies STM, which in turn uses this field to determine timer operation. 0b - Timer runs in Debug mode 1b - Timer stops in Debug mode
0 TEN	Timer Enable Enables the module timer. 0b - Disabled 1b - Enabled

40.5.3 Count (CNT)

Offset

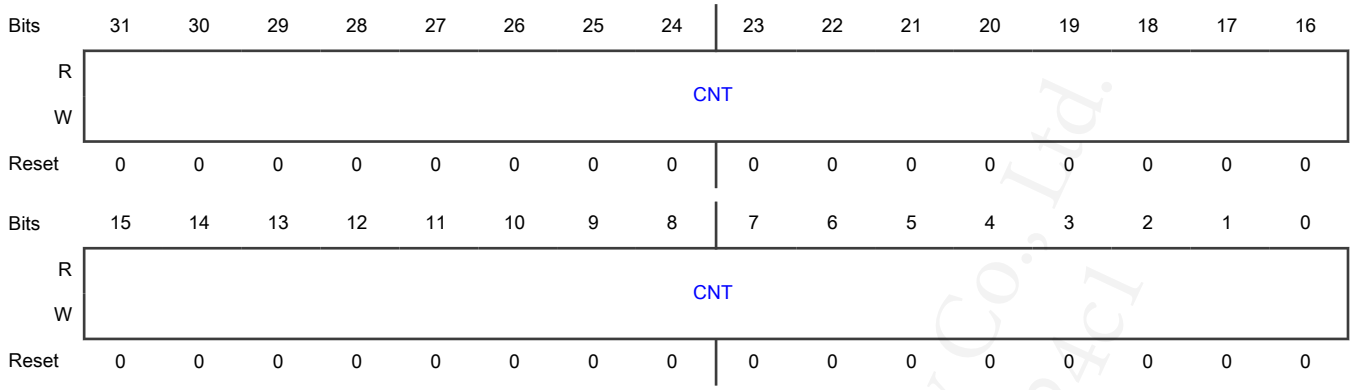
Register	Offset
CNT	4h

Function

Holds the timer count value.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 CNT	Timer Count The time base for all compare channels. When enabled, the timer count increments at the rate of the module clock divided by the prescale value.

40.5.4 Channel Control (CCR0 - CCR3)

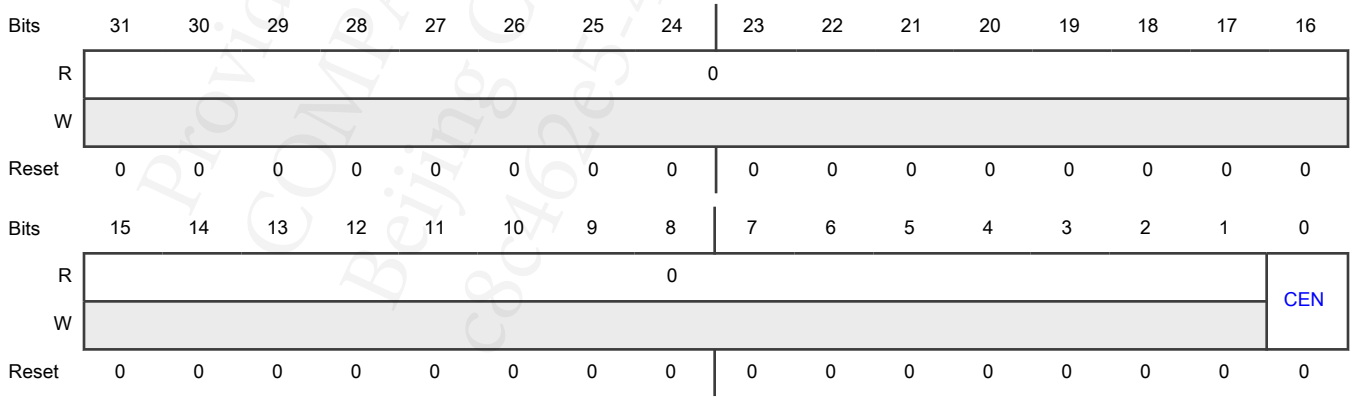
Offset

Register	Offset
CCR0	10h
CCR1	20h
CCR2	30h
CCR3	40h

Function

Enables channel n of the timer.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c84626c5-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-1 —	Reserved
0 CEN	Channel Enable 0b - Disabled 1b - Enabled

40.5.5 Channel Interrupt (CIR0 - CIR3)

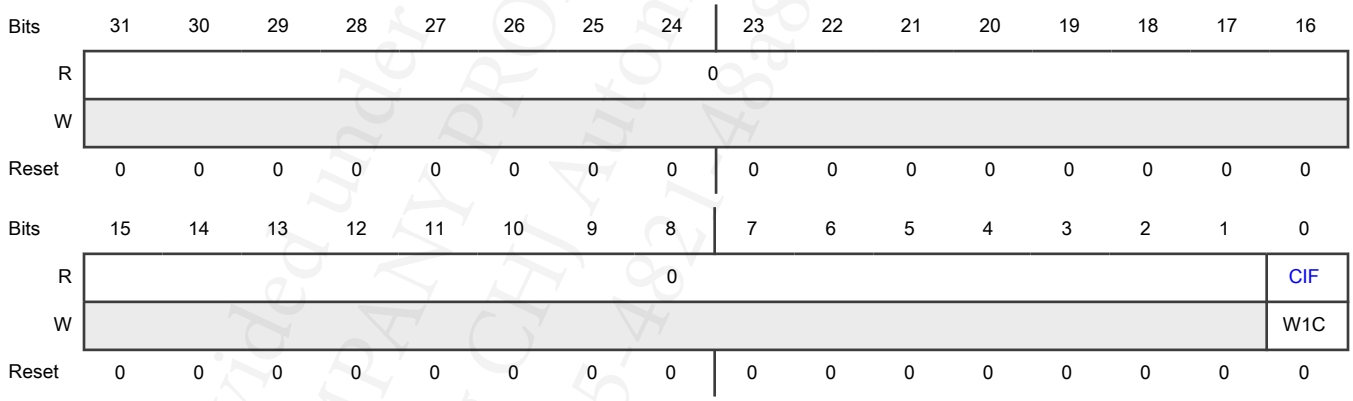
Offset

Register	Offset
CIR0	14h
CIR1	24h
CIR2	34h
CIR3	44h

Function

Indicates and clears the interrupt flag for channel n of the timer.

Diagram



Fields

Field	Function
31-1 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
0 CIF	Channel Interrupt Flag Indicates the channel IRQ is asserted due to a match on the channel. 0b - Read: IRQ is not asserted. Write: No effect. 1b - Read: IRQ is asserted. Write: Clear the flag.

40.5.6 Channel Compare (CMP0 - CMP3)

Offset

Register	Offset
CMP0	18h
CMP1	28h
CMP2	38h
CMP3	48h

Function

The compare value for channel n.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CMP															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CMP															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CMP	Channel Compare If the channel is enabled (CCR0[CEN]), when the timer count (CNT) matches this value, STM asserts the channel IRQ and sets the channel interrupt flag (CIR0[CIF]).

Provided under NDA only
COMPANY PROPRIETARY
Beijing CTI Automotive Technology Co., Ltd.
ca1625-4821-45a8-b251-590d3ab624c1

40.6 STM register descriptions

The STM programming model allows only 32-bit (word) accesses. An attempted reference using a different size or to a reserved address generates a bus error termination.

40.6.1 STM memory map

STM_0 base address: 4011_C000h

STM_1 base address: 4012_0000h

STM_2 base address: 4012_4000h

STM_3 base address: 4012_8000h

STM_4 base address: 4021_C000h

STM_5 base address: 4022_0000h

STM_6 base address: 4022_4000h

STM_7 base address: 4022_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	0000_0000h
4h	Count (CNT)	32	RW	0000_0000h
10h	Channel Control (CCR0)	32	RW	0000_0000h
14h	Channel Interrupt (CIR0)	32	W1C	0000_0000h
18h	Channel Compare (CMP0)	32	RW	0000_0000h
20h	Channel Control (CCR1)	32	RW	0000_0000h
24h	Channel Interrupt (CIR1)	32	W1C	0000_0000h
28h	Channel Compare (CMP1)	32	RW	0000_0000h
30h	Channel Control (CCR2)	32	RW	0000_0000h
34h	Channel Interrupt (CIR2)	32	W1C	0000_0000h
38h	Channel Compare (CMP2)	32	RW	0000_0000h
40h	Channel Control (CCR3)	32	RW	0000_0000h
44h	Channel Interrupt (CIR3)	32	W1C	0000_0000h
48h	Channel Compare (CMP3)	32	RW	0000_0000h

40.6.2 Control (CR)

Offset

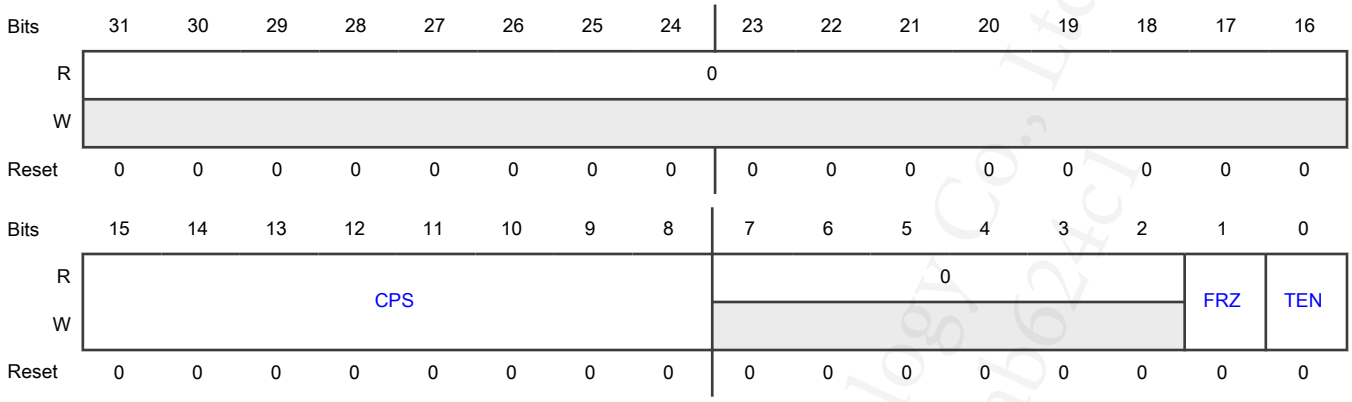
Register	Offset
CR	0h

Preliminary Draft
COMPANY CONFIDENTIAL
PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-1990ab624c1

Function

Contains fields for the prescale value, freeze control, and timer enable.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 CPS	Counter Prescaler Selects the module clock divide value for the prescaler (1–256). • 00h - Divide module clock by 1 • 01h - Divide module clock by 2 • ... • FFh - Divide module clock by 256
7-2 —	Reserved
1 FRZ	Freeze Stops the timer when the chip enters Debug mode. **NOTE** When the chip enters Debug mode, it notifies STM, which in turn uses this field to determine timer operation. 0b - Timer runs in Debug mode 1b - Timer stops in Debug mode
0 TEN	Timer Enable Enables the module timer.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled

40.6.3 Count (CNT)

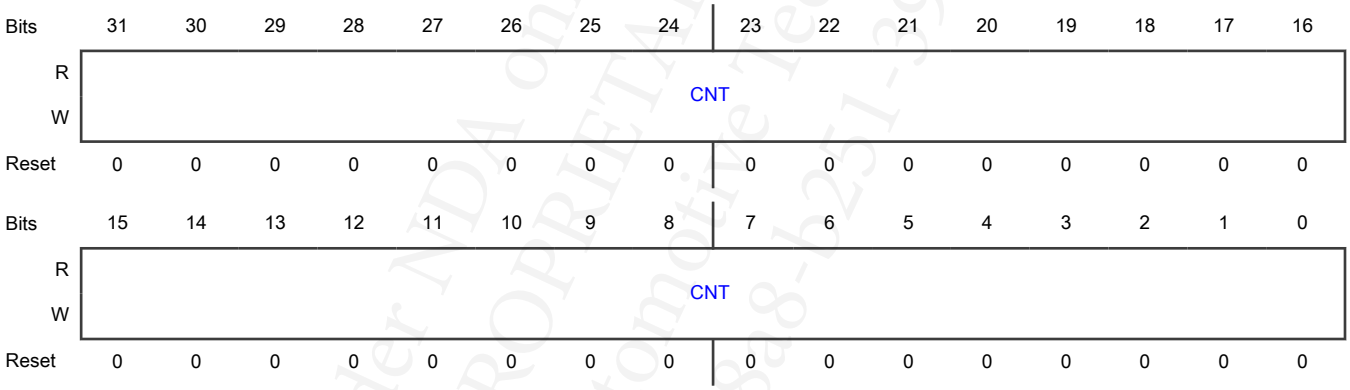
Offset

Register	Offset
CNT	4h

Function

Holds the timer count value.

Diagram



Fields

Field	Function
31-0 CNT	Timer Count The time base for all compare channels. When enabled, the timer count increments at the rate of the module clock divided by the prescale value.

40.6.4 Channel Control (CCR0 - CCR3)

Offset

Register	Offset
CCR0	10h

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESWIN Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-590d3ab624c1

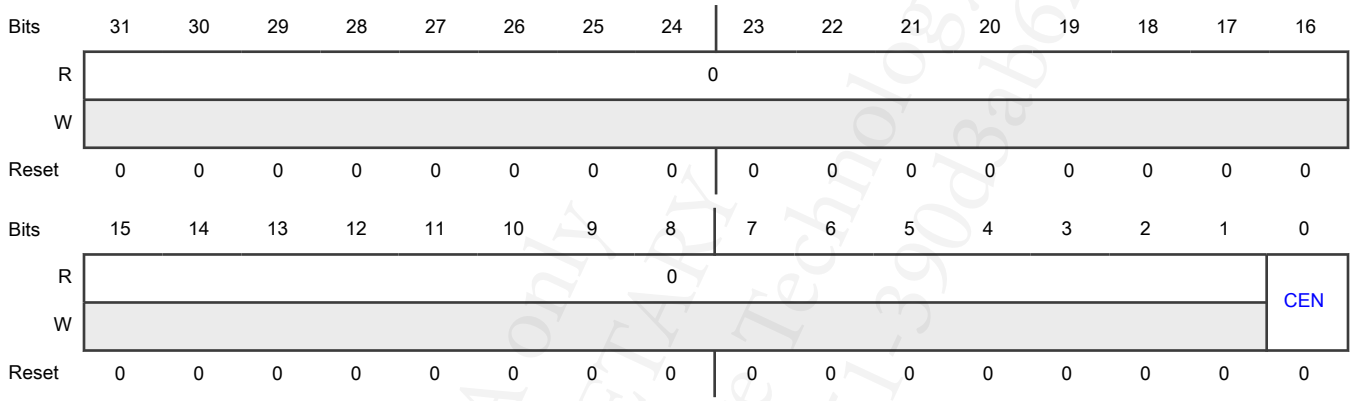
Table continued from the previous page...

Register	Offset
CCR1	20h
CCR2	30h
CCR3	40h

Function

Enables channel n of the timer.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CEN	Channel Enable 0b - Disabled 1b - Enabled

40.6.5 Channel Interrupt (CIR0 - CIR3)

Offset

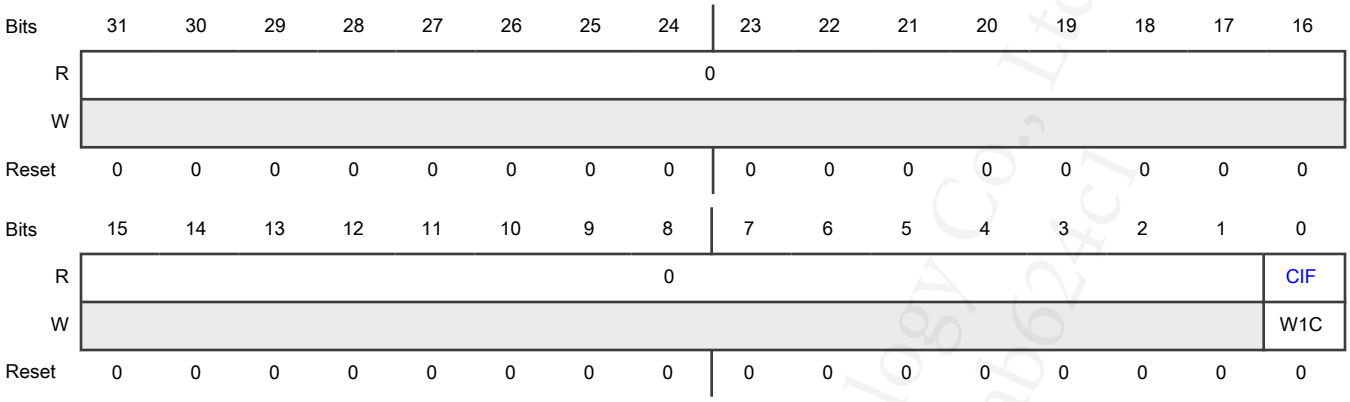
Register	Offset
CIR0	14h
CIR1	24h
CIR2	34h
CIR3	44h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Function

Indicates and clears the interrupt flag for channel n of the timer.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CIF	Channel Interrupt Flag Indicates the channel IRQ is asserted due to a match on the channel. 0b - Read: IRQ is not asserted. Write: No effect. 1b - Read: IRQ is asserted. Write: Clear the flag.

40.6.6 Channel Compare (CMP0 - CMP3)

Offset

Register	Offset
CMP0	18h
CMP1	28h
CMP2	38h
CMP3	48h

Function

The compare value for channel n.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48ad-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CMP															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMP															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 CMP	Channel Compare If the channel is enabled (CCR0[CEN]), when the timer count (CNT) matches this value, STM asserts the channel IRQ and sets the channel interrupt flag (CIR0[CIF]).

40.7 Configure the timer

1. Set the initial timer count (CNT[CNT]).
2. Specify STM behavior in chip Debug mode (CR[FRZ]).
3. Set the counter prescaler (CR[CPS]).
4. Start the timer (CR[TEN]).

40.8 Configure the compare channels

For each compare channel:

1. Set the channel compare value (CMP0[CMP]).
2. Enable the compare channel (CCR0[CEN]).

40.9 Respond to compare channel events

For each compare channel:

1. Check the channel interrupt flag (CIR0[CIF]).
2. If the channel interrupt flag is set, respond to the interrupt request.
3. When the channel interrupt has been handled, clear the channel interrupt flag (CIR0[CIF]).

40.10 Functional description

40.10.1 Count-up timer

STM has one 32-bit count-up timer that serves as the time base for four compare channels. When enabled, the counter increments at the module clock frequency divided by a prescaler value in the range from 1 to 256. When enabled in Normal mode, the timer increments continuously. The counter rolls over at FFFF_FFFFh to 0000_0000h with no restrictions at this boundary.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462c_4821-48a8-b251-390d3ab624c1

40.10.2 Compare channels

STM has four identical compare channels. Each channel includes a channel control register (CCR0), a channel interrupt register (CIR0), and a channel compare register (CMP0). When the channel is enabled and its channel compare value matches the timer count, STM sets the channel interrupt flag and generates an IRQ on that channel.

40.10.3 Behavior in different chip modes

STM supports the chip modes of operation as follows:

Chip mode	STM behavior
Normal	When the timer is enabled (CR[TEN] = 1), the timer counts up continuously.
Debug	If CR[FRZ] = 1, STM stops the timer. Otherwise, when the timer is enabled (CR[TEN] = 1), the timer counts up continuously.

40.11 Glossary

IRQ	Interrupt request

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 41
Software Watchdog Timer (SWT)

41.1 Chip-specific SWT information

41.1.1 SWT instances

This chip implements these SWTs. Each instance supports the indicated core.

- SWT_0: Cortex-M7 – core 0
- SWT_1: Cortex-M7 – core 1
- SWT_2: Cortex-M7 – core 2
- SWT_3: Cortex-A53 – core 0
- SWT_4: Cortex-A53 – core 1
- SWT_5: Cortex-A53 – core 2
- SWT_6: Cortex-A53 – core 3

NOTE

Stop mode in context to SWT refers to sleeping / WFI mode of core to which it is attached.

NOTE

For any operation to be performed on an SWT instance, its respective core must be enabled.

41.1.2 SWT time-out events: configuring reset and interrupt requests

For each SWT instance, configure the outcomes of SWT time-out events, as well as reactions to SWT reset and interrupt requests, using the Interrupt Then Reset Request field of the SWT Control Register (CR[ITR]). See related information in:

- "SWT0 timeout" section
- Interrupt Map file attached to this document
- Fault Map file attached to this document

41.1.3 Reset SWT only when a timeout triggers a reset request

The chapter describes that SWT has a feature that when a timeout triggers a reset request and the application does not require a chip reset, then you can make the SWT instance reset itself rather than the whole system.

In this chip, this feature is only available for SWT_1-SWT_6 when the corresponding reset is configured as interrupt in FCCU at the chip level. For SWT_0, this feature is not supported as it cannot be demoted to interrupt in MC_RGM.

41.2 Introduction

The Software Watchdog Timer (SWT) is a 32-bit window watchdog timer that enables the system to recover from situations such as:

- Software trapped in a loop
- A bus transaction failing to terminate

In regular operation, SWT requires periodic execution of a watchdog servicing operation. The servicing operation resets the timer to a specified timeout period. If this servicing action does not occur before the timer expires, SWT generates an interrupt or a hardware reset request.

You can configure SWT to generate a reset request or an interrupt on the initial timeout. SWT always generates a reset request on a second consecutive timeout.

41.2.1 Features

SWT has the following features:

- 32-bit countdown timer
- Selection of regular or window servicing
- Selection of reset request or interrupt on an initial timeout
- Selection of fixed or keyed service sequence
- Master access protection
- Hard and soft configuration lock bits

41.3 Initialize the SWT

To initialize the SWT, you must do the following:

- Select the clock source (if there are more than one possible sources).
- Set the timeout period.
- Control timeout behavior. There are two options:
  - Select reset request on timeout, or
  - Select interrupt on initial timeout.

You must intialize all registers before enabling SWT (CR[WEN]). You can initialize the registers in any sequence.

41.3.1 Select the clock source

See the chip-specific SWT information to find the clock source that drives the countdown timer.

41.3.2 Set the timeout period

When you enable SWT, it loads the greater of the specified watchdog timeout period or the minimum timeout period into an internal 32-bit countdown timer every time you perform a valid service operation.

To set the watchdog timeout period:

- Write the desired timeout period to TO[WTO].

The minimum timeout period is 256 clock cycles.

41.3.3 Control timeout behavior

SWT can respond to a timeout in one of two ways:

- Generate an immediate reset request on any timeout.
- Generate an interrupt and load the countdown timer on an inital timeout, then generate a reset request on a subsequent timeout.

41.3.3.1 Select reset request on timeout

To configure SWT to generate an immediate reset request on any timeout:

- Write 0 to CR[ITR].

pre-release NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c84625-4821-4588-9251-390d3ab624c1

41.3.3.2 Select interrupt on initial timeout

To configure SWT to generate an interrupt on an initial timeout:

- Write 1 to CR[ITR].

In this configuration, on the initial timeout:

- SWT loads the countdown timer with the timeout period (TO[WTO]).
- SWT indicates the interrupt with the timeout interrupt flag (IR[TIF]).
- If a second consecutive timeout occurs before writing the service sequence for the first timeout, SWT generates a reset request.

You clear the interrupt flag by writing 1 to IR[TIF].

41.3.4 Configure locking and unlocking

You can lock the SWT configuration with either a hard lock or a soft lock. When either lock is in effect, CR, TO, WN, and SK are read-only.

41.3.4.1 Enable the hard lock

To enable the hard lock:

- Write 1 to CR[HLK].

Hard lock is disabled only by a reset.

41.3.4.2 Enable the soft lock

To enable the soft lock:

- Write 1 to CR[SLK].

41.3.4.3 Unlock the soft lock

To unlock the soft lock:

1. Write C520h to SR[WSC].
2. Write D928h to SR[WSC].

This unlock sequence:

- Ignores service sequence writes.
- Recognizes the unlock sequence regardless of previous writes.
- Recognizes the unlock sequence regardless of the time between writes.
- Does not require CR[WEN] to be 1.

You can write this unlock sequence at any time.

NOTE

It is possible for a keyed service sequence to unlock soft lock. See Avoid accidental soft unlock for the procedure to handle this situation.

41.3.4.4 Avoid accidental soft unlock

When SWT operates in keyed sequence service, the sequence of service keys generated by the pseudorandom generator includes the unlock keys for soft lock. If one or more service routines use both unlock keys in the proper order, 0xC520 followed at any future time by 0xD928, SWT clears soft lock (CR[SLK] = 0). The unlock sequence logic ignores any service keys other than

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c462e5-4d21-48a8-b251-390d3ab624c1

the unlock keys, so the unlock keys don't have to be inserted consecutively. If the second unlock key is also the second key of a service operation, unlock occurs before the service operation completes.

To avoid accidental unlock:

If the service routine writes a key value of C520h to SR[WSC] as a key of the service operation, then do the following:

1. Complete the service operation.
2. Complete the unlock sequence by writing the second unlock key, D928h, to SR[WSC].
3. Reinitiate the soft lock (CR[SLK] = 1).

41.3.5 Select behavior on an invalid access

SWT can respond to an invalid access one of two ways:

- Generate a bus error.
- Generate a bus error and, if SWT is enabled, a reset request.

To select how SWT responds to an invalid access:

- Write the appropriate value to CR[RIA].

41.4 Enable the SWT

To enable the SWT:

- Write 1 to CR[WEN].

41.5 Initiate service operations

When enabled, SWT requires periodic execution of a servicing operation.

SWT can operate in one of the following service sequence modes:

- Fixed service sequence
- Keyed service sequence

41.5.1 Initiate a fixed service sequence

To initiate the fixed service sequence:

1. Select the fixed servicing mode (write 00b to CR[SMD]).
2. Write A602h to SR[WSC].
3. Write B480h to SR[WSC].

There is no timing requirement between the two writes. The service sequence logic ignores unlock-sequence writes.

41.5.2 Initiate a keyed service sequence

To initiate the keyed service sequence:

1. Select the keyed servicing mode (write 01b to CR[SMD]).
2. Read the next service key from SK[SK].
3. Write that service key to SR[WSC].
4. Repeat steps 2 and 3 one time.

See Service key generation for information about service keys.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4821-48a8-b251-390d3ab624c1

41.5.3 Select window service mode

In window service mode, you must write the service sequence only when the watchdog timer is less than the window start value (WN[WST]). If you write the service sequence outside of this window, the access is invalid and generates a bus error or reset request depending on the Reset on Invalid Access setting (CR[RIA]).

For example, if the timeout period is 5000 and the window start value is 1000, you must write the service sequence during the last 20% of the timeout period.

Synchronization logic in SWT causes a slight delay in the opening of the window. This delay can be up to three system clock cycles plus four counter clock cycles.

To select window service mode:

- Write 1 to (CR[WND].

41.6 Reset only the SWT

When a timeout triggers a reset request and the application does not require a system reset, you can make the SWT instance reset itself rather than the whole system.

To make a timeout reset request trigger only an SWT reset (rather than the whole system), write 1 to the Reset Request Flag (RRR[RRF]).

This action causes SWT to:

- Clear the reset request.
- If an interrupt request also occurred, clear the interrupt request (IR[TIF]).
- Reset and restart the SWT.

For systems with multiple SWT instances, this feature allows you to reset an individual SWT instance, avoiding the widespread impact of a system reset.

41.7 Test SWT operation

When you disable SWT, it loads the current countdown timer into CO[CNT]. When you enable SWT, it clears this register. You can use CO[CNT] to perform a software self-test of SWT.

To test SWT operation:

1. Enable the SWT (CR[WEN] = 1).
2. Do not service SWT for a fixed period of time that is less than the timeout value.
3. Disable the SWT (CR[WEN] = 0).
4. Read the value in CO[CNT] to determine whether the internal countdown timer is working properly.

NOTE

The value shown in CO[CNT] can lag behind the actual internal timer up to six system clock cycles plus eight counter clock cycles.

41.8 Functional description

41.8.1 Behavior in different chip modes

SWT supports the chip modes of operation as follows:

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-390d3ab624c1

Chip mode	SWT behavior
Normal	When SWT is enabled (CR[WEN] = 1), the SWT timer counts down continuously.
Debug	If CR[FRZ] = 1, SWT stops the timer; otherwise, the timer continues to run.
Stop or Standby	If CR[STP] = 1, SWT stops the timer; otherwise, the timer continues to run.

41.8.2 Register access latency

Accesses to SWT registers occur with no peripheral bus wait states. The switch fabric may add one or more system wait states. However, due to synchronization logic in the SWT design, recognition of the service sequence or configuration changes may require up to three system clock cycles plus seven counter clock cycles.

41.8.3 Service key generation

SWT generates service keys with the pseudorandom key generator defined by the following equation.

$$SK_{n+1} = (17 \times SK_n + 3) \bmod 2^{16}$$

Figure 227. Pseudorandom key generator

This algorithm generates a sequence of 2^{16} different key values before repeating. In keyed service sequence mode, each time you write a valid key to SR[WSC], SWT updates SK[SK] with the next key.

For example, if the previous service key (SK[SK]) is 100h, then the next service sequence keys are 1103h followed by 2136h.

41.9 SWT register descriptions

The SWT programming model allows only 32-bit (word) accesses.

Any of the following attempted accesses are invalid:

- Non-32-bit accesses
- Writes to read-only registers
- Writes of incorrect values to SR when SWT is enabled
- Accesses to reserved addresses
- Accesses by masters without permission

You control how SWT responds to an invalid access with CR[RIA].

41.9.1 SWT memory map

SWT_0 base address: 4010_0000h

SWT_1 base address: 4010_4000h

SWT_2 base address: 4010_8000h

SWT_3 base address: 4010_C000h

SWT_4 base address: 4020_0000h

SWT_5 base address: 4020_4000h

SWT_6 base address: 4020_8000h

Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-482-48a8-b251-390d3a6624 COMPANY PROPRIETARY Under NDA only

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CR)	32	RW	FF00_010Ah
4h	Interrupt (IR)	32	W1C	0000_0000h
8h	Timeout (TO)	32	RW	0073_FDE0h
Ch	Window (WN)	32	RW	0000_0000h
10h	Service (SR)	32	WO	0000_0000h
14h	Counter Output (CO)	32	RO	0000_0000h
18h	Service Key (SK)	32	RW	0000_0000h
1Ch	Reset Request (RRR)	32	W1C	0000_0000h

41.9.2 Control (CR)

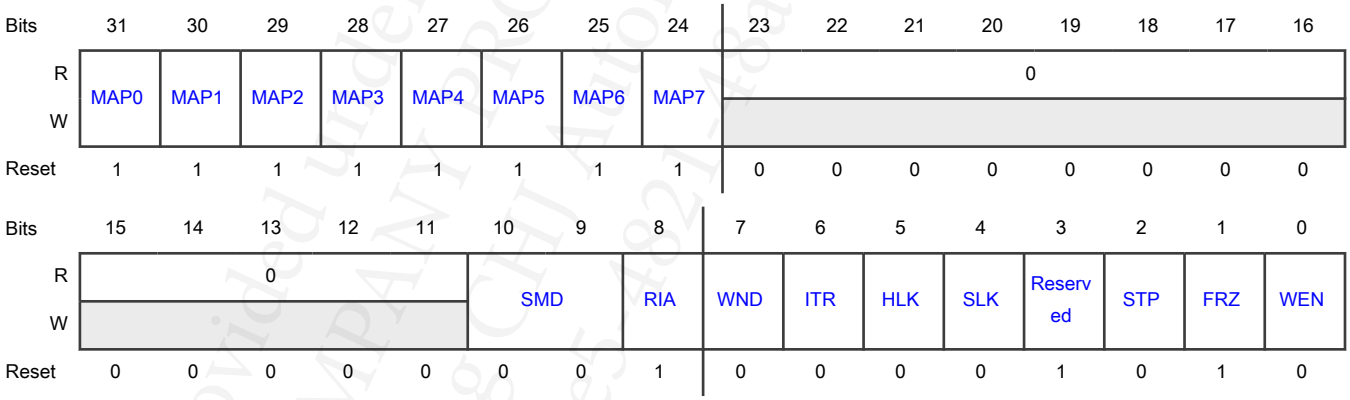
Offset

Register	Offset
CR	0h

Function

Contains fields for configuring and controlling SWT. The register is read-only if either hard lock or soft lock is enabled (either HLK or SLK is 1).

Diagram



Fields

Field	Function
31	Master Access Protection 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
MAP0	The platform bus master assignments are chip-specific. The number of this field corresponds to the XRDC DIDs of the bus master. MAP0 corresponds to XRDC DIDs 0 and 8. 0b - Access disabled 1b - Access enabled
30 MAP1	Master Access Protection 1 See the description for MAP0. MAP1 corresponds to XRDC DIDs 1 and 9. 0b - Access disabled 1b - Access enabled
29 MAP2	Master Access Protection 2 See the description for MAP0. MAP2 corresponds to XRDC DIDs 2 and 10. 0b - Access disabled 1b - Access enabled
28 MAP3	Master Access Protection 3 See the description for MAP0. MAP3 corresponds to XRDC DIDs 3 and 11. 0b - Access disabled 1b - Access enabled
27 MAP4	Master Access Protection 4 See the description for MAP0. MAP4 corresponds to XRDC DIDs 4 and 12. 0b - Access disabled 1b - Access enabled
26 MAP5	Master Access Protection 5 See the description for MAP0. MAP5 corresponds to XRDC DIDs 5 and 13. 0b - Access disabled 1b - Access enabled
25 MAP6	Master Access Protection 6 See the description for MAP0. MAP6 corresponds to XRDC DIDs 6 and 14. 0b - Access disabled 1b - Access enabled
24 MAP7	Master Access Protection 7 See the description for MAP0. MAP7 corresponds to XRDC DIDs 7 and 15.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Access disabled 1b - Access enabled
23-11 —	Reserved
10-9 SMD	Service Mode 00b - To service the watchdog, write the fixed sequence 0xA602, 0xB480 to SR. 01b - To service the watchdog, write two pseudorandom key values to SR. 10b - Do not use this value. Writing this value can cause the watchdog to not be serviced. 11b - Do not use this value. Writing this value can cause the watchdog not to be serviced.
8 RIA	Reset on Invalid Access Controls how SWT responds to an invalid access. **NOTE** For a description of how this chip implements SWT reset requests resulting from SWT invalid accesses, see the chip-specific SWT information. 0b - Generate a bus error 1b - If SWT is enabled (WEN = 1), generate a bus error and a reset request; otherwise, generate a bus error.
7 WND	Window Mode Specify regular or window mode operation. 0b - Can execute service sequence at any time 1b - Can execute service sequence only when the timeout counter is less than the value in WN
6 ITR	Interrupt Then Reset Request **NOTE** For a description of how this chip implements SWT reset requests and interrupt requests resulting from SWT timeouts, see the chip-specific SWT information. 0b - Generate a reset request on a timeout 1b - Generate an interrupt on an initial timeout; generate a reset request on a second consecutive timeout
5 HLK	Hard Lock Indicates that the hard lock is enabled. You cannot directly write 0 to this field. This field becomes 0 only after a reset. 0b - CR, TO, WN, and SK are read/write registers if SLK is also 0 1b - CR, TO, WN, and SK are read-only registers

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 SLK	Soft Lock Indicates that the soft lock is enabled. You cannot directly write 0 to this field. Clear this field by writing the unlock sequence to the service register. 0b - CR, TO, WN, and SK are read/write registers if HLK is also 0 1b - CR, TO, WN, and SK are read-only registers
3 —	Reserved
2 STP	Stop Mode Control Controls the watchdog timer when the chip enters Stop or Standby mode. 0b - Timer continues 1b - Timer stops
1 FRZ	Debug Mode Control Controls the watchdog timer when the chip enters Debug mode. 0b - Timer continues 1b - Timer stops
0 WEN	Watchdog Enable Enables or disables SWT. The reset value is 0. Therefore, after reset, you must enable SWT to start the countdown timer. 0b - Disabled 1b - Enabled

41.9.3 Interrupt (IR)

Offset

Register	Offset
IR	4h

Function

The timeout interrupt flag.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															TIF
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 TIF	Timeout Interrupt Flag Write 1 to this field to clear the flag and interrupt. Writing a 0 has no effect. 0b - No interrupt request 1b - Interrupt request due to an initial timeout

41.9.4 Timeout (TO)

Offset

Register	Offset
TO	8h

Function

Contains the 32-bit timeout period. The register is read-only if either hard lock or soft lock is enabled (CR[HLK] or CR[SLK] is 1).

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co., Ltd.
c8c462e5-4621-48a8-b251-390d8ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WTO															
Reset	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WTO															
Reset	1	1	1	1	1	1	0	1	1	1	1	0	0	0	0	0

Fields

Field	Function
31-0 WTO	Watchdog Timeout Watchdog timeout period in clock cycles. When software writes a service sequence or enables SWT, SWT loads the internal 32-bit countdown timer with this value or 100h, whichever is greater.

41.9.5 Window (WN)

Offset

Register	Offset
WN	Ch

Function

Contains the 32-bit window start value. SWT clears this register on reset. The register is read-only if either hard lock or soft lock is enabled (CR[HLK] or CR[SLK] is 1).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	WST															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	WST															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-39d3ab624c1

Fields

Field	Function
31-0 WST	Window Start Value When you enable window mode (CR[WND]), you can write the service sequence only when the internal timer is less than this value.

41.9.6 Service (SR)

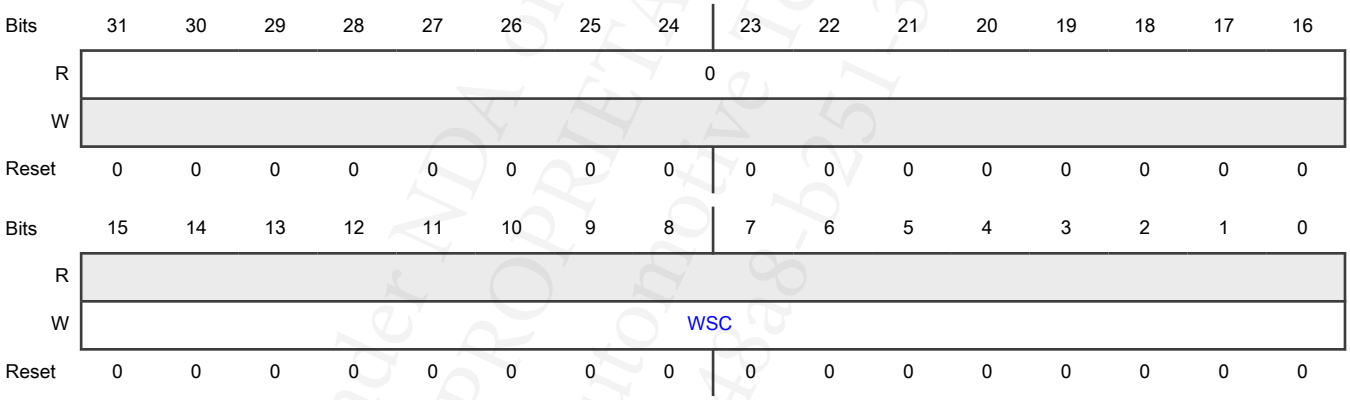
Offset

Register	Offset
SR	10h

Function

Initiates the service operation and resets the watchdog timer.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 WSC	Watchdog Service Code Use this field to service the watchdog and to unlock the Soft Lock (CR[SLK]). To service the watchdog: If SWT is in keyed service mode (CR[SMD]), write two pseudorandom key values to WSC (see Service key generation for details). Otherwise, write the following values to WSC, in the order shown:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1. A602h 2. B480h To unlock the Soft Lock (CR[SLK]), write the following values to WSC, in the order shown: 1. C520h 2. D928h When read, WSC always returns zero.

41.9.7 Counter Output (CO)

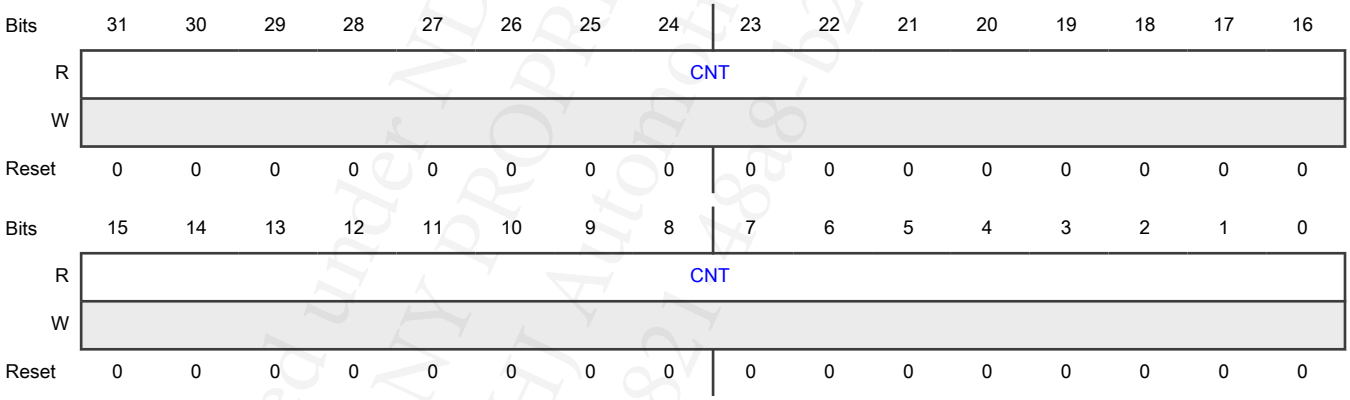
Offset

Register	Offset
CO	14h

Function

Shows the value of the internal timer when SWT is disabled.

Diagram



Fields

Field	Function
31-0 CNT	Watchdog Count When SWT is disabled (CR[WEN] is 0), CNT shows the value of the internal timer. When SWT is enabled (CR[WEN] is 1), it writes 0 to CNT. Values in this field can lag behind the internal timer value up to six system clock cycles plus eight counter clock cycles. Therefore, the CNT value that is read immediately after disabling SWT may be higher than the actual value of the internal timer.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cc625e5-4821-48a8-b251-390d3ab624c1

41.9.8 Service Key (SK)

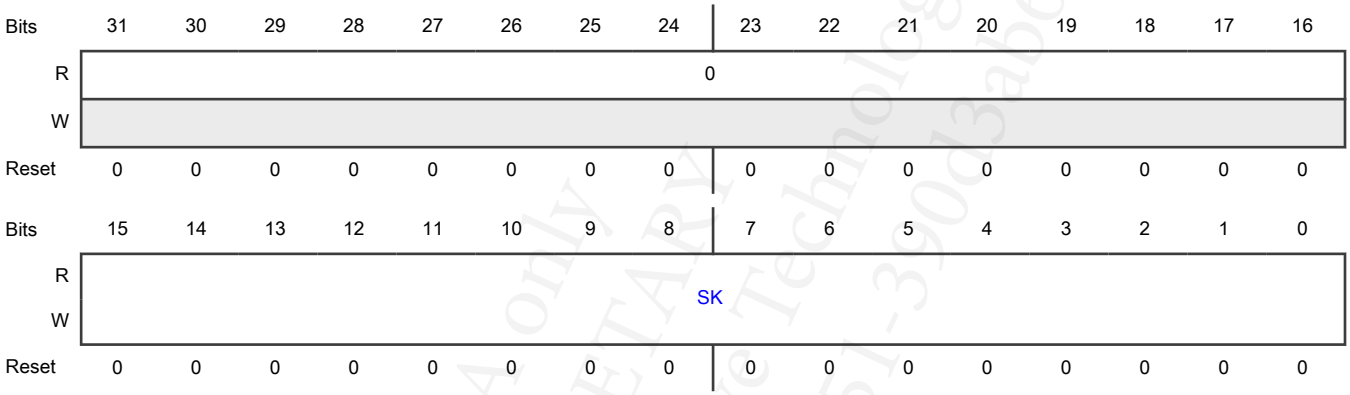
Offset

Register	Offset
SK	18h

Function

Holds the previous (or initial) service key value. This register is read-only if either hard lock or soft lock is enabled (CR[HLK] or CR[SLK] is 1).

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 SK	Service Key Holds the previous (or initial) service key value used in Initiate a keyed service sequence. If CR[SMD] is 01b, the next key value to write to SR is (17 * SK + 3) mod 2^{16}.

41.9.9 Reset Request (RRR)

Offset

Register	Offset
RRR	1Ch

Function

Contains the timeout reset request flag.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4824-48a8-b251-39003ab62401

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															RRF
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 RRF	Reset Request Flag Write 1 to clear the flag and request. Writing 0 has no effect. 0b - No reset request 1b - Any reset request initiated

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 42
Periodic Interrupt Timer (PIT)

42.1 Chip-specific PIT information

42.1.1 PIT module instance details

In this chip, in addition to timer channels for raising interrupts and triggering DMA channels, the PIT module is the source of:

- The external time tick for the FlexCAN free-running timer
- The external OS tick for the FlexRay stopwatch counter

42.1.2 PIT triggers

The following table shows the connections between the outputs of the PIT modules and other peripherals.

The PIT modules are connected to separate peripheral interface modules.

Timer	Channel	Peripheral	Function
PIT_0	0	DMAMUX_0	Periodic trigger input 0
PIT_0	1	DMAMUX_0	Periodic trigger input 1
PIT_0	2	DMAMUX_1	Periodic trigger input 0
PIT_0	3	DMAMUX_1	Periodic trigger input 1
PIT_0	4	FlexRay	External OS tick for stopwatch counter (configured by SRC_0 FLEXRAY_OS_TICK_INPUT_SELECT_REG)
PIT_0	5	FlexCAN_0, FlexCAN_1	External time tick for free-running timer
PIT_0	6	CTU	Trigger of CTU PWM
PIT_1	0	DMAMUX_2	Periodic trigger input 0
PIT_1	1	DMAMUX_2	Periodic trigger input 1
PIT_1	2	DMAMUX_3	Periodic trigger input 0
PIT_1	3	DMAMUX_3	Periodic trigger input 1
PIT_1	4	FlexCAN_2, FlexCAN_3	External time tick for free-running timer
PIT_1	5	—	—

42.2 Introduction

The following sections show the PIT's:

- Block diagram
- Features

42.2.1 Block diagram

The following figure shows the block diagram of PIT module.

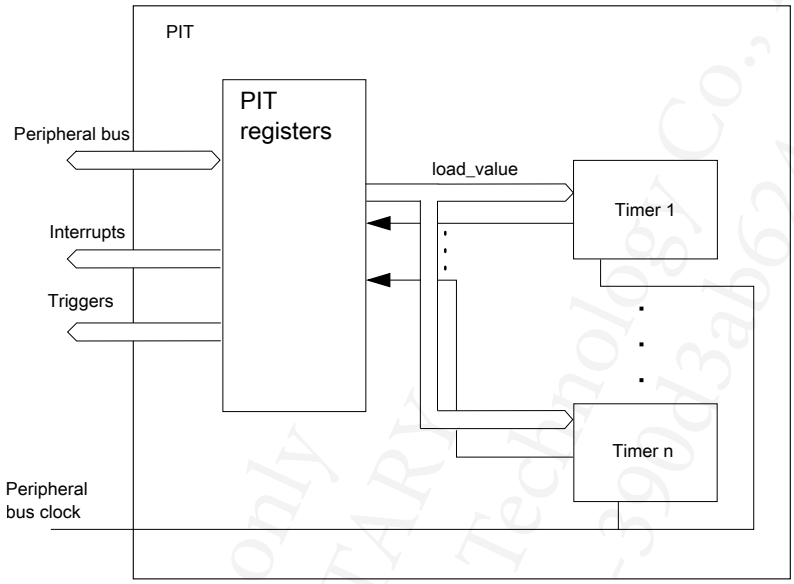


Figure 228. Block diagram of PIT

42.2.2 Features

The key features of the module are these:

- Ability of timers to generate DMA trigger pulses
- Ability of timers to generate interrupts
- Maskable interrupts
- Independent timeout periods for each timer
- All the timers use a downcounter

42.3 Modes of operation

This subsection briefly describes all operating modes supported by PIT.

- Run mode

  All functional parts of the PIT are running during normal Run mode.

42.4 PIT register descriptions

This section provides a detailed description of all registers accessible in the PIT module.

- See the chip-specific PIT information for the number of PIT channels used in this MCU.

NOTE

When a chip has more than one instance of the PIT, the different instances might have different feature and register sets. Please see the configuration information.

42.4.1 PIT memory map

PIT_0 base address: 4018_8000h

PIT_1 base address: 4028_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PIT Module Control Register (MCR)	32	RW	0000_0000h
E0h	PIT Upper Lifetime Timer Register (LTMR64H)	32	RO	0000_0000h
E4h	PIT Lower Lifetime Timer Register (LTMR64L)	32	RO	0000_0000h
100h	Timer Load Value Register (LDVAL0)	32	RW	0000_0000h
104h	Current Timer Value Register (CVAL0)	32	RO	0000_0000h
108h	Timer Control Register (TCTRL0)	32	RW	0000_0000h
10Ch	Timer Flag Register (TFLG0)	32	W1C	0000_0000h
110h	Timer Load Value Register (LDVAL1)	32	RW	0000_0000h
114h	Current Timer Value Register (CVAL1)	32	RO	0000_0000h
118h	Timer Control Register (TCTRL1)	32	RW	0000_0000h
11Ch	Timer Flag Register (TFLG1)	32	W1C	0000_0000h
120h	Timer Load Value Register (LDVAL2)	32	RW	0000_0000h
124h	Current Timer Value Register (CVAL2)	32	RO	0000_0000h
128h	Timer Control Register (TCTRL2)	32	RW	0000_0000h
12Ch	Timer Flag Register (TFLG2)	32	W1C	0000_0000h
130h	Timer Load Value Register (LDVAL3)	32	RW	0000_0000h
134h	Current Timer Value Register (CVAL3)	32	RO	0000_0000h
138h	Timer Control Register (TCTRL3)	32	RW	0000_0000h
13Ch	Timer Flag Register (TFLG3)	32	W1C	0000_0000h
140h	Timer Load Value Register (LDVAL4)	32	RW	0000_0000h
144h	Current Timer Value Register (CVAL4)	32	RO	0000_0000h
148h	Timer Control Register (TCTRL4)	32	RW	0000_0000h
14Ch	Timer Flag Register (TFLG4)	32	W1C	0000_0000h
150h	Timer Load Value Register (LDVAL5)	32	RW	0000_0000h
154h	Current Timer Value Register (CVAL5)	32	RO	0000_0000h
158h	Timer Control Register (TCTRL5)	32	RW	0000_0000h
15Ch	Timer Flag Register (TFLG5)	32	W1C	0000_0000h
160h	Timer Load Value Register (LDVAL6)	32	RW	0000_0000h
164h	Current Timer Value Register (CVAL6)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
168h	Timer Control Register (TCTRL6)	32	RW	0000_0000h
16Ch	Timer Flag Register (TFLG6)	32	W1C	0000_0000h

42.4.2 PIT Module Control Register (MCR)

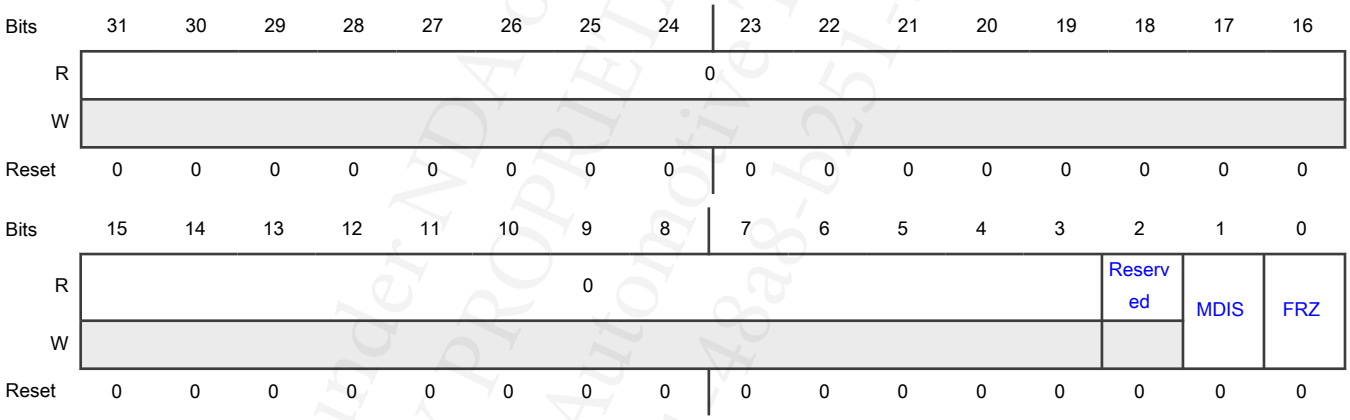
Offset

Register	Offset
MCR	0h

Function

This register enables or disables the PIT timer clocks and controls the timers when the PIT enters the Debug mode.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 —	Reserved
1 MDIS	Module Disable for PIT Disables the standard timers. You must write 1 to this field before setting up anything.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
	0b - Clock for standard PIT timers is enabled. 1b - Clock for standard PIT timers is disabled.
0 FRZ	Freeze Allows the timers to be stopped when the device enters the Debug mode. 0b - Timers continue to run in Debug mode. 1b - Timers are stopped in Debug mode.

42.4.3 PIT Upper Lifetime Timer Register (LTMR64H)

Offset

Register	Offset
LTMR64H	E0h

Function

This register is intended for applications that chain timer 0 and timer 1 to build a 64-bit Lifetime Timer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LTH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LTH															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 LTH	Life Timer value Shows the timer value of timer 1. If this register is read at a time t1, LTMR64L shows the value of timer 0 at time t1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c4625-4821-48a8-1251-390d3ab624c1

42.4.4 PIT Lower Lifetime Timer Register (LTMR64L)

Offset

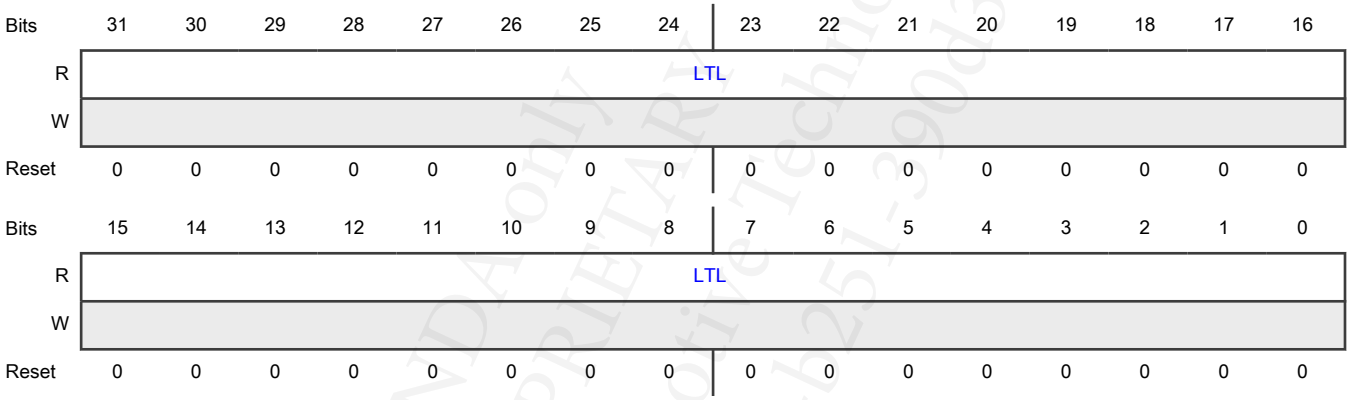
Register	Offset
LTMR64L	E4h

Function

This register is intended for applications that chain timer 0 and timer 1 to build a 64-bit Lifetime Timer.

To use LTMR64H and LTMR64L, timer 0 and timer 1 need to be chained. To obtain the correct value, first read LTMR64H and then LTMR64L. The value for the LTMR64H register is set to CVAL1 register at the time of the first access and the value of the LTMR64L register is set to CVAL0 register at first access. Therefore, the application is not affected by the carry-over effects of the running counter.

Diagram



Fields

Field	Function
31-0 LTL	Life Timer value Shows the value of timer 0 at the time LTMR64H was last read. It will only update if LTMR64H is read.

42.4.5 Timer Load Value Register (LDVAL0 - LDVAL6)

Offset

Register	Offset
LDVAL0	100h
LDVAL1	110h
LDVAL2	120h
LDVAL3	130h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
LDVAL4	140h
LDVAL5	150h
LDVAL6	160h

Function

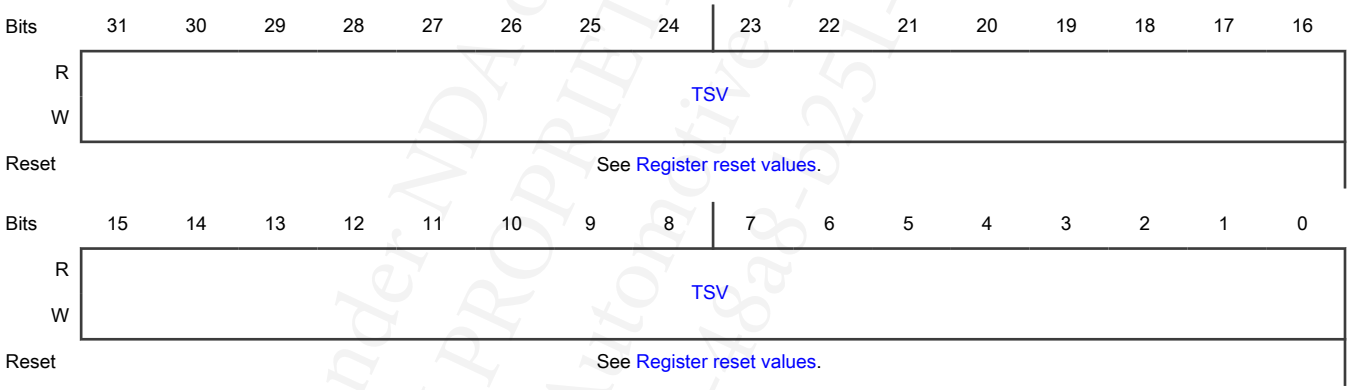
These registers select the timeout period for the timer interrupts.

NOTE

Each module instance supports a different number of registers.

Register supported	Register not supported
PIT_0_LDVAL0–LDVAL6	—
PIT_1_LDVAL0–LDVAL5	PIT_1_LDVAL6

Diagram



Register reset values

Register	Reset value
LDVAL0–LDVAL5	PIT_0,PIT_1: 0000_0000h
LDVAL6	0000_0000h

Fields

Field	Function
31-0	Timer Start Value

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-9251-390d3b624c1

Field	Function
TSV	Sets the timer start value. The timer counts down until it reaches 0, then generates an interrupt and loads this register value again. Writing a new value to this register does not restart the timer; instead the value is loaded after the timer expires. To abort the current cycle and start a timer period with the new value, the timer must be disabled and enabled again.

42.4.6 Current Timer Value Register (CVAL0 - CVAL6)

Offset

Register	Offset
CVAL0	104h
CVAL1	114h
CVAL2	124h
CVAL3	134h
CVAL4	144h
CVAL5	154h
CVAL6	164h

Function

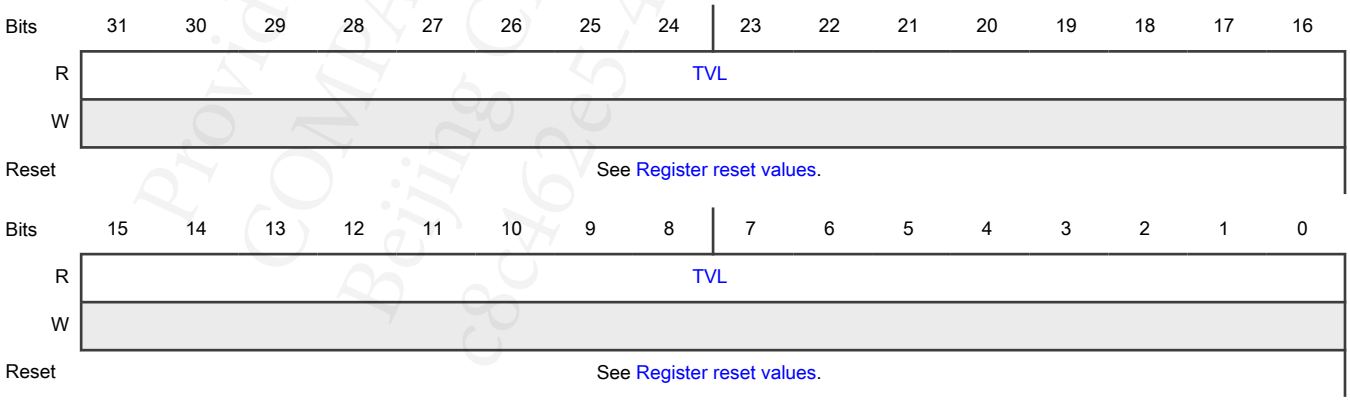
These registers indicate the current timer position.

NOTE

Each module instance supports a different number of registers.

Register supported	Register not supported
PIT_0_CVAL0–CVAL6	—
PIT_1_CVAL0–CVAL5	PIT_1_CVAL6

Diagram



Register reset values

Register	Reset value
CVAL0–CVAL5	PIT_0,PIT_1: 0000_0000h
CVAL6	0000_0000h

Fields

Field	Function
31-0 TVL	Current Timer Value Represents the current timer value, if the timer is enabled. **NOTE** • If the timer is disabled, do not use this field because its value is unreliable. • The timer uses a downcounter. The timer values are frozen in the Debug mode if MCR[FRZ] is set.

42.4.7 Timer Control Register (TCTRL0 - TCTRL6)

Offset

Register	Offset
TCTRL0	108h
TCTRL1	118h
TCTRL2	128h
TCTRL3	138h
TCTRL4	148h
TCTRL5	158h
TCTRL6	168h

Function

These registers contain the control bits for each timer.

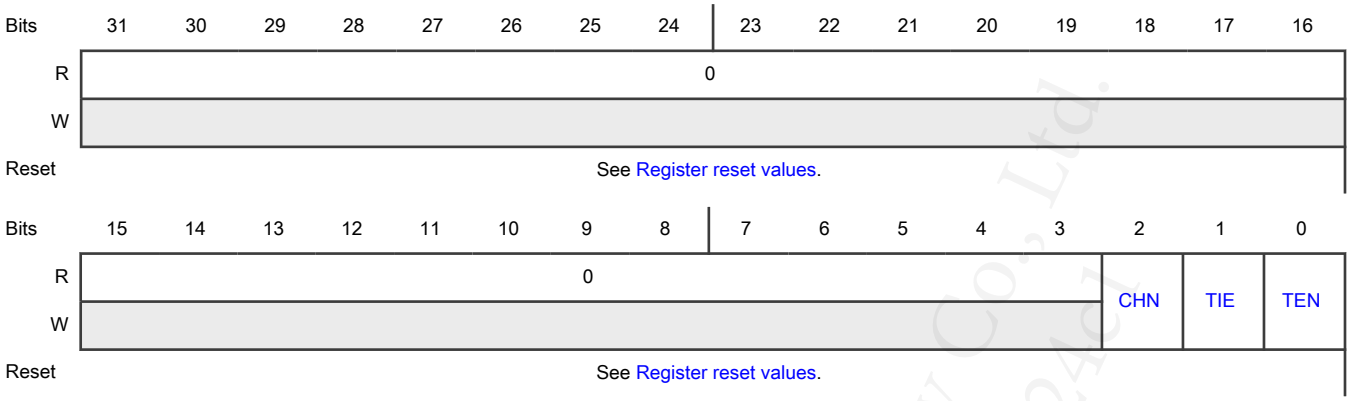
NOTE

Each module instance supports a different number of registers.

Register supported	Register not supported
PIT_0_TCTRL0–TCTRL6	—
PIT_1_TCTRL0–TCTRL5	PIT_1_TCTRL6

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3b624c1

Diagram



Register reset values

Register	Reset value
TCTRL0–TCTRL5	PIT_0,PIT_1: 0000_0000h
TCTRL6	0000_0000h

Fields

Field	Function
31-3 —	Reserved
2 CHN	Chain Mode When activated, timer n-1 needs to expire before timer n (n is > 0) can decrement by 1. Timer 0 cannot be chained. 0b - Timer is not chained. 1b - Timer is chained to a previous timer. For example, for channel 2, if this field is set, Timer 2 is chained to Timer 1.
1 TIE	Timer Interrupt Enable When an interrupt is pending, or if TFLGn[TIF] is set, enabling the interrupt causes an interrupt event. To avoid this, the associated TFLGn[TIF] must be cleared first. 0b - Interrupt requests from Timer n are disabled. 1b - Interrupt is requested whenever TIF is set.
0 TEN	Timer Enable Enables or disables the timer. 0b - Timer n is disabled. 1b - Timer n is enabled.

Provided under NDA only
COMPANY PROPRIETARY
Beijing ChiU Automotive Technology Co., Ltd.
8524623-821 48a8-b251-390d3ab624c1

42.4.8 Timer Flag Register (TFLG0 - TFLG6)

Offset

Register	Offset
TFLG0	10Ch
TFLG1	11Ch
TFLG2	12Ch
TFLG3	13Ch
TFLG4	14Ch
TFLG5	15Ch
TFLG6	16Ch

Function

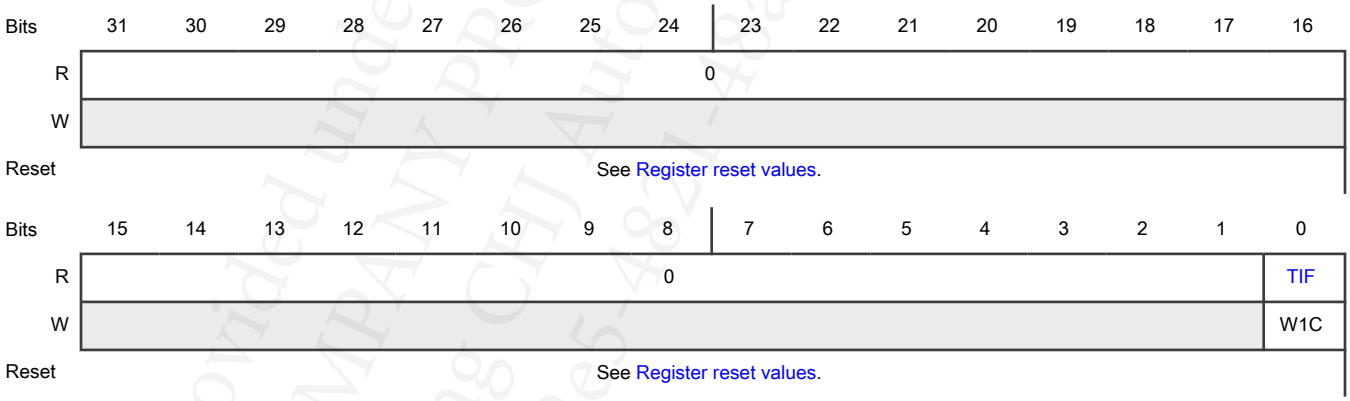
These registers hold the PIT interrupt flags.

NOTE

Each module instance supports a different number of registers.

Register supported	Register not supported
PIT_0_TFLG0–TFLG6	—
PIT_1_TFLG0–TFLG5	PIT_1_TFLG6

Diagram



Register reset values

Register	Reset value
TFLG0–TFLG5	PIT_0,PIT_1: 0000_0000h
TFLG6	0000_0000h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4626e-4821-4a8-b251-390d3ab624c1

Fields

Field	Function
31-1 —	Reserved
0 TIF	Timer Interrupt Flag Sets to 1 at the end of the timer period. Writing 1 to this flag clears it and writing 0 has no effect. If enabled, or, when TCTRLn[TIE] = 1, TIF causes an interrupt request. 0b - Timeout has not yet occurred. 1b - Timeout has occurred.

42.5 Functional description

This section provides the functional description of the module.

42.5.1 General operation

This section provides detailed information on the internal operation of the module. Each timer can be used to generate trigger pulses and interrupts, and each interrupt is available on a separate interrupt line.

42.5.1.1 Timers

The timers generate triggers at periodic intervals, when enabled. The timers load the start values as specified in their LDVAL registers, count down to 0 and then load the respective start values again. Each time a timer reaches 0, it generates a trigger pulse and sets the interrupt flag.

All interrupts can be enabled or masked by setting TCTRLn[TIE]. A new interrupt can be generated only after the previous one is cleared.

If desired, the current counter value of the timer can be read via the CVAL registers.

The counter period can be restarted by first disabling and then enabling the timer with TCTRLn[TEN]. See the following figure.

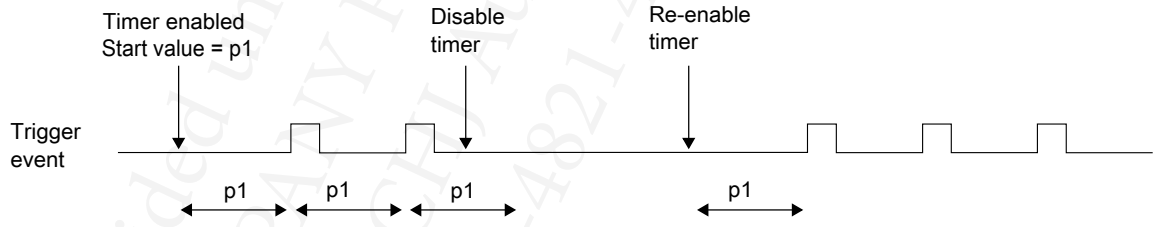


Figure 229. Stopping and starting a timer

The counter period of a running timer can be modified by first disabling the timer, setting a new load value, and then enabling the timer again. See the following figure.

Provided under IHS only
COMPANY PROPRIETARY
Beijing BCR Automotive Technology Co., Ltd.
c8c46285-4821-4b8-b621-390d3ab6240

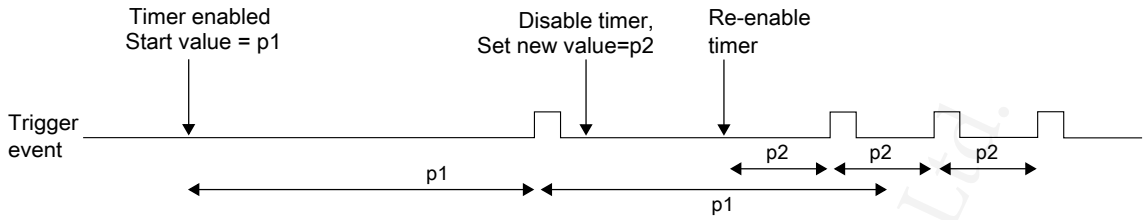


Figure 230. Modifying running timer period

It is also possible to change the counter period without restarting the timer, by writing LDVAL with the new load value. This value then loads after the next trigger event. See the following figure.

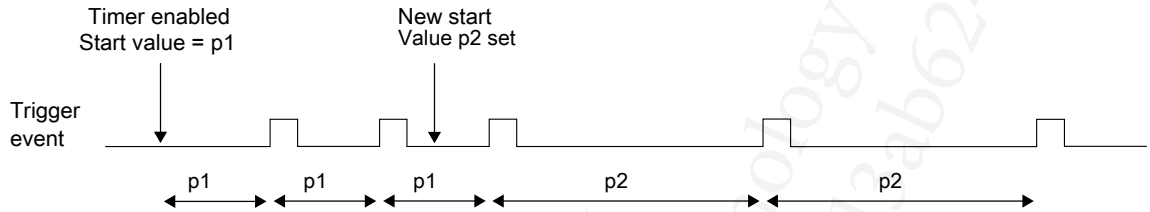


Figure 231. Dynamically setting a new load value

NOTE

- If MCR[FRZ] is written as 1 and when the timer is nearing 0 (CVALn =0x0), the debug mode command may not make it to the IP before the timer expires and generates a trigger.
- Debug Mode will be ignored if CVALn =0x0, but the trigger will remain asserted until the mode is removed. The user is recommended to exit Debug mode and then clear the interrupt TFLGn[TIF].
- If the timers are to be freezed for debug, sufficient time must be ensured for the IP to react.

42.5.1.2 Debug mode

In the Debug mode, the timers are frozen based on MCR[FRZ]. This is intended to aid software development, allowing the developer to halt the processor, investigate the current state of the system, for example, the timer values, and then continue the operation.

42.5.2 Interrupts

All the timers support interrupt generation.

Timer interrupts can be enabled by setting TCTRLn[TIE].

TFLGn[TIF] are set to 1 when a timeout occurs on the associated timer, and are cleared to 0 by writing a 1 to the corresponding TFLGn[TIF].

42.5.3 Chained timers

When a timer has chain mode enabled, it counts after the previous timer has expired. So if timer n-1 counts down to 0, counter n decrements the value by one. This allows to chain some of the timers together to form a longer timer. The first timer (timer 0) cannot be chained to any other timer.

42.6 Initialization and application information

In the example configuration:

- The PIT clock has a frequency of 50 MHz.

- Timer 1 creates an interrupt every 5.12 ms.
- Timer 3 creates a trigger event every 30 ms.

The PIT module must be activated by writing a 0 to MCR[MDIS].

The 50 MHz clock frequency equates to a clock period of 20 ns. Timer 1 needs to trigger every 5.12 ms/20 ns = 256,000 cycles and Timer 3 every 30 ms/20 ns = 1,500,000 cycles. The value for the LDVAL register trigger is calculated as:

LDVAL trigger = (period / clock period) -1

This means LDVAL1 and LDVAL3 must be written with 0x0003E7FF and 0x0016E35F, respectively.

The interrupt for Timer 1 is enabled by setting TCTRL1[TIE]. The timer is started by writing 1 to TCTRL1[TEN].

Timer 3 shall be used only for triggering. Therefore, Timer 3 is started by writing a 1 to TCTRL3[TEN]. Also, TCTRL3[TIE] stays at 0.

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;
```

```
// Timer 1
PIT_LDVAL1 = 0x0003E7FF; // setup timer 1 for 256000 cycles
PIT_TCTRL1 = TIE; // enable Timer 1 interrupts
PIT_TCTRL1 |= TEN; // start Timer 1
```

```
// Timer 3
PIT_LDVAL3 = 0x0016E35F; // setup timer 3 for 1500000 cycles
PIT_TCTRL3 |= TEN; // start Timer 3
```

42.7 Example configuration for chained timers

In the example configuration:

- The PIT clock has a frequency of 100 MHz.
- Timers 1 and 2 are available.
- An interrupt is raised every 1 minute.

The PIT module needs to be activated by writing a 0 to MCR[MDIS].

The 100 MHz clock frequency equates to a clock period of 10 ns, so the PIT needs to count for 6000 million cycles, which is more than a single timer can do. So, Timer 1 is set up to trigger every 6 s (600 million cycles). Timer 2 is chained to Timer 1 and programmed to trigger 10 times.

The value for the LDVAL register trigger is calculated as number of cycles-1, so LDVAL1 receives the value 0x23C345FF and LDVAL2 receives the value 0x00000009.

The interrupt for Timer 2 is enabled by setting TCTRL2[TIE], the Chain mode is activated by setting TCTRL2[CHN], and the timer is started by writing a 1 to TCTRL2[TEN].

TCTRL1[TEN] needs to be set, and TCTRL1[CHN] and TCTRL1[TIE] are cleared.

Freescale NXP Confidential Proprietary
COMPANY PROPRIETARY
Beijing C.T. Automotive Technology Co., Ltd.
c8c62e3-4421-48a8-b251-390d3ab624

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;
```

```
// Timer 2
PIT_LDVAL2 = 0x00000009; // setup Timer 2 for 10 counts
PIT_TCTRL2 = TIE; // enable Timer 2 interrupt
PIT_TCTRL2 |= CHN; // chain Timer 2 to Timer 1
PIT_TCTRL2 |= TEN; // start Timer 2
```

```
// Timer 1
PIT_LDVAL1 = 0x23C345FF; // setup Timer 1 for 600 000 000 cycles
PIT_TCTRL1 = TEN; // start Timer 1
```

42.8 Example configuration for the Lifetime Timer

To configure the Lifetime Timer, channels 0 and 1 need to be chained together.

First, the PIT module needs to be activated by writing a 0 to the MDIS bit in the CTRL register, and then the LDVAL registers need to be set to the maximum value.

The following example code matches the described setup:

```
// turn on PIT
PIT_MCR = 0x00;
```

```
// Timer 1
PIT_LDVAL1 = 0xFFFFFFFF; // setup timer 1 for maximum counting period
PIT_TCTRL1 = 0x0; // disable timer 1 interrupts
PIT_TCTRL1 |= CHN; // chain timer 1 to timer 0
PIT_TCTRL1 |= TEN; // start timer 1
```

```
// Timer 0
PIT_LDVAL0 = 0xFFFFFFFF; // setup timer 0 for maximum counting period
PIT_TCTRL0 = TEN; // start timer 0
```

To access the Lifetime Timer, read first LTMR64H and then LTMR64L.

```
current_uptime = PIT_LTMR64H<<32;
current_uptime = current_uptime + PIT_LTMR64L;
```

Preliminary Information COMPANY PROPRIETARY Beijing CHY Automotive Technology Co., Ltd. c8c4625-a821-48a2-b251-390d3ab624c1

Chapter 43
FlexTimer (FTM)

43.1 Chip-specific FTM information

43.1.1 FTM instances

The chip contains two FTM instances: FTM_0 and FTM_1.

The following table summarizes the features for each instance.

Table 264. FTM instance features

Feature	FTM_0	FTM_1
Number of channels	6	6
Input filter on channel 0	Yes	Yes
Input filter on channel 1	Yes	Yes
Input filter on channel 2	Yes	Yes
Input filter on channel 3	Yes	Yes
Number of fault inputs	0	0
Quadrature decoder	Yes	Yes
DMA support	Yes	Yes
Output Buffer Enable active value	1	1
Half-cycle reload	Yes	Yes
Dithering	Yes	Yes

43.1.2 FTM global time base

This chip provides the optional FTM global time base feature. For more details, see the Global time base (GTB) section of this chapter. FTM0 provides the only source for the FTM global time base. The FTM1 module can share the same time base as shown in the following figure.

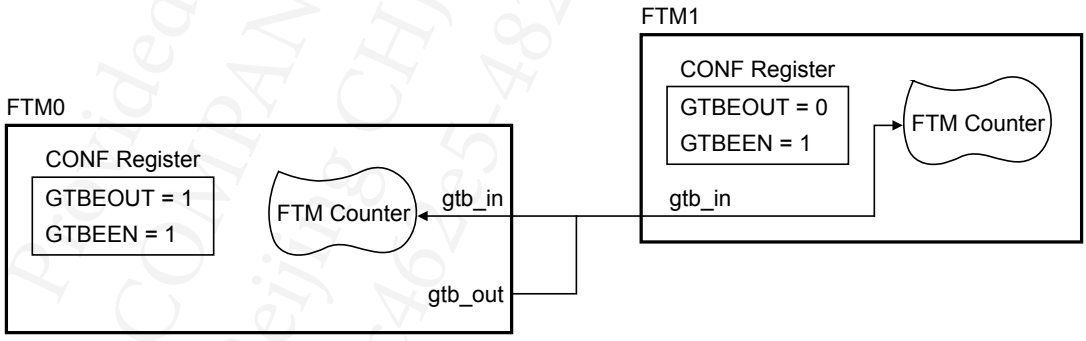


Figure 232. FTM global time base configuration

43.1.3 Initialization triggers and external triggers

The following table shows FlexTimer trigger connections.

Table 265. FTM trigger connections

Timer	Trigger or channel	Connection
FTM_0	Trigger_0 input	GMAC_0
FTM_0	Trigger_1 input	Reserved
FTM_0	Trigger_2 input	Reserved
FTM_1	Trigger_0 input	GMAC_0
FTM_1	Trigger_1 input	Reserved
FTM_1	Trigger_2 input	Reserved
FTM_0	Channel_0 output	CTU trigger 0
FTM_0	Channel_1 output	CTU trigger 1
FTM_0	Channel_2 output	CTU trigger 2
FTM_0	Channel_5 output	GMAC_0
FTM_1	Channel_0 output	CTU trigger 3
FTM_1	Channel_1 output	CTU trigger 4
FTM_1	Channel_2 output	CTU trigger 5
FTM_1	Channel_5 output	GMAC_0

43.2 Introduction

The FlexTimer module (FTM) is a two-to-eight channel timer that supports input capture, output compare, and the generation of PWM signals to control electric motor and power management applications. The FTM time reference is a 16-bit counter that can be used as an unsigned or signed counter.

43.2.1 Features

The FTM features include:

- FTM source clock is selectable
  - Source clock can be the FTM input clock, the fixed frequency clock, or an external clock
  - Fixed frequency clock is an additional clock input to allow the selection of an on chip clock source other than the FTM input clock
  - Selecting external clock connects FTM clock to a chip level input pin therefore allowing to synchronize the FTM counter with an off chip clock source
- Prescaler divide-by 1, 2, 4, 8, 16, 32, 64, or 128
- 16-bit counter
  - It can be a free-running counter or a counter with initial and final value
  - The counting can be up or up-down
- Each channel can be configured for input capture, output compare, or edge-aligned PWM mode
- In Input Capture mode:

  - The capture can occur on rising edges, falling edges or both edges
  - An input filter can be selected for some channels. One unique prescaler is available for all filters
- In Output Compare mode the output signal can be set, cleared, or toggled on match
- All channels can be configured for center-aligned PWM mode
- Each pair of channels can be combined to generate a PWM signal with independent control of both edges of PWM signal
- The FTM channels can operate as pairs with equal outputs, pairs with complementary outputs, or independent channels with independent outputs
- The deadtime insertion is available for each complementary pair
- Generation of match triggers
- Software control of PWM outputs
- The polarity of each channel is configurable
- The generation of an interrupt per channel
- The generation of an interrupt when the counter overflows
- The generation of an interrupt when a register reload point occurs
- Synchronized loading of write buffered FTM registers
- Half cycle and Full cycle register reload capacity
- Write protection for critical registers
- Backwards compatible with TPM
- Testing of input capture mode
- Direct access to input pin states
- Dual edge capture for pulse and period width measurement
- Quadrature decoder with input filters, relative position counting, and interrupt on position count or capture of position count on external event
- The FTM channels can be selected to generate a trigger pulse on channel output instead of a PWM
- Dithering capability to simulate fine edge control for both PWM period or PWM duty cycle

43.2.2 Modes of operation

When the chip is in an active Debug mode, the FTM temporarily suspends all counting until the chip returns to normal user operating mode. During Stop mode, all FTM input clocks are stopped, so the FTM is effectively disabled until clocks resume. During Wait mode, the FTM continues to operate normally. If the FTM does not need to produce a real time reference or provide the interrupt sources needed to wake the chip from Wait mode, the power can then be saved by disabling FTM functions before entering Wait mode.

43.2.3 Block Diagram

The FTM uses one input/output (I/O) pin per channel, CHn (FTM channel (n)) where n is the channel number (0–7).

NOTE

The number of channels supported can vary for each instance of the FTM module on a chip. See the chip-specific FTM information to see how many channels are supported for each module instance. For example, if a module instance supports only six channels, references to channel numbers 6 and 7 do not apply for that instance.

The following figure shows the FTM structure. The central component of the FTM is the 16-bit counter with programmable initial and final values and its counting can be up or up-down.

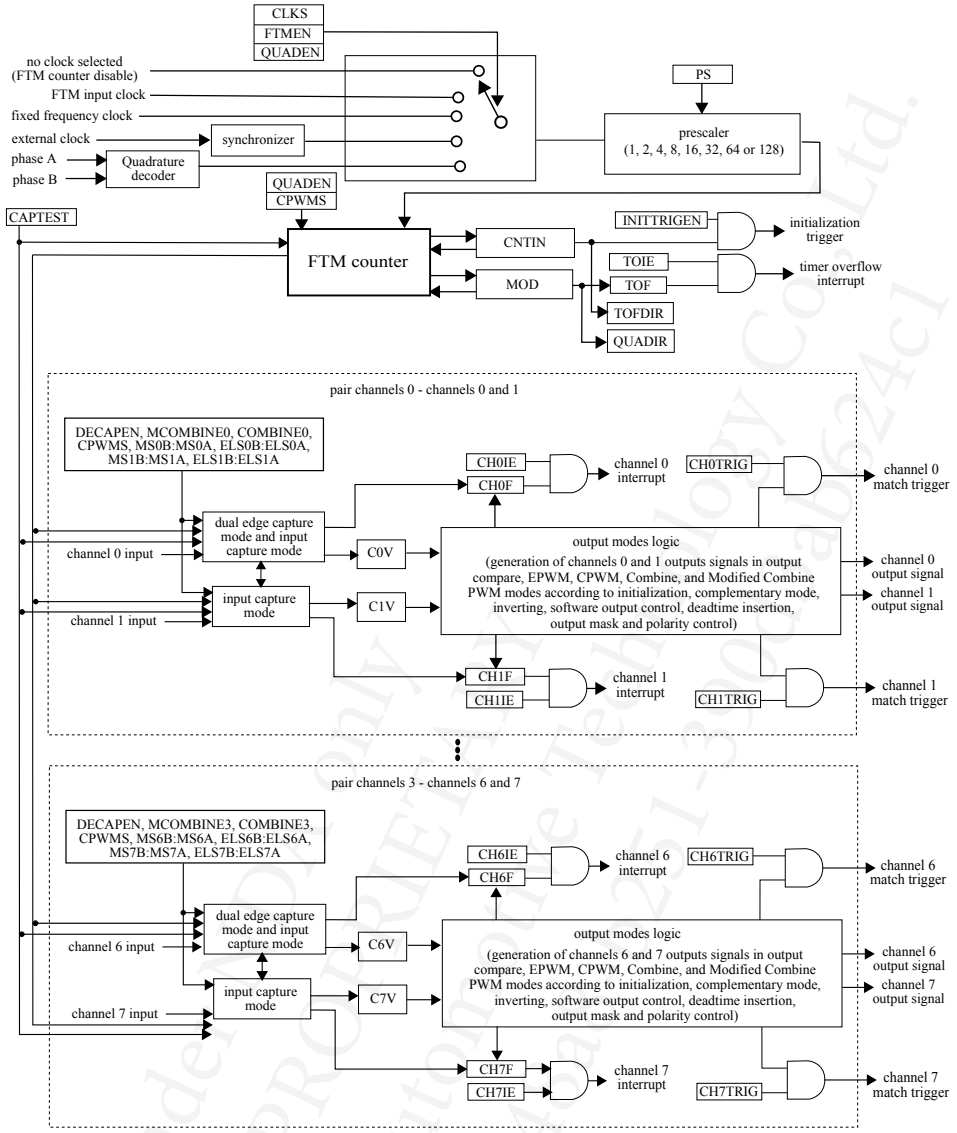


Figure 233. FTM Block Diagram

43.3 FTM signal descriptions

Table 266 shows the user-accessible signals for the FTM.

Table 266. FTM signal descriptions

Signal	Description	I/O	Function
EXTCLK	External clock. FTM external clock can be selected to drive the FTM counter.	I	The external clock input signal is used as the FTM counter clock if selected by CLKS[1:0] bits in the SC register. This clock signal must not exceed 1/4 of FTM input clock frequency. The FTM counter prescaler selection and settings are also used when an external clock is selected.

Table continues on the next page...

Table 266. FTM signal descriptions (continued)

Signal	Description	I/O	Function
CHn	FTM channel (n), where n can be 7-0	I/O	Each FTM channel can be configured to operate either as input or output. The direction associated with each channel, input or output, is selected according to the mode assigned for that channel.
PHA	Quadrature decoder phase A input. Input pin associated with quadrature decoder phase A.	I	The quadrature decoder phase A input is used as the Quadrature Decoder mode is selected. The phase A input signal is one of the signals that control the FTM counter increment or decrement in the Quadrature Decoder Mode.
PHB	Quadrature decoder phase B input. Input pin associated with quadrature decoder phase B.	I	The quadrature decoder phase B input is used as the Quadrature Decoder mode is selected. The phase B input signal is one of the signals that control the FTM counter increment or decrement in the Quadrature Decoder Mode.

43.4 Memory map and register definition

43.4.1 Memory map

This section presents a high-level summary of the FTM registers and how they are mapped.

The registers and bits of an unavailable function in the FTM remain in the memory map and in the reset value, but they have no active function.

NOTE

The number of channels supported can vary for each instance of the FTM module on a chip. See the chip-specific FTM information to see how many channels are supported for each module instance.

43.4.2 Register descriptions

Accesses to reserved addresses result in transfer errors. Registers for absent channels are considered reserved. Double buffered register writes must be done using 32-bit operations.

43.4.3 FTM register descriptions

43.4.3.1 FTM memory map

FTM_0 base address: 401F_4000h

FTM_1 base address: 402E_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Status And Control (SC)	32	RW	0000_0000h
4h	Counter (CNT)	32	RW	0000_0000h
8h	Modulo (MOD)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
Ch	Channel (n) Status And Control (C0SC)	32	RW	0000_0000h
10h	Channel (n) Value (C0V)	32	RW	0000_0000h
14h	Channel (n) Status And Control (C1SC)	32	RW	0000_0000h
18h	Channel (n) Value (C1V)	32	RW	0000_0000h
1Ch	Channel (n) Status And Control (C2SC)	32	RW	0000_0000h
20h	Channel (n) Value (C2V)	32	RW	0000_0000h
24h	Channel (n) Status And Control (C3SC)	32	RW	0000_0000h
28h	Channel (n) Value (C3V)	32	RW	0000_0000h
2Ch	Channel (n) Status And Control (C4SC)	32	RW	0000_0000h
30h	Channel (n) Value (C4V)	32	RW	0000_0000h
34h	Channel (n) Status And Control (C5SC)	32	RW	0000_0000h
38h	Channel (n) Value (C5V)	32	RW	0000_0000h
4Ch	Counter Initial Value (CNTIN)	32	RW	0000_0000h
50h	Capture And Compare Status (STATUS)	32	RW	0000_0000h
54h	Features Mode Selection (MODE)	32	RW	0000_0004h
58h	Synchronization (SYNC)	32	RW	0000_0000h
5Ch	Initial State For Channels Output (OUTINIT)	32	RW	0000_0000h
60h	Output Mask (OUTMASK)	32	RW	0000_0000h
64h	Function For Linked Channels (COMBINE)	32	RW	0000_0000h
68h	Deadtime Configuration (DEADTIME)	32	RW	0000_0000h
6Ch	FTM External Trigger (EXTTRIG)	32	RW	0000_0000h
70h	Channels Polarity (POL)	32	RW	0000_0000h
74h	Fault Mode Status (FMS)	32	RW	0000_0000h
78h	Input Capture Filter Control (FILTER)	32	RW	0000_0000h
80h	Quadrature Decoder Control And Status (QDCTRL)	32	RW	0000_0000h
84h	Configuration (CONF)	32	RW	0000_0000h
8Ch	Synchronization Configuration (SYNCONF)	32	RW	0000_0000h
90h	FTM Inverting Control (INVCTRL)	32	RW	0000_0000h
94h	FTM Software Output Control (SWOCTRL)	32	RW	0000_0000h
98h	FTM PWM Load (PWMLOAD)	32	RW	0000_0000h
9Ch	Half Cycle Register (HCR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
200h	Mirror of Modulo Value (MOD_MIRROR)	32	RW	0000_0000h
204h - 218h	Mirror of Channel (n) Match Value (C0V_MIRROR - C5V_MIRROR)	32	RW	0000_0000h

43.4.3.2 Status And Control (SC)

Offset

Register	Offset
SC	0h

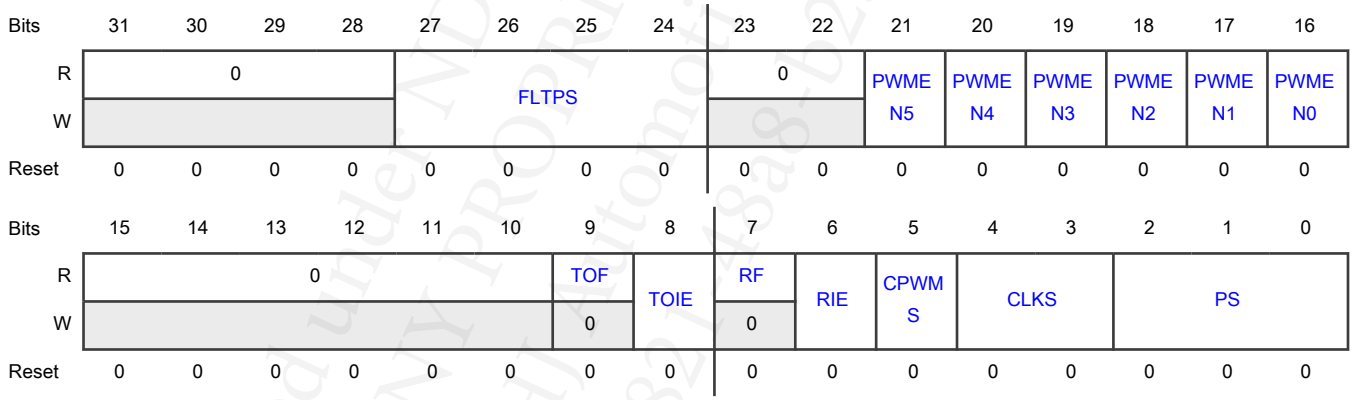
Function

SC contains the overflow status flag and control bits used to configure the interrupt enable, FTM configuration, clock source, filter prescaler, and prescaler factor.

This register also contains the output enable control bits and the reload opportunity flag control.

These controls relate to all channels within this module.

Diagram



Fields

Field	Function
31-28 —	Reserved
27-24 FLTPS	Filter Prescaler The bits FLTPS selects the clock prescaler used in the FTM filters:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• channel input filters • quadrature decoder phase A and B inputs filters Writing to the bits FLTPS has immediate effect. 0000b - Divide by 1 0001b - Divide by 2 0010b - Divide by 3 0011b - Divide by 4 0100b - Divide by 5 0101b - Divide by 6 0110b - Divide by 7 0111b - Divide by 8 1000b - Divide by 9 1001b - Divide by 10 1010b - Divide by 11 1011b - Divide by 12 1100b - Divide by 13 1101b - Divide by 14 1110b - Divide by 15 1111b - Divide by 16
23-22 —	Reserved
21-16 PWMENn	Channel n PWM enable bit This bit enables the PWM channel output. This bit should be set to 0 (output disabled) when an input mode is used. 0b - Channel output port is disabled. 1b - Channel output port is enabled.
15-10 —	Reserved
9 TOF	Timer Overflow Flag Set by hardware when the FTM counter passes the value in the MOD register. The TOF bit is cleared by reading the SC register while TOF is set and then writing a 0 to TOF bit. Writing a 1 to TOF has no effect.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing C-I Automotive Technology Co., Ltd.
cac62e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
	If another FTM overflow occurs between the read and write operations, the write operation has no effect; therefore, TOF remains set indicating an overflow has occurred. In this case, a TOF interrupt request is not lost due to the clearing sequence for a previous TOF. 0b - FTM counter has not overflowed. 1b - FTM counter has overflowed.
8 TOIE	Timer Overflow Interrupt Enable Enables FTM overflow interrupts. 0b - Disable TOF interrupts. Use software polling. 1b - Enable TOF interrupts. An interrupt is generated when TOF equals one.
7 RF	Reload Flag The RF bit is set at each selected reload point. See Reload Points. The RF bit is cleared by reading the SC register while RF is set and then writing a 0 to RF bit. Writing 1 to RF has no effect. If another selected reload point happens between the read and write operations, the write operation has no effect; therefore, RF remains set. 0b - A selected reload point did not happen. 1b - A selected reload point happened.
6 RIE	Reload Point Interrupt Enable Enables the reload point interrupt. 0b - Reload point interrupt is disabled. 1b - Reload point interrupt is enabled.
5 CPWMS	Center-Aligned PWM Select Selects CPWM mode. This mode configures the FTM to operate in Up-Down Counting mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - FTM counter operates in Up Counting mode. 1b - FTM counter operates in Up-Down Counting mode.
4-3 CLKS	Clock Source Selection Selects one of the three FTM counter clock sources. This field is write protected. It can be written only when MODE[WPDIS] = 1. 00b - No clock selected. This in effect disables the FTM counter. 01b - FTM input clock 10b - Fixed frequency clock 11b - External clock
2-0	Prescale Factor Selection

Table continues on the next page...

Table continued from the previous page...

Field	Function
PS	Selects one of 8 division factors for the clock source selected by CLKS. The new prescaler factor affects the clock source on the next FTM input clock cycle after the new value is updated into the register bits. This field is write protected. It can be written only when MODE[WPDIS] = 1. 000b - Divide by 1 001b - Divide by 2 010b - Divide by 4 011b - Divide by 8 100b - Divide by 16 101b - Divide by 32 110b - Divide by 64 111b - Divide by 128

43.4.3.3 Counter (CNT)

Offset

Register	Offset
CNT	4h

Function

The CNT register contains the FTM counter value.

Reset clears the CNT register. Writing any value to COUNT updates the counter with its initial value, CNTIN.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4818-b251-390d3ab624c1

Fields

Field	Function
31-16 —	Reserved
15-0 COUNT	Counter Value

43.4.3.4 Modulo (MOD)

Offset

Register	Offset
MOD	8h

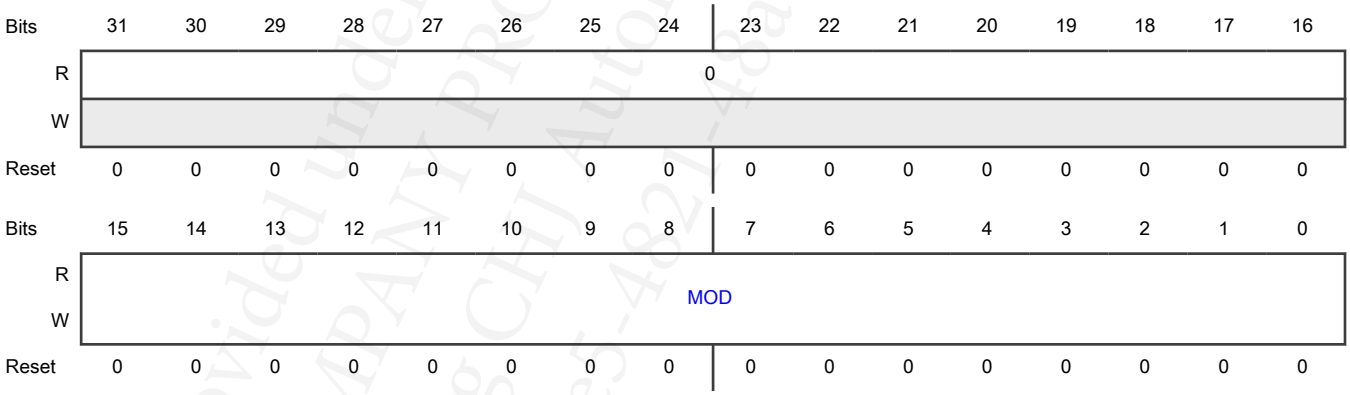
Function

The Modulo register contains the modulo value for the FTM counter. After the FTM counter reaches the modulo value, the overflow flag (TOF) becomes set at the next clock cycle, and the next value of FTM counter depends on the selected counting method; see Counter.

Writes to the MOD register are done on its write buffer. The MOD register is updated with its write buffer value according to Registers updated from write buffers. If FTMEN = 0, a write to SC register resets manually this write coherency mechanism.

Initialize the FTM counter, by writing to CNT, before writing to the MOD register to avoid confusion about when the first counter overflow will occur.

Diagram



Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-4a8-b21-90d3ab624c1

Table continued from the previous page...

Field	Function
—	
15-0 MOD	MOD Modulo Value

43.4.3.5 Channel (n) Status And Control (C0SC - C5SC)

Offset

Register	Offset
C0SC	Ch
C1SC	14h
C2SC	1Ch
C3SC	24h
C4SC	2Ch
C5SC	34h

Function

CnSC contains channel (n) status bits and control bits that select the channel (n) mode and its functionality.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					CHOV	CHIS	TRIGMODE	CHF	CHIE	MSB	MSA	ELSB	ELSA	ICRST	DMA
W									0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-11 —	Reserved

Table continues on the next page...

provided under NDA only
COMPANY PROPRIETARY
Beijing CEC Automotive Technology Co., Ltd.
c8c462e5-9021-4848-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
10 CHOV	Channel (n) Output Value The CHOV bit has the final value of the channel (n) output. **NOTE** The CHOV bit should be ignored when the channel (n) is not in an output mode. 0b - The channel (n) output is zero. 1b - The channel (n) output is one.
9 CHIS	Channel (n) Input State The CHIS bit has the value of the channel (n) input after the double-sampling or the filtering (if the channel (n) filter is enabled) both them are inside the FTM. **NOTE** The CHIS bit should be ignored when the channel (n) is not in an input mode. **NOTE** When the pair channels is on dual edge mode, the channel (n+1) CHIS bit is the channel (n+1) input value and not the channel (n) input value (this signal is the input signal used by the dual edge mode). 0b - The channel (n) input is zero. 1b - The channel (n) input is one.
8 TRIGMODE	Trigger mode control This bit controls the trigger generation on FTM channel outputs. This mode is allowed only if when FTM channel is configured to EPWM or CPWM modes. If a match in the channel occurs, a trigger pulse with one FTM clock cycle width will be generated in the channel output. See Channel trigger output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - Channel outputs will generate the normal PWM outputs without generating a pulse. 1b - If a match in the channel occurs, a trigger generation on channel output will happen. The trigger pulse width has one FTM clock cycle.
7 CHF	Channel (n) Flag Set by hardware when an event occurs on the channel (n). CHF is cleared by reading the CnSC register while CHF is set and then writing a 0 to the CHF bit. Writing a 1 to CHF has no effect. If another event occurs between the read and write operations, the write operation has no effect; therefore, CHF remains set indicating an event has occurred. In this case a CHF interrupt request is not lost due to the clearing sequence for a previous CHF. 0b - No channel (n) event has occurred. 1b - A channel (n) event has occurred.
6	Channel (n) Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
CHIE	Enables channel (n) interrupt. 0b - Disable channel (n) interrupt. Use software polling. 1b - Enable channel (n) interrupt.
5 MSB	Channel (n) Mode Select Used on the selection of the channel (n) mode. See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
4 MSA	Channel (n) Mode Select Used on the selection of the channel (n) mode. See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
3 ELSB	Channel (n) Edge or Level Select Used on the selection of the channel (n) mode. See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
2 ELSA	Channel (n) Edge or Level Select Used on the selection of the channel (n) mode. See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
1 ICRST	FTM counter reset by the selected input capture event. FTM counter reset is driven by the selected event of the channel (n) in the Input Capture mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - FTM counter is not reset when the selected channel (n) input event is detected. 1b - FTM counter is reset when the selected channel (n) input event is detected.
0 DMA	DMA Enable Enables DMA transfers for the channel. 0b - Disable DMA transfers. 1b - Enable DMA transfers.

43.4.3.6 Channel (n) Value (C0V - C5V)

Offset

Register	Offset
C0V	10h
C1V	18h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
C2V	20h
C3V	28h
C4V	30h
C5V	38h

Function

These registers contain the captured FTM counter value for the input modes or the match value for the output modes.

In Input Capture , Capture Test, and Dual Edge Capture modes, any write to a CnV register is ignored.

In output modes, writes to the CnV register are done on its write buffer. The CnV register is updated with its write buffer value according to Registers updated from write buffers. If FTMEN = 0, a write to CnSC register resets manually this write coherency mechanism.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VAL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 VAL	Channel Value Captured FTM counter value of the input modes or the match value for the output modes

43.4.3.7 Counter Initial Value (CNTIN)

Offset

Register	Offset
CNTIN	4Ch

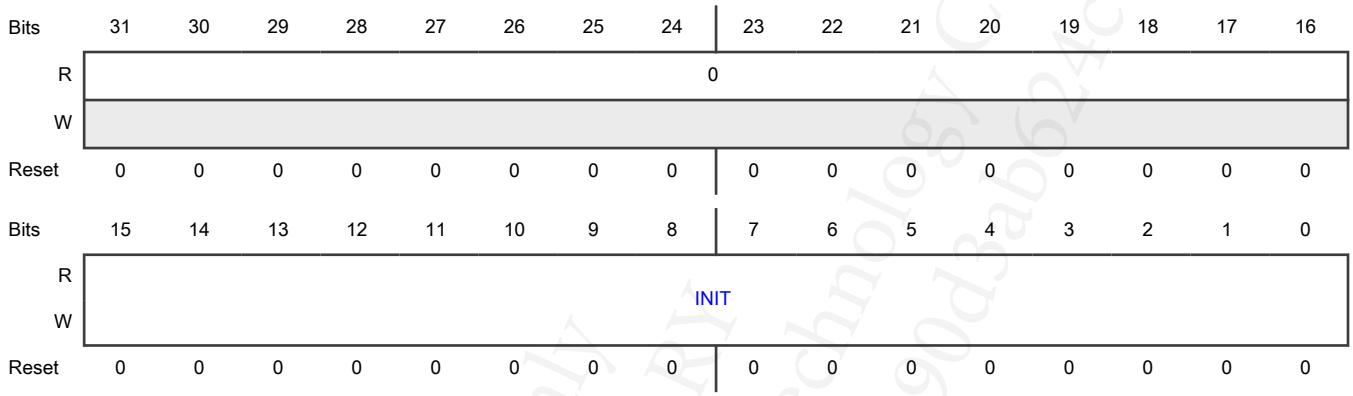
Function

The Counter Initial Value register contains the initial value for the FTM counter.

Writing to the CNTIN register latches the value into a buffer. The CNTIN register is updated with the value of its write buffer according to Registers updated from write buffers.

When the FTM clock is initially selected, by writing a non-zero value to the CLKS bits, the FTM counter starts with the value 0x0000. To avoid this behavior, before the first write to select the FTM clock, write the new value to the CNTIN register and then initialize the FTM counter by writing any value to the CNT register.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 INIT	INIT Initial Value Of The FTM Counter

43.4.3.8 Capture And Compare Status (STATUS)

Offset

Register	Offset
STATUS	50h

Function

The STATUS register contains a copy of the status flag CHF bit in CnSC for each FTM channel for software convenience.

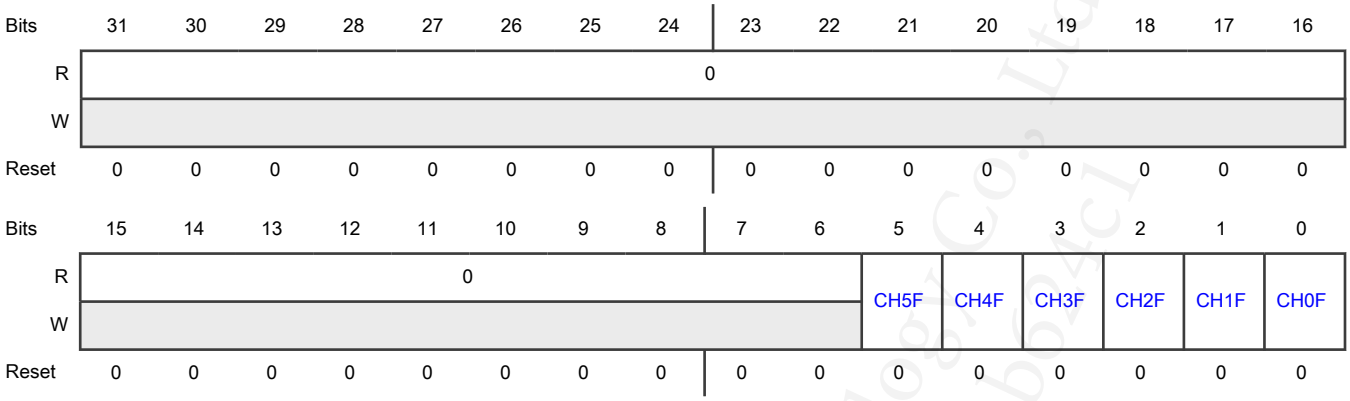
Each CHF bit in STATUS is a mirror of CHF bit in CnSC. All CHF bits can be checked using only one read of STATUS. All CHF bits can be cleared by reading STATUS followed by writing 0x00 to STATUS.

Hardware sets the individual channel flags when an event occurs on the channel. CHF is cleared by reading STATUS while CHF is set and then writing a 0 to the CHF bit. Writing a 1 to CHF has no effect.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ec02e5-4821-48a8-b251-390d8ab624c1

If another event occurs between the read and write operations, the write operation has no effect; therefore, CHF remains set indicating an event has occurred. In this case, a CHF interrupt request is not lost due to the clearing sequence for a previous CHF.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 CHnF	Channel n Flag See the register description. 0b - No channel event has occurred. 1b - A channel event has occurred.

43.4.3.9 Features Mode Selection (MODE)

Offset

Register	Offset
MODE	54h

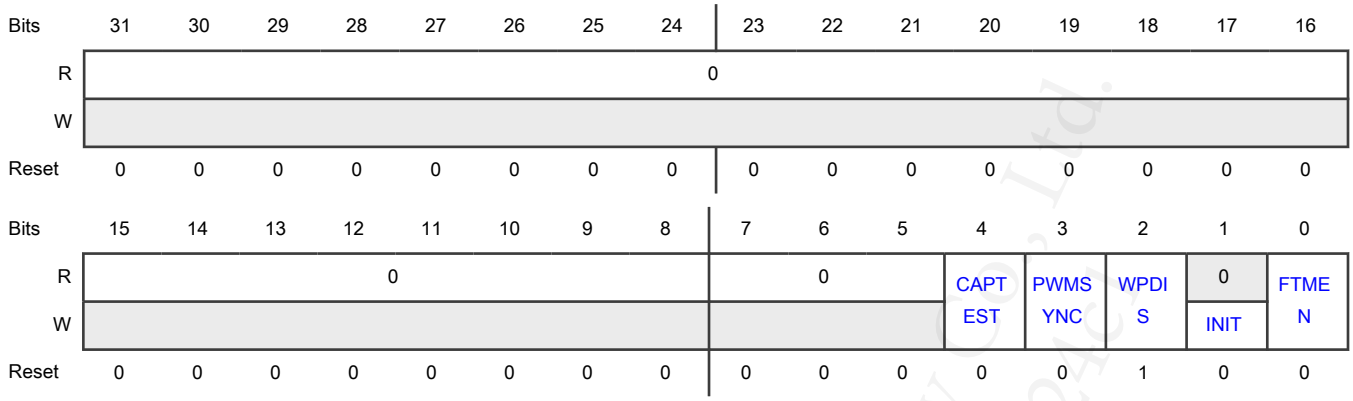
Function

This register contains the global enable bit for FTM-specific features and the control bits used to configure:

- Capture Test mode
- PWM synchronization
- Write protection
- Channel output initialization

These controls relate to all channels within this module.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-5 —	Reserved
4 CAPTEST	Capture Test Mode Enable Enables the capture test mode. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - Capture test mode is disabled. 1b - Capture test mode is enabled.
3 PWMSYNC	PWM Synchronization Mode Selects which triggers can be used by MOD, CnV, OUTMASK, and FTM counter synchronization. See PWM synchronization. The PWMSYNC bit configures the synchronization when SYNCMODE is 0. 0b - No restrictions. Software and hardware triggers can be used by MOD, CnV, OUTMASK, and FTM counter synchronization. 1b - Software trigger can only be used by MOD and CnV synchronization, and hardware triggers can only be used by OUTMASK and FTM counter synchronization.
2 WPDIS	Write Protection Disable When write protection is enabled (WPDIS = 0), write protected bits cannot be written. When write protection is disabled (WPDIS = 1), write protected bits can be written. The WPDIS bit is the negation of the WPEN bit. WPDIS is cleared when 1 is written to WPEN. WPDIS is set when WPEN bit is read as a 1 and then 1 is written to WPDIS. Writing 0 to WPDIS has no effect. 0b - Write protection is enabled. 1b - Write protection is disabled.
1	Initialize The Channels Output

Table continues on the next page...

Table continued from the previous page...

Field	Function
INIT	When a 1 is written to INIT bit the channels output is initialized according to the state of their corresponding bit in the OUTINIT register. Writing a 0 to INIT bit has no effect. The INIT bit is always read as 0.
0 FTMEN	FTM Enable This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - TPM compatibility. Free running counter and synchronization compatible with TPM. 1b - Free running counter and synchronization are different from TPM behavior.

43.4.3.10 Synchronization (SYNC)

Offset

Register	Offset
SYNC	58h

Function

This register configures the PWM synchronization.

A synchronization event can perform the synchronized update of MOD, CnV, and OUTMASK registers with the value of their write buffer and the FTM counter initialization.

NOTE

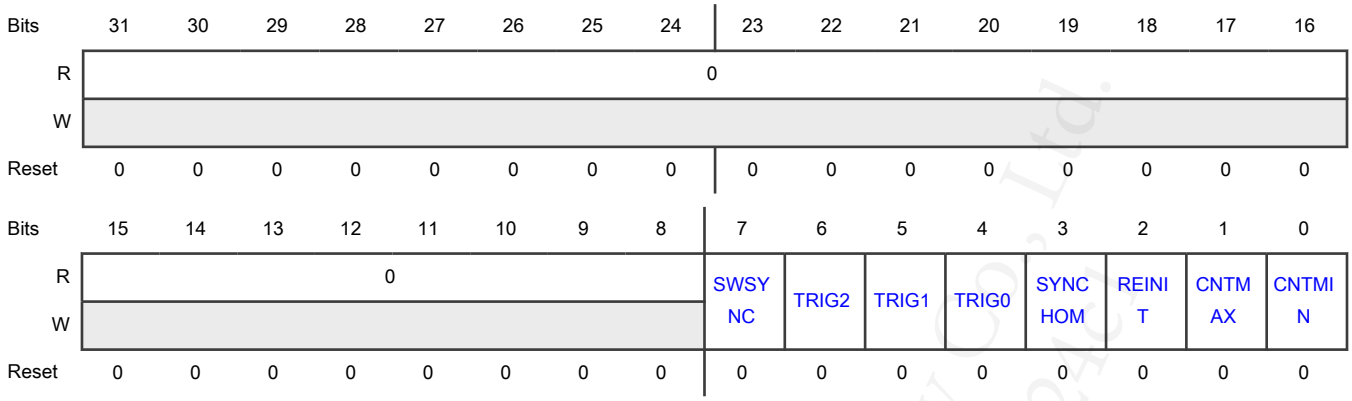
The software trigger, SWSYNC bit, and hardware triggers TRIG0, TRIG1, and TRIG2 bits have a potential conflict if used together when SYNCMODE = 0. Use only hardware or software triggers but not both at the same time, otherwise unpredictable behavior is likely to happen.

The selection of the loading point, CNTMAX and CNTMIN bits, is intended to provide the update of MOD, CNTIN, and CnV registers across all enabled channels simultaneously. The use of the loading point selection together with SYNCMODE = 0 and hardware trigger selection, TRIG0, TRIG1, or TRIG2 bits, is likely to result in unpredictable behavior.

The synchronization event selection also depends on the PWMSYNC (MODE register) and SYNCMODE (SYNCONF register) bits. See PWM synchronization.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJT Automotive Technology Co., Ltd.
c8c462e5-4821-8e-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-8 —	Reserved
7 SWSYNC	PWM Synchronization Software Trigger Selects the software trigger as the PWM synchronization trigger. The software trigger happens when a 1 is written to SWSYNC bit. 0b - Software trigger is not selected. 1b - Software trigger is selected.
6 TRIG2	PWM Synchronization Hardware Trigger 2 Enables hardware trigger 2 to the PWM synchronization. Hardware trigger 2 happens when a rising edge is detected at the trigger 2 input signal. 0b - Trigger is disabled. 1b - Trigger is enabled.
5 TRIG1	PWM Synchronization Hardware Trigger 1 Enables hardware trigger 1 to the PWM synchronization. Hardware trigger 1 happens when a rising edge is detected at the trigger 1 input signal. 0b - Trigger is disabled. 1b - Trigger is enabled.
4 TRIG0	PWM Synchronization Hardware Trigger 0 Enables hardware trigger 0 to the PWM synchronization. Hardware trigger 0 occurs when a rising edge is detected at the trigger 0 input signal. 0b - Trigger is disabled. 1b - Trigger is enabled.
3	Output Mask Synchronization

Table continues on the next page...

Table continued from the previous page...

Field	Function
SYNCHOM	Selects when the OUTMASK register is updated with the value of its buffer. 0b - OUTMASK register is updated with the value of its buffer in all rising edges of the FTM input clock. 1b - OUTMASK register is updated with the value of its buffer only by the PWM synchronization.
2 REINIT	FTM Counter Reinitialization by Synchronization Determines if the FTM counter is reinitialized when the selected trigger for the synchronization is detected (FTM counter synchronization). The REINIT bit configures the synchronization when SYNCMODE is zero. 0b - FTM counter continues to count normally. 1b - FTM counter is updated with its initial value when the selected trigger is detected.
1 CNTMAX	Maximum Loading Point Enable Selects the maximum loading point to PWM synchronization (Synchronization Points). If CNTMAX is 1, the selected loading point is when the FTM counter reaches its maximum value (MOD register). 0b - The maximum loading point is disabled. 1b - The maximum loading point is enabled.
0 CNTMIN	Minimum Loading Point Enable Selects the minimum loading point to PWM synchronization (Synchronization Points). If CNTMIN is 1, the selected loading point is when the FTM counter reaches its minimum value (CNTIN register). 0b - The minimum loading point is disabled. 1b - The minimum loading point is enabled.

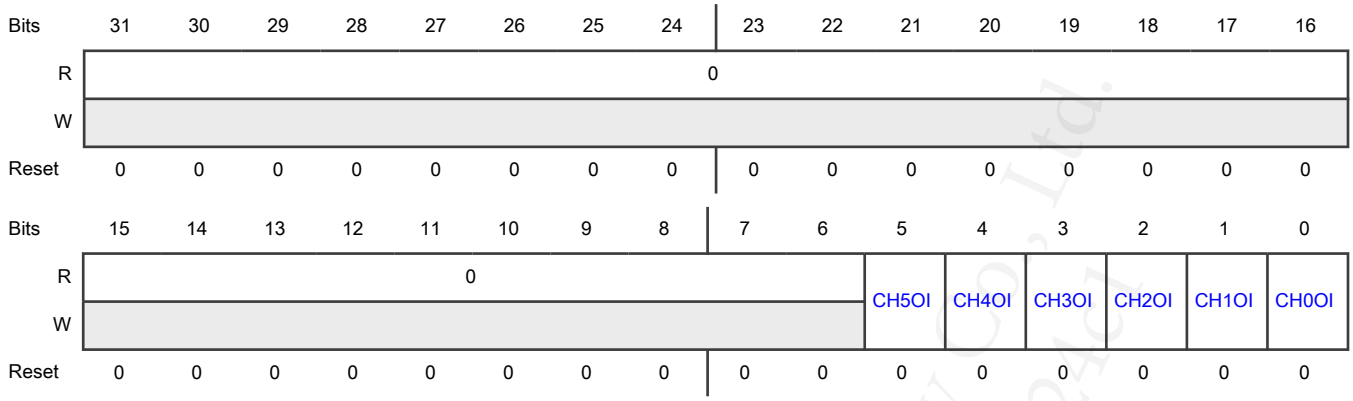
43.4.3.11 Initial State For Channels Output (OUTINIT)

Offset

Register	Offset
OUTINIT	5Ch

Provided by NXP Semiconductors
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
c8c462e5-4821-48a8-b261-3903a3b62fc1

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 CHnOI	Channel n Output Initialization Value Selects the value that is forced into the channel output when the initialization occurs. 0b - The initialization value is 0. 1b - The initialization value is 1.

43.4.3.12 Output Mask (OUTMASK)

Offset

Register	Offset
OUTMASK	60h

Function

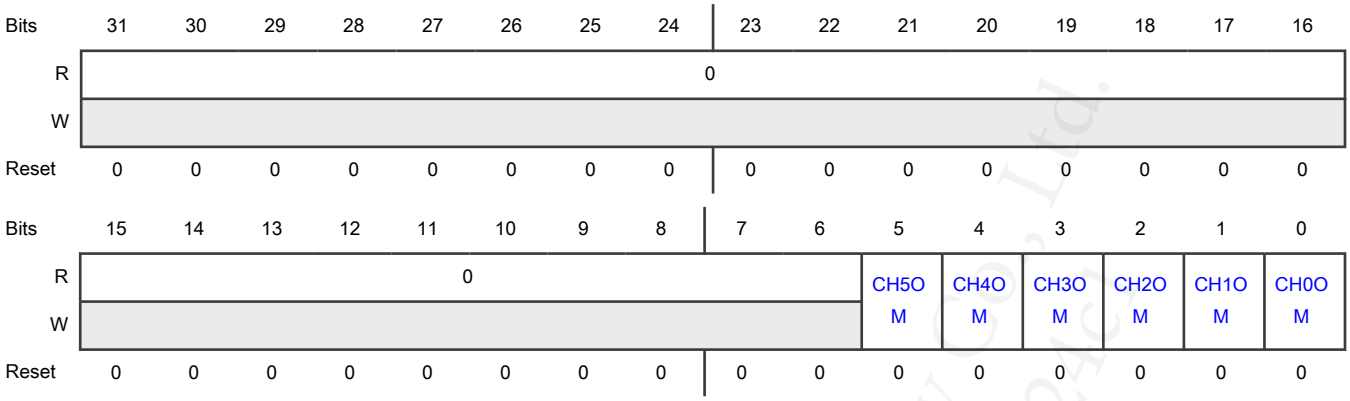
This register provides a mask for each FTM channel. The mask of a channel determines if its output responds, that is, it is masked or not, when a match occurs. This feature is used for BLDC control where the PWM signal is presented to an electric motor at specific times to provide electronic commutation.

Any write to the OUTMASK register, stores the value in its write buffer. The register is updated with the value of its write buffer according to PWM synchronization.

Output Mask bits must not be set for trigger mode.

Printed under NDA only
COMPANY PROPRIETARY
Beijing eSOL Automotive Technology Co., Ltd.
c8c462e5-821-48a8-b251-390d8ab624c

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 CHnOM	Channel n Output Mask Defines if the channel output is masked or unmasked. 0b - Channel output is not masked. It continues to operate normally. 1b - Channel output is masked. It is forced to its inactive state.

43.4.3.13 Function For Linked Channels (COMBINE)

Offset

Register	Offset
COMBINE	64h

Function

This register contains the configuration bits for each pair of channels.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4321-48a8-b25f-390d8ab62402

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								MCOMBIN...	0	SYNCEN2	DTEN2	DECAP2	DECAPEN2	COMP2	COMBINE2
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MCOMBIN...	0	SYNCEN1	DTEN1	DECAP1	DECAPEN1	COMP1	COMBINE1	MCOMBIN...	0	SYNCEN0	DTEN0	DECAP0	DECAPEN0	COMP0	COMBINE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23 MCOMBINE2	Modified Combine Mode For n = 4 Used on the selection of the modified combine mode for channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
22 —	Reserved
21 SYNCEN2	Synchronization Enable For n = 4 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0b - The PWM synchronization in this pair of channels is disabled. 1b - The PWM synchronization in this pair of channels is enabled.
20 DTEN2	Deadtime Enable For n = 4 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - The deadtime insertion in this pair of channels is disabled. 1b - The deadtime insertion in this pair of channels is enabled.
19 DECAP2	Dual Edge Capture Mode Captures For n = 4 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - The dual edge captures are inactive. 1b - The dual edge captures are active.
18 DECAPEN2	Dual Edge Capture Mode Enable For n = 4 Enables the Dual Edge Capture mode in the channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
17 COMP2	Complement Of Channel (n) For n = 4 In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - If the channels (n) and (n+1) are in Combine Mode or Modified Combine PWM Mode, the channel (n+1) output is the same as the channel (n) output. If the channel (n+1) is in Output Compare Mode, EPWM or CPWM, the channel (n+1) output is independent from channel (n) output. 1b - The channel (n+1) output is the complement of the channel (n) output.
16 COMBINE2	Combine Channels For n = 4 Used on the selection of the combine mode for channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
15 MCOMBINE1	Modified Combine Mode For n = 2 Used on the selection of the modified combine mode for channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
14 —	Reserved
13 SYNCEN1	Synchronization Enable For n = 2 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0b - The PWM synchronization in this pair of channels is disabled. 1b - The PWM synchronization in this pair of channels is enabled.
12 DTEN1	Deadtime Enable For n = 2 Enables the deadtime insertion in the channels (n) and (n+1). This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - The deadtime insertion in this pair of channels is disabled. 1b - The deadtime insertion in this pair of channels is enabled.
11 DECAP1	Dual Edge Capture Mode Captures For n = 2

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if Dual Edge Capture – One-Shot mode is selected and when the capture of channel (n+1) event is made. 0b - The dual edge captures are inactive. 1b - The dual edge captures are active.
10 DECAPEN1	Dual Edge Capture Mode Enable For n = 2 Enables the Dual Edge Capture mode in the channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
9 COMP1	Complement Of Channel (n) For n = 2 In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - If the channels (n) and (n+1) are in Combine Mode or Modified Combine PWM Mode, the channel (n+1) output is the same as the channel (n) output. If the channel (n+1) is in Output Compare Mode, EPWM or CPWM, the channel (n+1) output is independent from channel (n) output. 1b - The channel (n+1) output is the complement of the channel (n) output.
8 COMBINE1	Combine Channels For n = 2 Used on the selection of the combine mode for channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
7 MCOMBINE0	Modified Combine Mode For n = 0 Used on the selection of the modified combine mode for channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
6 —	Reserved
5 SYNCEN0	Synchronization Enable For n = 0 Enables PWM synchronization of registers C(n)V and C(n+1)V. 0b - The PWM synchronization in this pair of channels is disabled. 1b - The PWM synchronization in this pair of channels is enabled.
4 DTEN0	Deadtime Enable For n = 0 Enables the deadtime insertion in the channels (n) and (n+1).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - The deadtime insertion in this pair of channels is disabled. 1b - The deadtime insertion in this pair of channels is enabled.
3 DECAP0	Dual Edge Capture Mode Captures For n = 0 Enables the capture of the FTM counter value according to the channel (n) input event and the configuration of the dual edge capture bits. This field applies only when DECAPEN = 1. DECAP bit is cleared automatically by hardware if dual edge capture – one-shot mode is selected and when the capture of channel (n+1) event is made. 0b - The dual edge captures are inactive. 1b - The dual edge captures are active.
2 DECAPEN0	Dual Edge Capture Mode Enable For n = 0 Enables the Dual Edge Capture mode in the channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.
1 COMP0	Complement Of Channel (n) For n = 0 In Complementary mode the channel (n+1) output is the inverse of the channel (n) output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - If the channels (n) and (n+1) are in Combine Mode or Modified Combine PWM Mode, the channel (n+1) output is the same as the channel (n) output. If the channel (n+1) is in Output Compare Mode, EPWM or CPWM, the channel (n+1) output is independent from channel (n) output. 1b - The channel (n+1) output is the complement of the channel (n) output.
0 COMBINE0	Combine Channels For n = 0 Used on the selection of the combine mode for channels (n) and (n+1). See Channel Modes. This field is write protected. It can be written only when MODE[WPDIS] = 1.

43.4.3.14 Deadtime Configuration (DEADTIME)

Offset

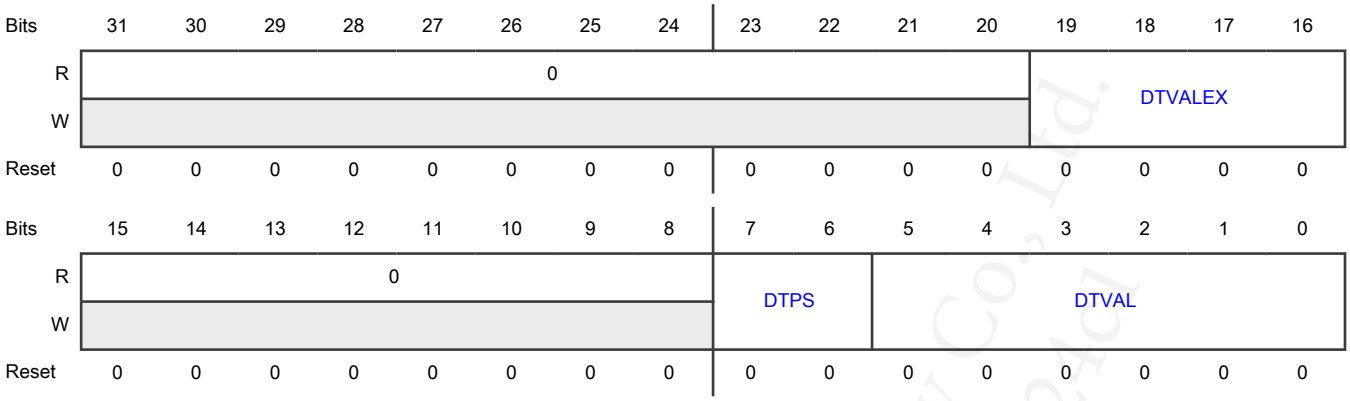
Register	Offset
DEADTIME	68h

Function

This register selects the deadtime prescaler and value for all pair of channels.

Provided under NDA only
COMPANY CONFIDENTIAL
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482f-4aa6-8f61-50d3ab624c1

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 DTVALEX	Extended Deadtime Value This field is a bit extension of the DTVAL field. It defines the 4 most significant bits of the deadtime value. The maximum deadtime value is extended to 1023 using the concatenation {DTVALEX, DTVAL}. Deadtime insert value = (DTPS × {DTVALEX, DTVAL}). This field is write protected. It can be written only when MODE[WPDIS] = 1. **NOTE** If full compatibility is needed with previous software versions, write 0 to DTVALEX bits.
15-8 —	Reserved
7-6 DTPS	Deadtime Prescaler Value Selects the division factor of the FTM input clock. This prescaled clock is used by the deadtime counter. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0xb - Divide the FTM input clock by 1. 10b - Divide the FTM input clock by 4. 11b - Divide the FTM input clock by 16.
5-0 DTVAL	Deadtime Value Selects the deadtime value. This field is write protected. It can be written only when MODE[WPDIS] = 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CAT Automotive Technology Co., Ltd.
c8d162e3-4521-48a8-b251-390d8ab624c1

43.4.3.15 FTM External Trigger (EXTTRIG)

Offset

Register	Offset
EXTTRIG	6Ch

Function

This register:

- Indicates when the external trigger was generated
- Enables the generation of a trigger when the FTM counter is equal to its initial value
- Selects which channels are used in the generation of the external trigger

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						0	0	TRIGF	INITTRI...	CH1TRIG	CH0TRIG	CH5TRIG	CH4TRIG	CH3TRIG	CH2TRIG
W									0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7 TRIGF	Channel Trigger Flag Set by hardware when a channel trigger is generated. Clear TRIGF by reading EXTTRIG while TRIGF is set and then writing a 0 to TRIGF. Writing a 1 to TRIGF has no effect. If another channel trigger is generated before the clearing sequence is completed, the sequence is reset so TRIGF remains set after the clear sequence is completed for the earlier TRIGF.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No channel trigger was generated. 1b - A channel trigger was generated.
6 INITTRIGEN	Initialization Trigger Enable Enables the generation of the trigger when the FTM counter is equal to the CNTIN register. 0b - The generation of initialization trigger is disabled. 1b - The generation of initialization trigger is enabled.
5 CH1TRIG	Channel 1 External Trigger Enable Enables the generation of the external trigger when FTM counter = C1V. 0b - The generation of this external trigger is disabled. 1b - The generation of this external trigger is enabled.
4 CH0TRIG	Channel 0 External Trigger Enable Enables the generation of the external trigger when FTM counter = C0V. 0b - The generation of this external trigger is disabled. 1b - The generation of this external trigger is enabled.
3 CH5TRIG	Channel 5 External Trigger Enable Enables the generation of the external trigger when FTM counter = C5V. 0b - The generation of this external trigger is disabled. 1b - The generation of this external trigger is enabled.
2 CH4TRIG	Channel 4 External Trigger Enable Enables the generation of the external trigger when FTM counter = C4V. 0b - The generation of this external trigger is disabled. 1b - The generation of this external trigger is enabled.
1 CH3TRIG	Channel 3 External Trigger Enable Enables the generation of the external trigger when FTM counter = C3V. 0b - The generation of this external trigger is disabled. 1b - The generation of this external trigger is enabled.
0 CH2TRIG	Channel 2 External Trigger Enable Enables the generation of the external trigger when FTM counter = C2V. 0b - The generation of this external trigger is disabled. 1b - The generation of this external trigger is enabled.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHANGHONG Electronics Technology Co., Ltd.
ca02e5_821_8a8251-390d3ab624c1

43.4.3.16 Channels Polarity (POL)

Offset

Register	Offset
POL	70h

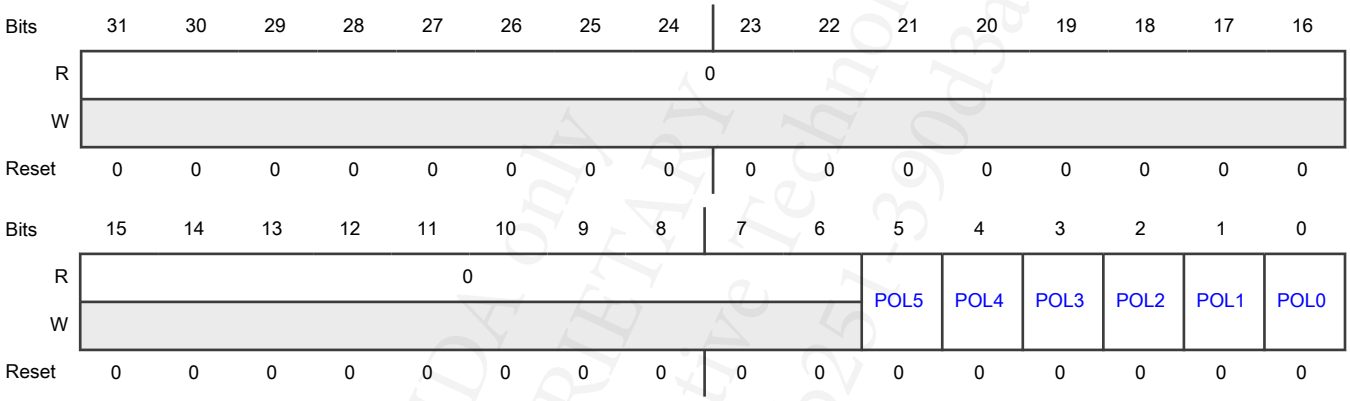
Function

This register defines the output polarity of the FTM channels.

NOTE

The channel safe value is the value of its POL bit.

Diagram



Fields

Field	Function
31-6 —	Reserved
5-0 POLn	Channel n Polarity Defines the polarity of the channel output. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - The channel polarity is active high. 1b - The channel polarity is active low.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46253-821-48a8-b251-390d8ab624c1

43.4.3.17 Fault Mode Status (FMS)

Offset

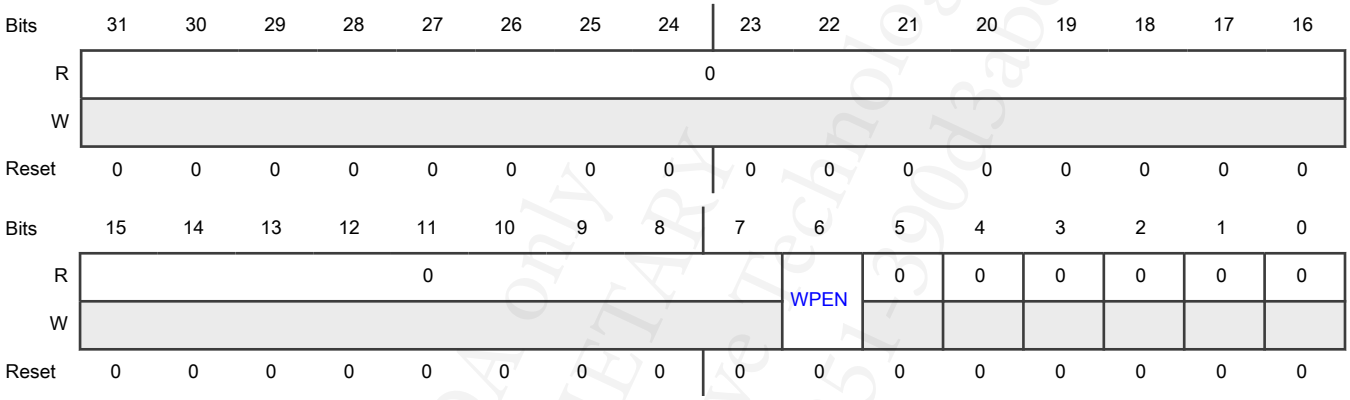
Register	Offset
FMS	74h

Function

This register contains:

- the write protection enable bit

Diagram



Fields

Field	Function
31-7 —	Reserved
6 WPEN	Write Protection Enable The WPEN bit is the negation of the WPDIS bit. WPEN is set when 1 is written to it. WPEN is cleared when WPEN bit is read as a 1 and then 1 is written to WPDIS. Writing 0 to WPEN has no effect. 0b - Write protection is disabled. Write protected bits can be written. 1b - Write protection is enabled. Write protected bits cannot be written.
5 —	Reserved
4 —	Reserved
3	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
2 —	Reserved
1 —	Reserved
0 —	Reserved

43.4.3.18 Input Capture Filter Control (FILTER)

Offset

Register	Offset
FILTER	78h

Function

This register selects the filter value for the inputs of channels.

Channels 4, 5, 6 and 7 do not have an input filter.

NOTE

Writing to the FILTER register has immediate effect and must be done only when the channels 0, 1, 2, and 3 are not in input modes. Failure to do this could result in a missing valid signal.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CH3FVAL				CH2FVAL				CH1FVAL				CH0FVAL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4824-4888-b251-390d3ab624c1

Fields

Field	Function
31-16 —	Reserved
15-12 CH3FVAL	Channel 3 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
11-8 CH2FVAL	Channel 2 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
7-4 CH1FVAL	Channel 1 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.
3-0 CH0FVAL	Channel 0 Input Filter Selects the filter value for the channel input. The filter is disabled when the value is zero.

43.4.3.19 Quadrature Decoder Control And Status (QDCTRL)

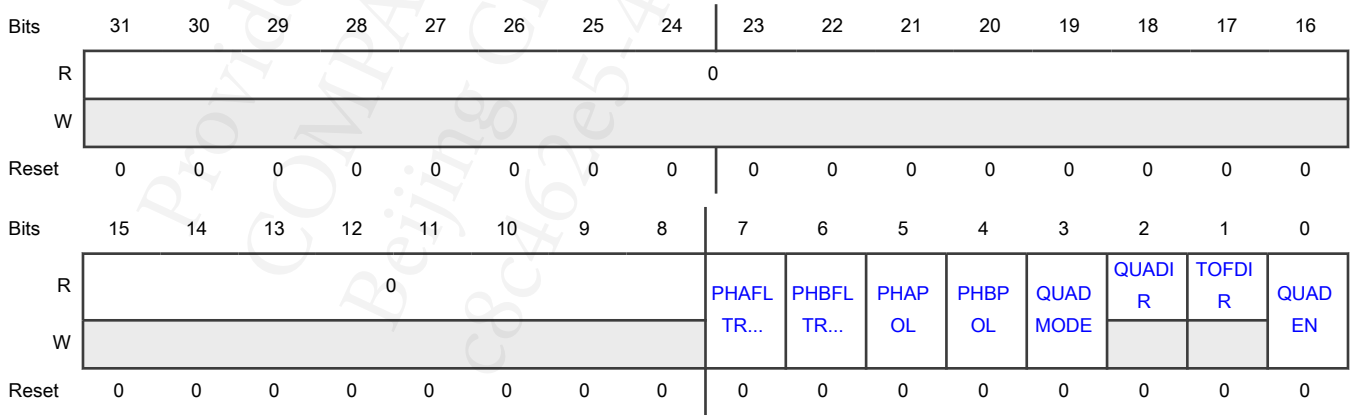
Offset

Register	Offset
QDCTRL	80h

Function

This register has the control and status bits for the Quadrature Decoder mode.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 PHAFLTREN	Phase A Input Filter Enable Enables the filter for the quadrature decoder phase A input. The filter value for the phase A input is defined by the CH0FVAL field of FILTER. The phase A filter is also disabled when CH0FVAL is zero. 0b - Phase A input filter is disabled. 1b - Phase A input filter is enabled.
6 PHBFLTREN	Phase B Input Filter Enable Enables the filter for the quadrature decoder phase B input. The filter value for the phase B input is defined by the CH1FVAL field of FILTER. The phase B filter is also disabled when CH1FVAL is zero. 0b - Phase B input filter is disabled. 1b - Phase B input filter is enabled.
5 PHAPOL	Phase A Input Polarity Selects the polarity for the quadrature decoder phase A input. 0b - Normal polarity. Phase A input signal is not inverted before identifying the rising and falling edges of this signal. 1b - Inverted polarity. Phase A input signal is inverted before identifying the rising and falling edges of this signal.
4 PHBPOL	Phase B Input Polarity Selects the polarity for the quadrature decoder phase B input. 0b - Normal polarity. Phase B input signal is not inverted before identifying the rising and falling edges of this signal. 1b - Inverted polarity. Phase B input signal is inverted before identifying the rising and falling edges of this signal.
3 QUADMODE	Quadrature Decoder Mode Selects the encoding mode used in the Quadrature Decoder mode. 0b - Phase A and phase B encoding mode. 1b - Count and direction encoding mode.
2 QUADIR	FTM Counter Direction In Quadrature Decoder Mode Indicates the counting direction. 0b - Counting direction is decreasing (FTM counter decrement). 1b - Counting direction is increasing (FTM counter increment).
1	Timer Overflow Direction In Quadrature Decoder Mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
TOFDIR	Indicates if the TOF bit was set on the top or the bottom of counting. 0b - TOF bit was set on the bottom of counting. There was an FTM counter decrement and FTM counter changes from its minimum value (CNTIN register) to its maximum value (MOD register). 1b - TOF bit was set on the top of counting. There was an FTM counter increment and FTM counter changes from its maximum value (MOD register) to its minimum value (CNTIN register).
0 QUADEN	Quadrature Decoder Mode Enable Enables the Quadrature Decoder mode. In this mode, the phase A and B input signals control the FTM counter direction. The Quadrature Decoder mode has precedence over the other modes. This field is write protected. It can be written only when MODE[WPDIS] = 1. 0b - Quadrature Decoder mode is disabled. 1b - Quadrature Decoder mode is enabled.

43.4.3.20 Configuration (CONF)

Offset

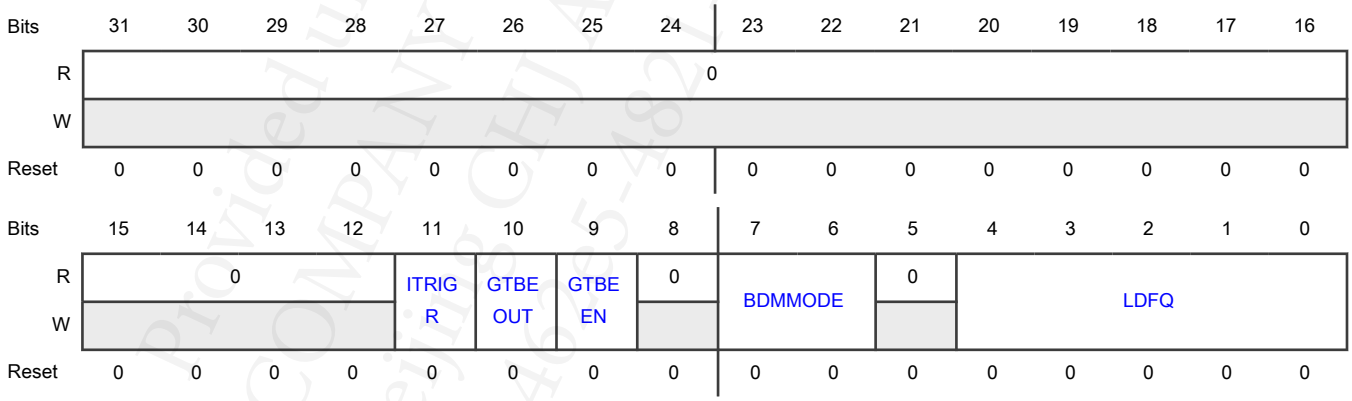
Register	Offset
CONF	84h

Function

This register selects the frequency of the reload opportunities, the FTM behavior in Debug mode, the use of an external global time base, and the global time base signal generation.

This register also controls if initialization trigger should be generated when a reload point is reached.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e65-4824-48a8-b251-390d3ab6f4c1

Fields

Field	Function
31-12 —	Reserved
11 ITRIGR	Initialization trigger on Reload Point This bit controls whether an initialization trigger is generated when a reload point configured by PWMLOAD register is reached considering the FTM_CONF[LDFQ] settings. 0b - Initialization trigger is generated on counter wrap events. 1b - Initialization trigger is generated when a reload point is reached.
10 GTBEOUT	Global Time Base Output Enables the global time base signal generation to other FTMs. 0b - A global time base signal generation is disabled. 1b - A global time base signal generation is enabled.
9 GTBEEN	Global Time Base Enable Configures the FTM to use an external global time base signal that is generated by another FTM. 0b - Use of an external global time base is disabled. 1b - Use of an external global time base is enabled.
8 —	Reserved
7-6 BDMMODE	Debug Mode Selects the FTM behavior in Debug mode. See Debug mode.
5 —	Reserved
4-0 LDFQ	Frequency of the Reload Opportunities The LDFQ[4:0] bits define the number of enabled reload opportunities should happen until an enabled reload opportunity becomes a reload point. See Reload Points LDFQ = 0: All reload opportunities are reload points. LDFQ = 1: There is a reload point each 2 reload oportunities. LDFQ = 2: There is a reload point each 3 reload oportunities. LDFQ = 3: There is a reload point each 4 reload oportunities. This pattern continues up to a maximum of 32.

Protected under NDA only
CONFIDENTIAL PROPRIETARY
Beijing Auto Motive Technology Co., Ltd.
c8c46e1-4888-b251-390d3ab624c1

43.4.3.21 Synchronization Configuration (SYNCONF)

Offset

Register	Offset
SYNCONF	8Ch

Function

This register selects the PWM synchronization configuration, SWOCTRL, INVCTRL and CNTIN registers synchronization, if FTM clears the TRIGj bit, where j = 0, 1, 2, when the hardware trigger j is detected.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0											HWSOC	HWINVC	HWOM	HWWRBUF	HWRSTCNT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			SWSOC	SWINVC	SWOM	SWWRBUF	SWRSTCNT	SYNCMODE	0	SWOC	INVC	0	CNTINC	0	HWTRIGM...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-21 —	Reserved
20 HWSOC	Software output control synchronization is activated by a hardware trigger 0b - A hardware trigger does not activate the SWOCTRL register synchronization. 1b - A hardware trigger activates the SWOCTRL register synchronization.
19 HWINVC	Inverting control synchronization is activated by a hardware trigger 0b - A hardware trigger does not activate the INVCTRL register synchronization. 1b - A hardware trigger activates the INVCTRL register synchronization.
18 HWOM	Output mask synchronization is activated by a hardware trigger 0b - A hardware trigger does not activate the OUTMASK register synchronization. 1b - A hardware trigger activates the OUTMASK register synchronization.
17 HWWRBUF	MOD, HCR, CNTIN, and CV registers synchronization is activated by a hardware trigger 0b - A hardware trigger does not activate MOD, HCR, CNTIN, and CV registers synchronization.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - A hardware trigger activates MOD, HCR, CNTIN, and CV registers synchronization.
16 HWRSTCNT	FTM counter synchronization is activated by a hardware trigger 0b - A hardware trigger does not activate the FTM counter synchronization. 1b - A hardware trigger activates the FTM counter synchronization.
15-13 —	Reserved
12 SWSOC	Software output control synchronization is activated by the software trigger 0b - The software trigger does not activate the SWOCTRL register synchronization. 1b - The software trigger activates the SWOCTRL register synchronization.
11 SWINVC	Inverting control synchronization is activated by the software trigger 0b - The software trigger does not activate the INVCTRL register synchronization. 1b - The software trigger activates the INVCTRL register synchronization.
10 SWOM	Output mask synchronization is activated by the software trigger 0b - The software trigger does not activate the OUTMASK register synchronization. 1b - The software trigger activates the OUTMASK register synchronization.
9 SWWRBUF	MOD, HCR, CNTIN, and CV registers synchronization is activated by the software trigger 0b - The software trigger does not activate MOD, HCR, CNTIN, and CV registers synchronization. 1b - The software trigger activates MOD, HCR, CNTIN, and CV registers synchronization.
8 SWRSTCNT	FTM counter synchronization is activated by the software trigger 0b - The software trigger does not activate the FTM counter synchronization. 1b - The software trigger activates the FTM counter synchronization.
7 SYNCMODE	Synchronization Mode Selects the PWM Synchronization mode. 0b - Legacy PWM synchronization is selected. 1b - Enhanced PWM synchronization is selected.
6 —	Reserved
5 SWOC	SWOCTRL Register Synchronization 0b - SWOCTRL register is updated with its buffer value at all rising edges of FTM input clock. 1b - SWOCTRL register is updated with its buffer value by the PWM synchronization.
4	INVCTRL Register Synchronization

Table continues on the next page...

Table continued from the previous page...

Field	Function
INVC	0b - INVCTRL register is updated with its buffer value at all rising edges of FTM input clock. 1b - INVCTRL register is updated with its buffer value by the PWM synchronization.
3 —	Reserved
2 CNTINC	CNTIN Register Synchronization 0b - CNTIN register is updated with its buffer value at all rising edges of FTM input clock. 1b - CNTIN register is updated with its buffer value by the PWM synchronization.
1 —	Reserved
0 HWTRIGMODE	Hardware Trigger Mode 0b - FTM clears the TRIGj bit when the hardware trigger j is detected, where j = 0, 1,2. 1b - FTM does not clear the TRIGj bit when the hardware trigger j is detected, where j = 0, 1,2.

43.4.3.22 FTM Inverting Control (INVCTRL)

Offset

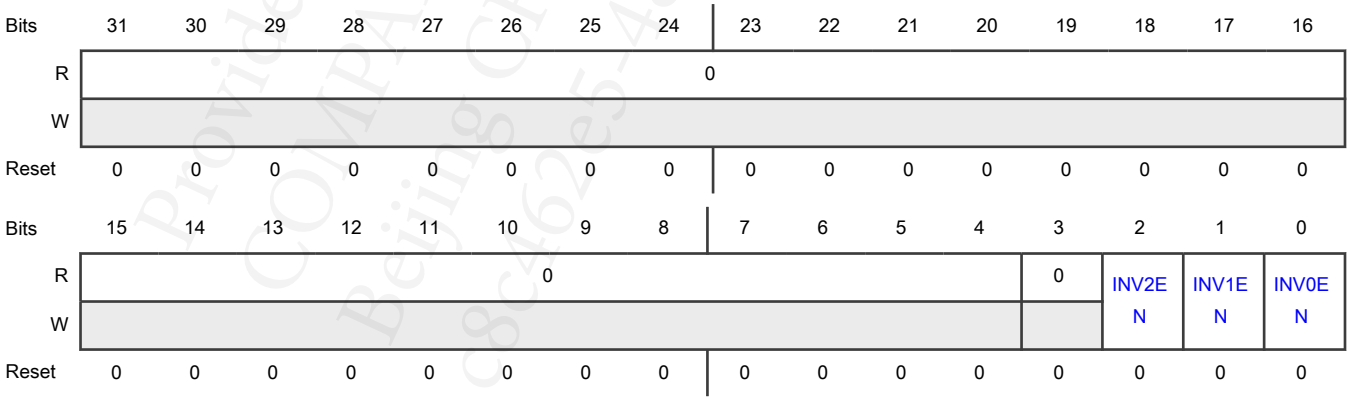
Register	Offset
INVCTRL	90h

Function

This register controls when the channel (n) output becomes the channel (n+1) output, and channel (n+1) output becomes the channel (n) output. Each INVmEN bit enables the inverting operation for the corresponding pair channels m.

This register has a write buffer. The INVmEN bit is updated by the INVCTRL register synchronization.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 —	Reserved
2 INV2EN	Pair Channels 2 Inverting Enable 0b - Inverting is disabled. 1b - Inverting is enabled.
1 INV1EN	Pair Channels 1 Inverting Enable 0b - Inverting is disabled. 1b - Inverting is enabled.
0 INV0EN	Pair Channels 0 Inverting Enable 0b - Inverting is disabled. 1b - Inverting is enabled.

43.4.3.23 FTM Software Output Control (SWOCTRL)

Offset

Register	Offset
SWOCTRL	94h

Function

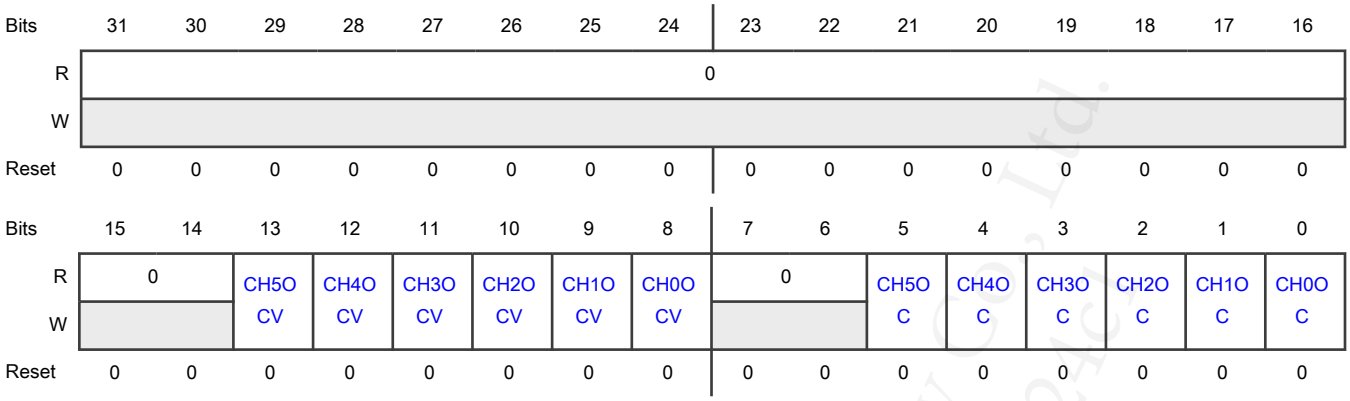
This register enables software control of channel (n) output and defines the value forced to the channel (n) output:

- The CH(n)OC bits enable the control of the corresponding channel (n) output by software.
- The CH(n)OCV bits select the value that is forced at the corresponding channel (n) output.

This register has a write buffer. The fields are updated by the SWOCTRL register synchronization.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15-14 —	Reserved
13-8 CHnOCV	Channel n Software Output Control Value 0b - The software output control forces 0 to the channel output. 1b - The software output control forces 1 to the channel output.
7-6 —	Reserved
5-0 CHnOC	Channel n Software Output Control Enable 0b - The channel output is not affected by software output control. 1b - The channel output is affected by software output control.

43.4.3.24 FTM PWM Load (PWMLOAD)

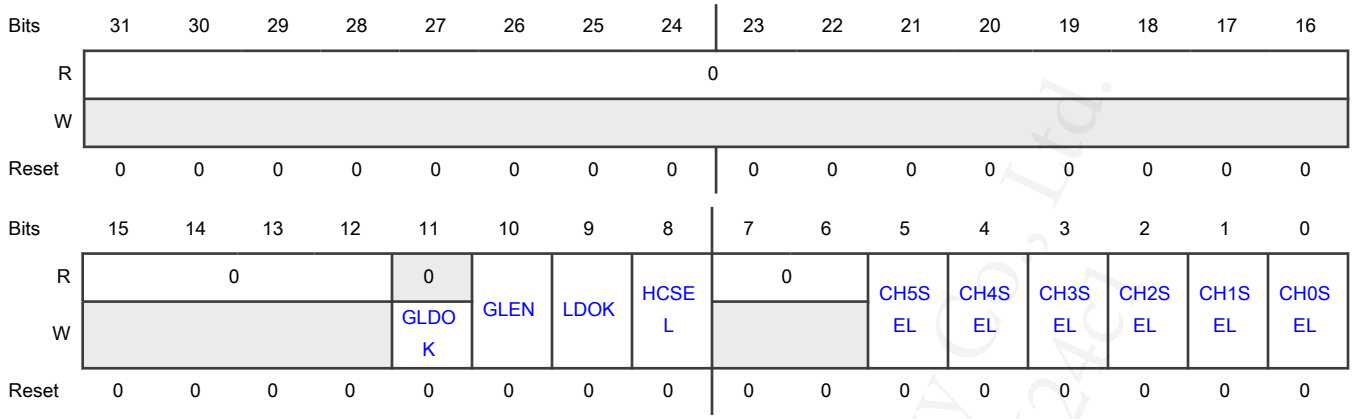
Offset

Register	Offset
PWMLOAD	98h

Function

Enables the reload of the MOD, HCR, CNTIN, C(n)V, and C(n+1)V registers with the values of their write buffers when the FTM counter changes from the MOD register value to its next value or when a channel (j) match occurs. A match occurs for channel (j) when FTM counter = C(j)V. A reload can also occurs when FTM counter = HCR register at a half cycle match. This register also controls the local and global load mechanisms.

Diagram



Fields

Field	Function
31-12 —	Reserved
11 GLDOK	Global Load OK This bit controls the global load mechanism. It generates a pulse at FTM module global load output with one FTM clock cycle width, which is used to set LDOK bits of FTM and other modules (including other FTMs). This bit is self-cleared and read value is always zero. The global load mechanism depends on SoC specific information. Refer to FTM SoC specific information to more details. 0b - No action. 1b - LDOK bit is set.
10 GLEN	Global Load Enable This bit enables the global load mechanism implemented by GLDOK. If GLEN bit is set, then an external event on the FTM global load input sets the LDOK bit. The clear of the LDOK bit is done by CPU writes '0' to the bit. 0b - Global Load Ok disabled. 1b - Global Load OK enabled. A pulse event on the module global load input sets the LDOK bit.
9 LDOK	Load Enable Enables the loading of the MOD, CNTIN, HCR and CV registers with the values of their buffers. The LDOK bit can also be set by the Global Load mechanism if GLEN bit is enabled. 0b - Loading updated values is disabled. 1b - Loading updated values is enabled.
8 HCSEL	Half Cycle Select

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit enables the half cycle match as a reload oportunity. A half cycle is defined by when the FTM counter matches the HCR register. 0b - Half cycle reload is disabled and it is not considered as a reload opportunity. 1b - Half cycle reload is enabled and it is considered as a reload opportunity.
7-6 —	Reserved
5-0 CHnSEL	Channel n Select 0b - Channel match is not included as a reload opportunity. 1b - Channel match is included as a reload opportunity.

43.4.3.25 Half Cycle Register (HCR)

Offset

Register	Offset
HCR	9Ch

Function

The Half Cycle Register contains the match value for FTM half cycle reload feature. After FTM counter reaches this value, a reload opportunity is generated if FTM_PWMLOAD[HCSEL] is enabled.

Writing to the HCR register latches the value into a buffer. The HCR register is updated with the value of its write buffer according to Registers updated from write buffers.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HCVAL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a2-9251-390d3ab624c1

Fields

Field	Function
31-16 —	Reserved
15-0 HCVAL	Half Cycle Value

43.4.3.26 Mirror of Modulo Value (MOD_MIRROR)

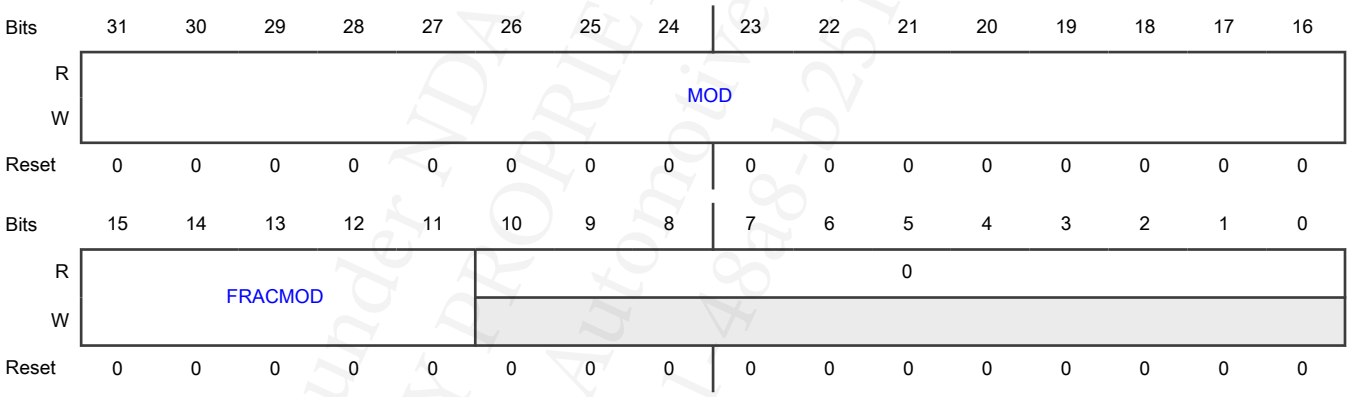
Offset

Register	Offset
MOD_MIRROR	200h

Function

This register contains the integer and fractional modulo value for the FTM counter.

Diagram



Fields

Field	Function
31-16 MOD	Mirror of the Modulo Integer Value See the field MOD of the register MOD.
15-11 FRACMOD	Modulo Fractional Value The modulo fractional value is used in the PWM period dithering. This value is added to an internal accumulator at the end of each PWM period.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Writes to the field FRACMOD are done on its write buffer. The FRACMOD is updated with its write buffer value according to Registers updated from write buffers. If FTMEN = 0, a write to SC register resets manually this write coherency mechanism.
10-0 —	Reserved

43.4.3.27 Mirror of Channel (n) Match Value (C0V_MIRROR - C5V_MIRROR)

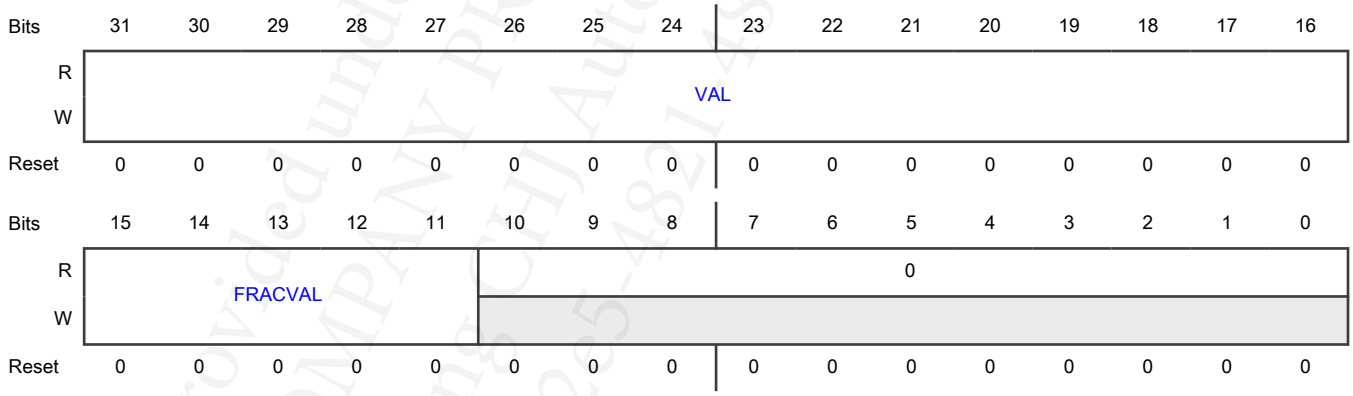
Offset

Register	Offset
C0V_MIRROR	204h
C1V_MIRROR	208h
C2V_MIRROR	20Ch
C3V_MIRROR	210h
C4V_MIRROR	214h
C5V_MIRROR	218h

Function

This register contains the integer and fractional value of the channel (n) match.

Diagram



Fields

Field	Function
31-16	Mirror of the Channel (n) Match Integer Value

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
18462554821 43a8-b251-390d8ab624c1

Table continued from the previous page...

Field	Function
VAL	See the field VAL of the register CnV.
15-11 FRACVAL	Channel (n) Match Fractional Value The channel (n) match fractional value is used in the PWM edge dithering. This value is added to the channel (n) internal accumulator at the end of each PWM period. Writes to the field FRACVAL are done on its write buffer. The FRACVAL is updated with its write buffer value according to Registers updated from write buffers. If FTMEN = 0, a write to CnSC register resets manually this write coherency mechanism.
10-0 —	Reserved

43.5 Functional Description

43.5.1 Clock source

The FTM has only one clock domain: the FTM input clock.

43.5.1.1 Counter clock source

The CLKS[1:0] bits select one of three possible clock sources for the FTM counter or disable the FTM counter. After any chip reset, CLKS[1:0] = 0:0 so no clock source is selected.

The CLKS[1:0] bits may be read or written at any time. Disabling the FTM counter by writing 0:0 to the CLKS[1:0] bits does not affect the FTM counter value or other registers.

The fixed frequency clock is an alternative clock source for the FTM counter that allows the selection of a clock other than the FTM input clock or an external clock. This clock input is defined by chip integration. Refer to the chip specific documentation for further information. Due to FTM hardware implementation limitations, the frequency of the fixed frequency clock must not exceed 1/2 of the FTM input clock frequency.

The external clock passes through a synchronizer clocked by the FTM input clock to assure that counter transitions are properly aligned to FTM input clock transitions.Therefore, to meet Nyquist criteria considering also jitter, the frequency of the external clock source must not exceed 1/4 of the FTM input clock frequency.

43.5.2 Prescaler

The selected counter clock source passes through a prescaler that is a 7-bit counter. The value of the prescaler is selected by the PS[2:0] bits. The following figure shows an example of the prescaler counter and FTM counter.

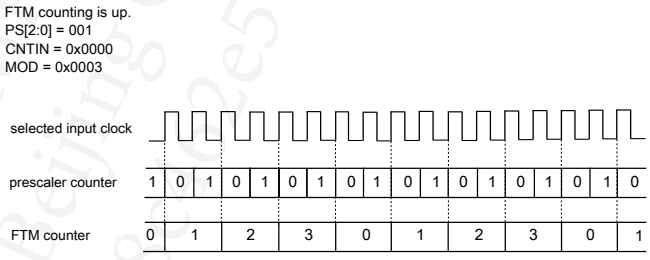


Figure 234. Example of the prescaler counter

43.5.3 Counter

The FTM has a 16-bit counter that is used by the channels either for input or output modes. The FTM counter clock is the selected clock divided by the prescaler.

The FTM counter has these modes of operation:

- Up counting
- Up-down counting
- Quadrature Decoder Mode

43.5.3.1 Up counting

Up counting is selected when:

- QUADEN = 0, and
- CPWMS = 0

CNTIN defines the starting value of the count and MOD defines the final value of the count, see the following figure. The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is reloaded with the value of CNTIN.

The FTM period when using up counting is (MOD – CNTIN + 0x0001) × period of the FTM counter clock.

The TOF bit is set when the FTM counter changes from MOD to CNTIN.

A counter event happens at the same time of TOF bit set when the FTM counter changes from MOD to CNTIN. See Counter events for more details.

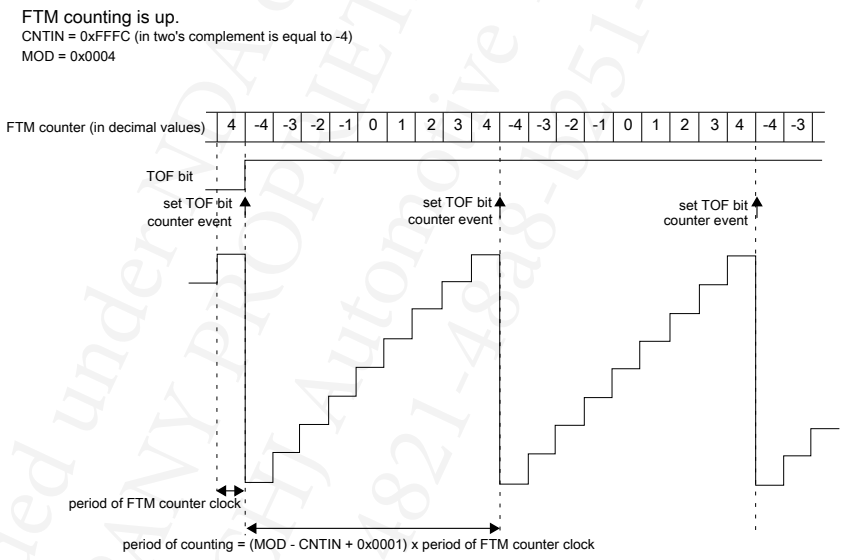


Figure 235. Example of FTM up and signed counting

Table 267. FTM counting based on CNTIN value

When	Then
CNTIN = 0x0000	The FTM counting is equivalent to TPM up counting, that is, up and unsigned counting. See the following figure.
CNTIN[15] = 1	The initial value of the FTM counter is a negative number in two's complement, so the FTM counting is up and signed.

Table continues on the next page...

Table 267. FTM counting based on CNTIN value (continued)

When	Then
CNTIN[15] = 0 and CNTIN ≠ 0x0000	The initial value of the FTM counter is a positive number, so the FTM counting is up and unsigned.

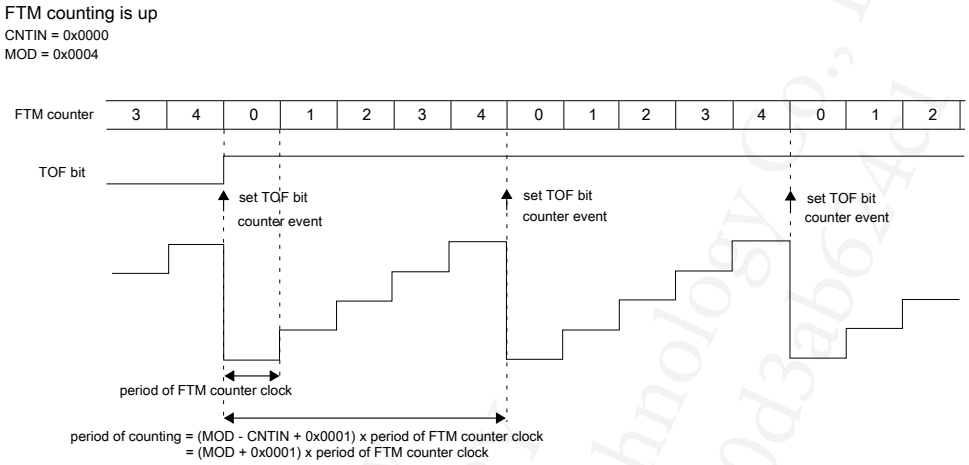


Figure 236. Example of FTM up counting with CNTIN = 0x0000

NOTE

- FTM operation is only valid when the value of the CNTIN register is less than the value of the MOD register, either in the unsigned counting or signed counting. It is the responsibility of the software to ensure that the values in the CNTIN and MOD registers meet this requirement. Any values of CNTIN and MOD that do not satisfy this criteria can result in unpredictable behavior.
- MOD = CNTIN is a redundant condition. In this case, the FTM counter is always equal to MOD and the TOF bit is set in each rising edge of the FTM counter clock.
- When MOD = 0x0000, CNTIN = 0x0000, for example after reset, and FTMEN = 1, the FTM counter remains stopped at 0x0000 until a non-zero value is written into the MOD or CNTIN registers.
- Setting CNTIN to be greater than the value of MOD is not recommended as this unusual setting may make the FTM operation difficult to comprehend. However, there is no restriction on this configuration, and an example is shown in the following figure.

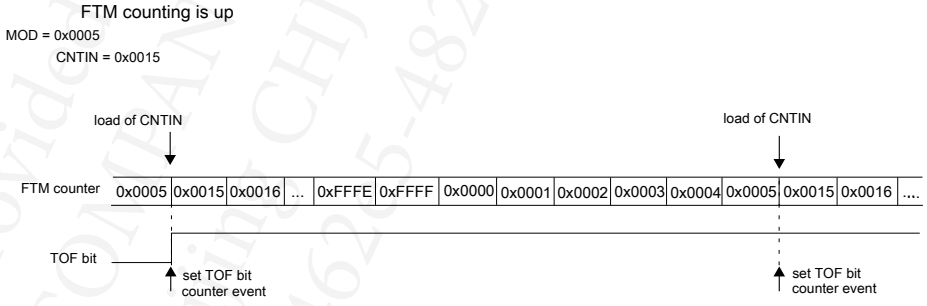


Figure 237. Example of up counting when the value of CNTIN is greater than the value of MOD

43.5.3.2 Up-down counting

Up-down counting is selected when:

- QUADEN = 0, and
- CPWMS = 1

CNTIN defines the starting value of the count and MOD defines the final value of the count. The value of CNTIN is loaded into the FTM counter, and the counter increments until the value of MOD is reached, at which point the counter is decremented until it returns to the value of CNTIN and the up-down counting restarts.

The FTM period when using up-down counting is 2 × (MOD – CNTIN) × period of the FTM counter clock.

The TOF bit is set when the FTM counter changes from MOD to (MOD – 1).

If (CNTIN = 0x0000), the FTM counting is equivalent to TPM up-down counting, that is, up-down and unsigned counting. See the following figure.

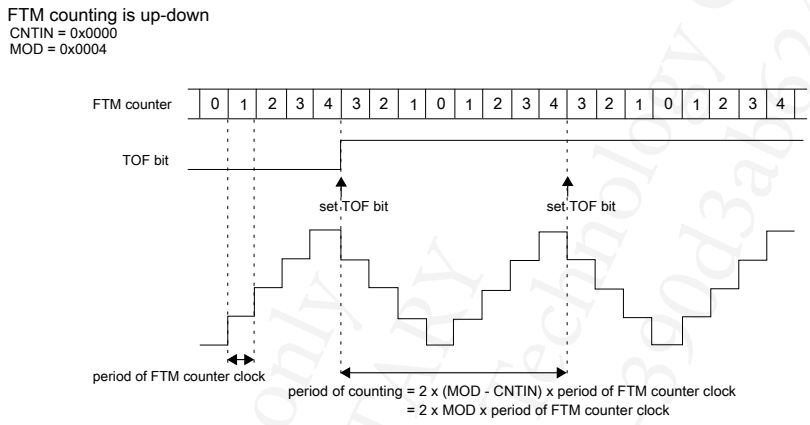


Figure 238. Example of up-down counting when CNTIN = 0x0000

NOTE

When CNTIN is different from zero in the up-down counting, a valid CPWM signal is generated:

- if CnV > CNTIN, or
- if CnV = 0 or if CnV[15] = 1. In this case, 0% CPWM is generated.

The figure below shows the possible counter events when in up-down counting mode. See Counter events for more details.

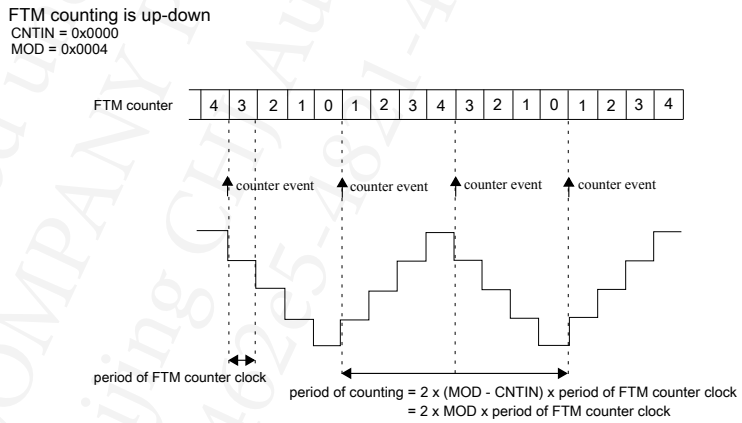


Figure 239. Example of counter events in up-down counting mode when CNTIN = 0x0000

43.5.3.3 Free running counter

If (FTMEN = 0) and (MOD = 0x0000 or MOD = 0xFFFF), the FTM counter is a free running counter. In this case, the FTM counter runs free from 0x0000 through 0xFFFF and the TOF bit is set when the FTM counter changes from 0xFFFF to 0x0000. See the following figure.

A counter event occurs at the same time of TOF bit set when the FTM counter changes from 0xFFFF to 0x0000. See Counter events for more details.

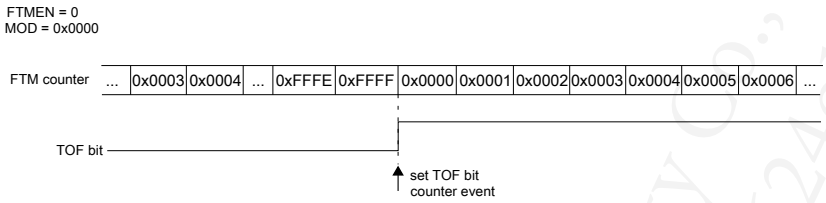


Figure 240. Example when the FTM counter is free running

The FTM counter is also a free running counter when:

- FTMEN = 1
- QUADEN = 0
- CPWMS = 0
- CNTIN = 0x0000, and
- MOD = 0xFFFF

43.5.3.4 Counter reset

Any one of the following cases resets the FTM counter to the value in the CNTIN register and the channels output to its initial value, except for channels in Output Compare mode.

- Any write to CNT.
- FTM counter synchronization.
- A channel in Input Capture mode with ICRST = 1 (FTM Counter Reset in Input Capture Mode).

Note that reseting the counter also generates a counter event. See Counter events for more details.

43.5.3.5 Counter events

Counter events can be used as reload opportunities to FTM register sychronization mechanism. See Reload Points for more details. There are some possible counter events depending on the counter mode. Please see the table below for more details.

Table 268. FTM counter events

When	Then
FTM counter is in up counting mode or freerunning	• A counter event happens at the same time of TOF bit set when the FTM counter changes from MOD to CNTIN (counter wrap). Figure at Up counting shows the counter event generation. • When in freerunning, there is a counter event when FTM counter changes from 0xFFFF to 0x0000. Figure at Free running counter shows the counter event generation.

Table continues on the next page...

Table 268. FTM counter events (continued)

When	Then
FTM counter is in up-down counting mode	• In up-down counting mode, there are two possible counter events when FTM counter turns from down to up counting and when counter turns from up to down counting. User can select which point will be used to generate the counter event. Figure at Up-down counting shows the possible counter events.
FTM counter is reseted (see Counter reset) or a value different from zero is written at CLKS field	• In up-counting mode, all counter reset events or a write in the CLKS with a value different from zero generates a counter event. • In up-down counting mode, counter reset events only generates a counter event if the minimum load point when FTM counter turns from down to up counting is configured. A write in the CLKS with a value different from zero always generates a counter event in up-down counting mode.

43.5.4 Channel Modes

The following table shows the channel modes selection.

Table 269. Channel Modes Selection

DECAPEN	MCOMBINE	COMBINE	CPWMS	MSB:MSA	ELSB:ELSA	Mode	Configuration
X	X	X	X	XX	00	Pin not used for FTM—revert the channel pin to general purpose I/O or other peripheral control	
0	0	0	0	00	01	Input Capture	Capture on Rising Edge Only
					10		Capture on Falling Edge Only
					11		Capture on Rising or Falling Edge
				01	01	Output Compare	Toggle Output on match
					10		Clear Output on match
					11		Set Output on match
				1X	10	Edge-Aligned PWM	High-true pulses (clear

Table continues on the next page...

Table 269. Channel Modes Selection (continued)

DECAPEN	MCOMBINE	COMBINE	CPWMS	MSB:MSA	ELSB:ELSA	Mode	Configuration
							Output on match)
					X1		Low-true pulses (set Output on match)
			1	XX	10	Center-Aligned PWM	High-true pulses (clear Output on match-up)
					X1		Low-true pulses (set Output on match-up)
		1	0	XX	10	Combine PWM	High-true pulses (set on channel (n) match, and clear on channel (n+1) match)
					X1		Low-true pulses (clear on channel (n) match, and set on channel (n+1) match)
	1	0	X	XX	XX	Reserved for future use	
	1	1	0	XX	10	Modified Combine PWM	High-true pulses (set on channel (n) match, and clear on channel (n+1) match)
					X1		Low-true pulses (clear on channel (n) match, and set on channel (n+1) match)
1	0	0	0	X0	See Table 270.	Dual Edge Capture	One-Shot Capture mode

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-49a8-b251-390d3ab6241

Table 269. Channel Modes Selection (continued)

DECAPEN	MCOMBINE	COMBINE	CPWMS	MSB:MSA	ELSB:ELSA	Mode	Configuration
				X1			Continuous Capture mode

Table 270. Dual Edge Capture Mode — Edge Polarity Selection

ELSB	ELSA	Channel Port Enable	Detected Edges
0	0	Disabled	No edge
0	1	Enabled	Rising edge
1	0	Enabled	Falling edge
1	1	Enabled	Rising and falling edges

43.5.5 Input Capture Mode

The Input Capture mode is selected when:

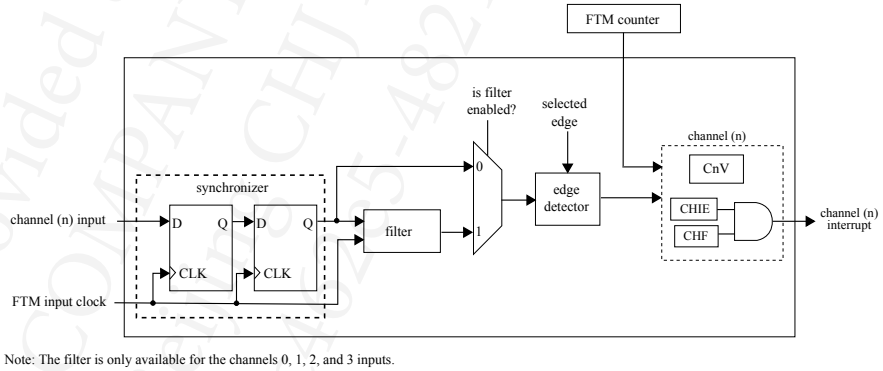
- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 0
- CPWMS = 0
- MSB:MSA = 0:0, and
- ELSB:ELSA ≠ 0:0

When a selected edge occurs on the channel input, the current value of the FTM counter is captured into the CnV register, at the same time the CHF bit is set and the channel interrupt is generated if enabled by CHIE = 1. See the following figure.

When a channel is configured for input capture, the FTMxCHn pin is an edge-sensitive input. ELSB:ELSA bits determine which edge, falling or rising, triggers input-capture event.

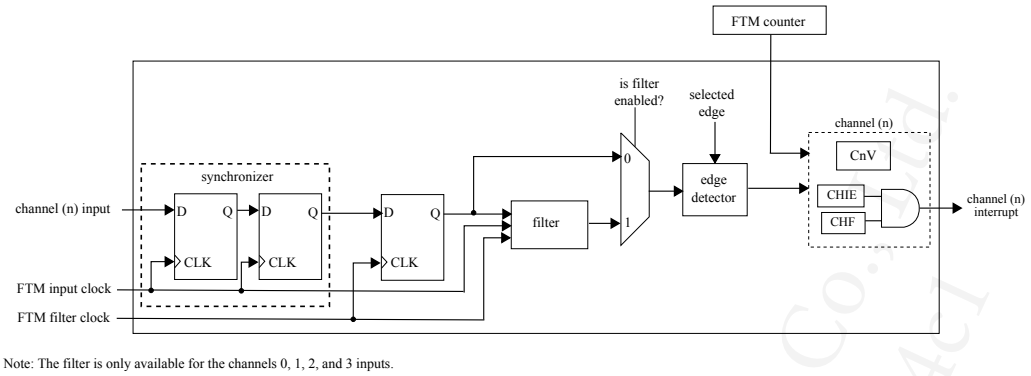
Writes to the CnV register are ignored in input capture mode.

While in Debug mode, the input capture function works as configured. When a selected edge event occurs, the FTM counter value, which is frozen because of Debug, is captured into the CnV register and the CHF bit is set.



Note: The filter is only available for the channels 0, 1, 2, and 3 inputs.

Figure 241. Diagram for Input Capture Mode when FLTPS[3:0] = 0



Note: The filter is only available for the channels 0, 1, 2, and 3 inputs.

Figure 242. Diagram for Input Capture Mode when FLTPS[3:0] ≠ 0

43.5.5.1 Filter for Input Capture Mode

The filter is only available on channels 0, 1, 2, and 3.

If FLTPS[3:0] = 0, the channel input after being synchronized by FTM input clock (Figure 241) is the filter input.

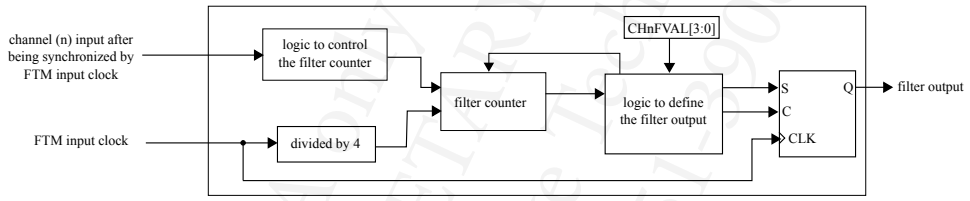


Figure 243. Channel Input Filter when FLTPS[3:0] = 0

If FLTPS[3:0] ≠ 0, the channel input after being synchronized by FTM input clock and being sampled by FTM filter clock (Figure 242) is the filter input.

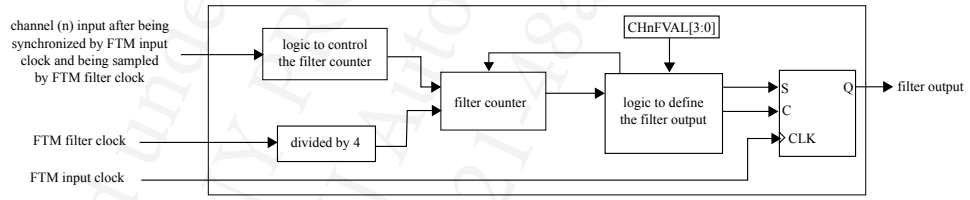


Figure 244. Channel Input Filter when FLTPS[3:0] ≠ 0

NOTE

The maximum frequency for the channel input to be detected correctly is FTM filter clock divided by 4, which is required to meet Nyquist criteria for signal sampling.

When there is a state change in the channel input, the counter is reset and starts counting up. As long as the new state is stable on the channel input, the counter continues to increment. When the counter is equal to CHnFVAL[3:0], the new channel input signal value is validated. It is then transmitted as a pulse to the edge detector.

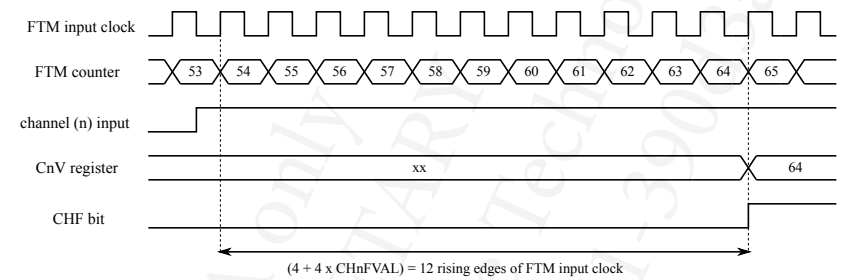
If the opposite edge appears on the channel input signal before it can be validated, the counter is reset. At the next input transition, the counter starts counting again. If a pulse is sampled as a value less than (CHnFVAL[3:0] x 4) consecutive rising edges of FTM filter clock, it is regarded as a glitch and is not passed on to the edge detector.

The table below shows the delay that is added by the FTM channel input filter according to its configuration.

Table 271. FTM Channel Input Filter Delay

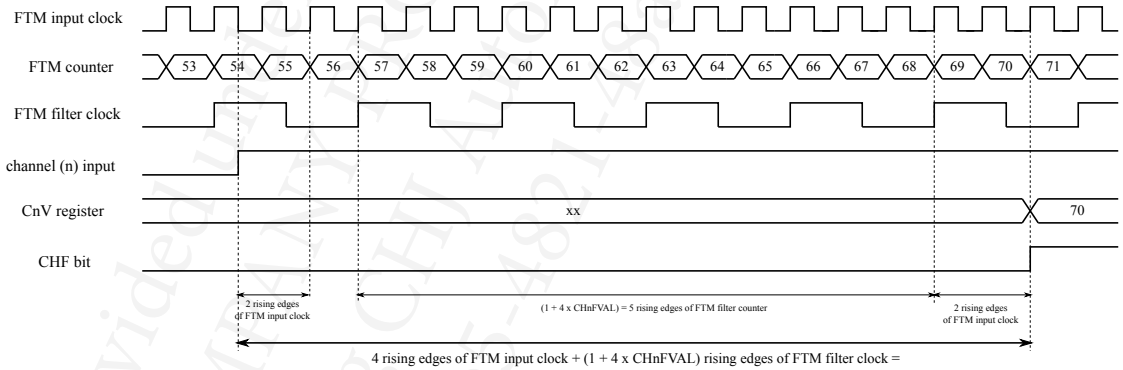
FTM channel input filter	FLTPS[3:0] bits	Number of rising edges between the selected edge on channel input and setting CHF bit
• channel does not have the input filter, or • channel input filter is disabled (CHnFVAL[3:0] = 0)	FLTPS[3:0] = 0	• 3 rising edges of FTM input clock
	FLTPS[3:0] ≠ 0	• 3 rising edges of FTM input clock, plus • 1 rising edge of FTM filter clock
• channel has the input filter, and • channel input filter is enabled (CHnFVAL[3:0] ≠ 0)	FLTPS[3:0] = 0	• (4 + 4 × CHnFVAL[3:0]) rising edges of FTM input clock
	FLTPS[3:0] ≠ 0	• 4 rising edges of FTM input clock, plus • (1 + 4 × CHnFVAL[3:0]) rising edges of FTM filter clock

The following figures illustrate two examples of channel input filter.



Note:
PS[2:0] = 3'b000
channel (n) in input capture mode with capture only on rising edges
CHnFVAL[3:0] = 4'h2 (channel (n) input filter is enabled)

Figure 245. Example of Channel Input Filter when FLTPS[3:0] = 0



Note:
PS[2:0] = 3'b000
channel (n) in input capture mode with capture only on rising edges
CHnFVAL[3:0] = 4'h1 (channel (n) input filter is enabled)
FLTPS[3:0] = 4'h2 (divide by 3)

Figure 246. Example of Channel Input Filter when FLTPS[3:0] ≠ 0

43.5.5.2 FTM Counter Reset in Input Capture Mode

If the channel (n) is in input capture mode and CnSC[ICRST = 1], then when the selected input capture event occurs in the channel (n) input signal, the current value of the FTM counter is captured into the CnV register, the CHF bit is set, the channel (n) interrupt is generated (if CHIE = 1) and the FTM counter is reset to the CNTIN register value.

This allows the FTM to measure a period/pulse being applied to the channel (n) input (number of the FTM input clocks) without having to implement a subtraction calculation in software subsequent to the event occurring.

The figure below shows the FTM counter reset when the selected input capture event is detected in a channel in input capture mode with ICRST = 1.

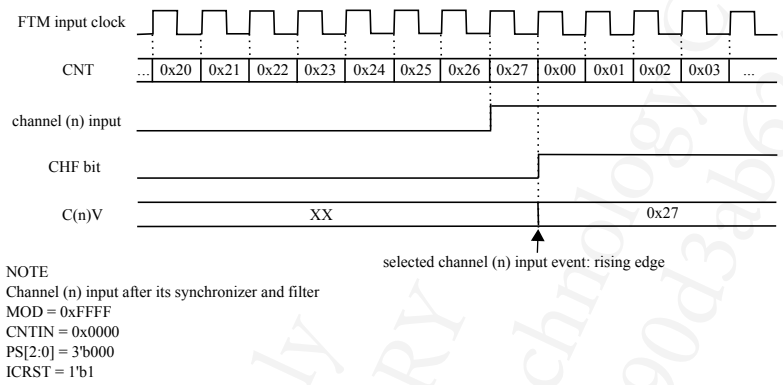


Figure 247. Example of the Input Capture mode with ICRST = 1

NOTE

- It is expected that the ICRST bit be set only when the channel is in input capture mode.
- If the FTM counter is reset because the channel is in input capture mode with ICRST = 1, then the prescaler counter (Prescaler) is also reset.

43.5.6 Output Compare mode

The Output Compare mode is selected when:

- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 0
- CPWMS = 0, and
- MSB:MSA = 0:1

In Output Compare mode, the FTM can generate timed pulses with programmable position, polarity, duration, and frequency. When the counter matches the value in the CnV register of an output compare channel, the channel (n) output can be set, cleared, or toggled.

When a channel is initially configured to Toggle mode, the previous value of the channel output is held until the first output compare event occurs.

The CHF bit is set and the channel (n) interrupt is generated if CHIE = 1 at the channel (n) match (FTM counter = CnV).

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e34821-48a8-b251-390d3ab624c1

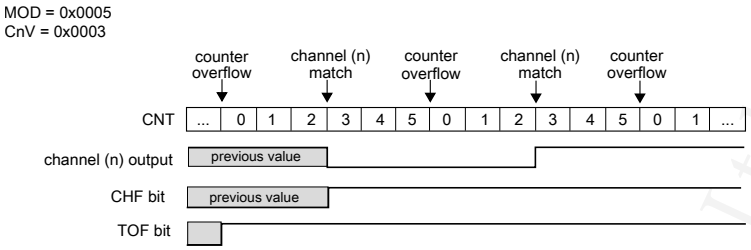


Figure 248. Example of the Output Compare mode when the match toggles the channel output

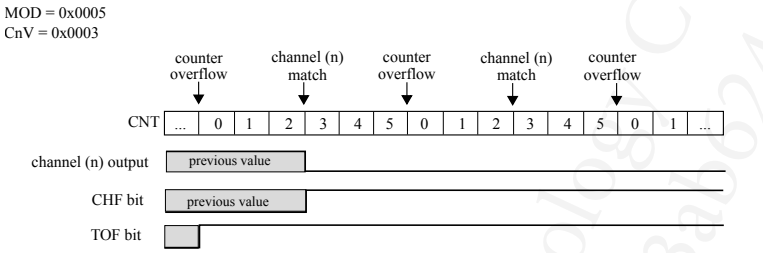


Figure 249. Example of the Output Compare mode when the match clears the channel output

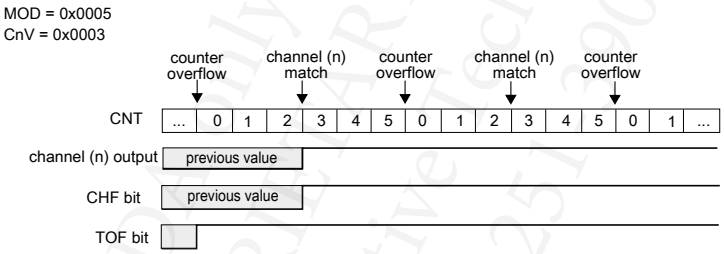


Figure 250. Example of the Output Compare mode when the match sets the channel output

If (ELSB:ELSA = 0:0) when the counter reaches the value in the CnV register, the CHF bit is set and the channel (n) interrupt is generated if CHIE = 1, however the channel (n) output is not modified and controlled by FTM.

43.5.7 Edge-Aligned PWM (EPWM) mode

The Edge-Aligned mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 0
- CPWMS = 0, and
- MSB = 1

The EPWM period is determined by (MOD − CNTIN + 0x0001) and the pulse width (duty cycle) is determined by (CnV − CNTIN).

The CHF bit is set and the channel (n) interrupt is generated if CHIE = 1 at the channel (n) match (FTM counter = CnV), that is, at the end of the pulse width.

This type of PWM signal is called edge-aligned because the leading edges of all PWM signals are aligned with the beginning of the period, which is the same for all channels within an FTM.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cc42e5-4821-48a2-8251-2903a624c1

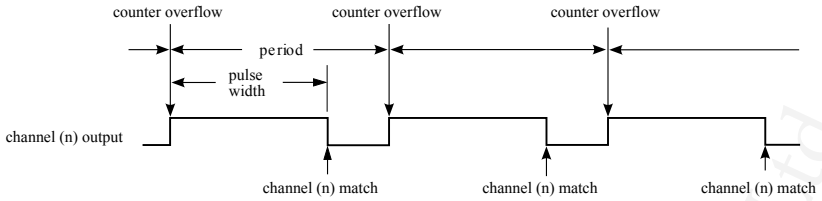


Figure 251. EPWM period and pulse width with ELSB:ELSA = 1:0

If (ELSB:ELSA = 0:0) when the counter reaches the value in the CnV register, the CHF bit is set and the channel (n) interrupt is generated if CHIE = 1, however the channel (n) output is not controlled by FTM.

If (ELSB:ELSA = 1:0), then the channel (n) output is forced high at the counter overflow when the CNTIN register value is loaded into the FTM counter, and it is forced low at the channel (n) match (FTM counter = CnV). See the following figure.

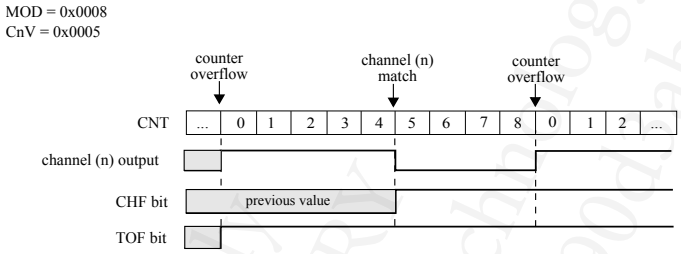


Figure 252. EPWM signal with ELSB:ELSA = 1:0

If (ELSB:ELSA = X:1), then the channel (n) output is forced low at the counter overflow when the CNTIN register value is loaded into the FTM counter, and it is forced high at the channel (n) match (FTM counter = CnV). See the following figure.

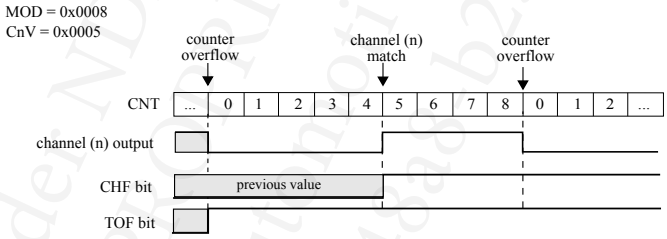


Figure 253. EPWM signal with ELSB:ELSA = X:1

If (CnV = 0x0000), then the channel (n) output is a 0% duty cycle EPWM signal and CHF bit is not set even when there is the channel (n) match.

If (CnV > MOD), then the channel (n) output is a 100% duty cycle EPWM signal and CHF bit is not set. Therefore, MOD must be less than 0xFFFF in order to get a 100% duty cycle EPWM signal.

NOTE

When CNTIN is different from zero the following EPWM signals can be generated:

- 0% EPWM signal if CnV = CNTIN,
- EPWM signal between 0% and 100% if CNTIN < CnV <= MOD,
- 100% EPWM signal when CNTIN > CnV or CnV > MOD.

43.5.8 Center-Aligned PWM (CPWM) mode

The Center-Aligned mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 0, and
- CPWMS = 1

The CPWM pulse width (duty cycle) is determined by 2 × (CnV − CNTIN) and the period is determined by 2 × (MOD − CNTIN). See the following figure. MOD must be kept in the range of 0x0001 to 0x7FFF because values outside this range can produce ambiguous results.

In the CPWM mode, the FTM counter counts up until it reaches MOD and then counts down until it reaches CNTIN.

The CHF bit is set and channel (n) interrupt is generated (if CHIE = 1) at the channel (n) match (FTM counter = CnV) when the FTM counting is down (at the begin of the pulse width) and when the FTM counting is up (at the end of the pulse width).

This type of PWM signal is called center-aligned because the pulse width centers for all channels are aligned with the value of CNTIN.

The other channel modes are not compatible with the up-down counter (CPWMS = 1). Therefore, all FTM channels must be used in CPWM mode when (CPWMS = 1).

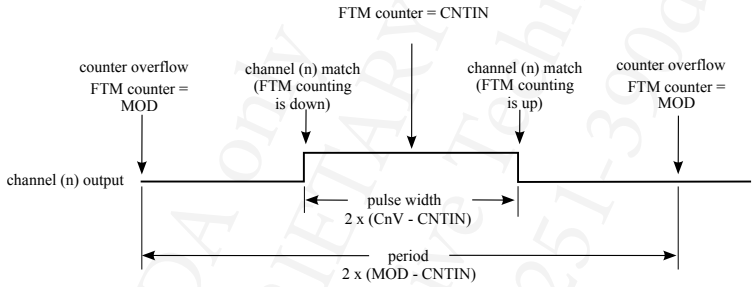


Figure 254. CPWM period and pulse width with ELSB:ELSA = 1:0

If (ELSB:ELSA = 0:0) when the FTM counter reaches the value in the CnV register, the CHF bit is set and the channel (n) interrupt is generated (if CHIE = 1), however the channel (n) output is not controlled by FTM.

If (ELSB:ELSA = 1:0), then the channel (n) output is forced high at the channel (n) match (FTM counter = CnV) when counting down, and it is forced low at the channel (n) match when counting up. See the following figure.

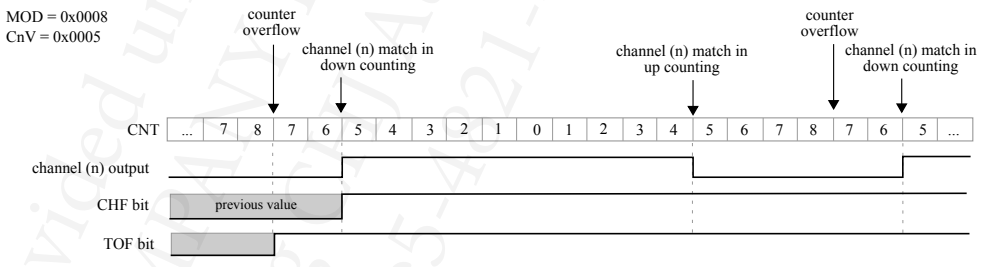


Figure 255. CPWM signal with ELSB:ELSA = 1:0

If (ELSB:ELSA = X:1), then the channel (n) output is forced low at the channel (n) match (FTM counter = CnV) when counting down, and it is forced high at the channel (n) match when counting up. See the following figure.

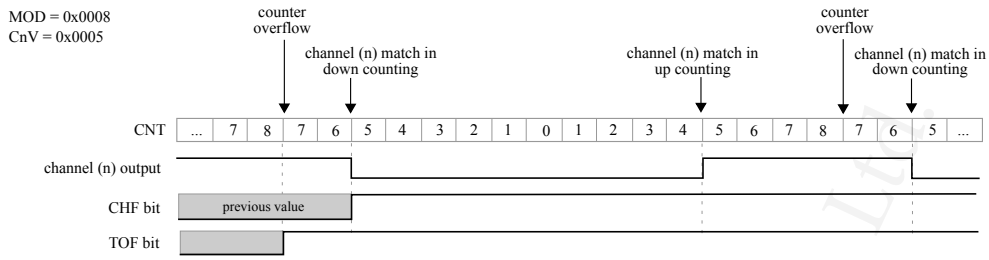


Figure 256. CPWM signal with ELSB:ELSA = X:1

If (CnV = 0x0000) or CnV is a negative value, that is (CnV[15] = 1), then the channel (n) output is a 0% duty cycle CPWM signal and CHF bit is not set even when there is the channel (n) match.

If CnV is a positive value, that is (CnV[15] = 0), (CnV ≥ MOD), and (MOD ≠ 0x0000), then the channel (n) output is a 100% duty cycle CPWM signal and CHF bit is not set even when there is the channel (n) match. This implies that the usable range of periods set by MOD is 0x0001 through 0x7FFE, 0x7FFF if you do not need to generate a 100% duty cycle CPWM signal. This is not a significant limitation because the resulting period is much longer than required for normal applications.

The CPWM mode must not be used when the FTM counter is a free running counter.

43.5.9 Combine mode

The Combine mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- MCOMBINE = 0
- COMBINE = 1, and
- CPWMS = 0

In Combine mode, an even channel (n) and adjacent odd channel (n+1) are combined to generate a PWM signal in the channel (n) output.

In the Combine mode, the PWM period is determined by (MOD − CNTIN + 0x0001) and the PWM pulse width (duty cycle) is determined by (|C(n+1)V − C(n)V|).

The channel (n) CHF bit is set and its interrupt is generated, if channel (n) CHIE = 1, at the channel (n) match (FTM counter = C(n)V). The channel (n+1) CHF bit is set and its interrupt is generated, if channel (n+1) CHIE = 1, at the channel (n+1) match (FTM counter = C(n+1)V).

If channel (n) ELSB:ELSA = 1:0, then the channel (n) output is forced low at the beginning of the period (FTM counter = CNTIN) and at the channel (n+1) match (FTM counter = C(n+1)V). It is forced high at the channel (n) match (FTM counter = C(n)V). See the following figure.

If channel (n) ELSB:ELSA = X:1, then the channel (n) output is forced high at the beginning of the period (FTM counter = CNTIN) and at the channel (n+1) match (FTM counter = C(n+1)V). It is forced low at the channel (n) match (FTM counter = C(n)V). See the following figure.

In Combine mode, the channel (n+1) ELSB:ELSA bits are not used in the generation of the channels (n) and (n+1) output. However, if channel (n) ELSB:ELSA = 0:0, then the channel (n) output is not controlled by FTM, and if channel (n+1) ELSB:ELSA = 0:0, then the channel (n+1) output is not controlled by FTM.

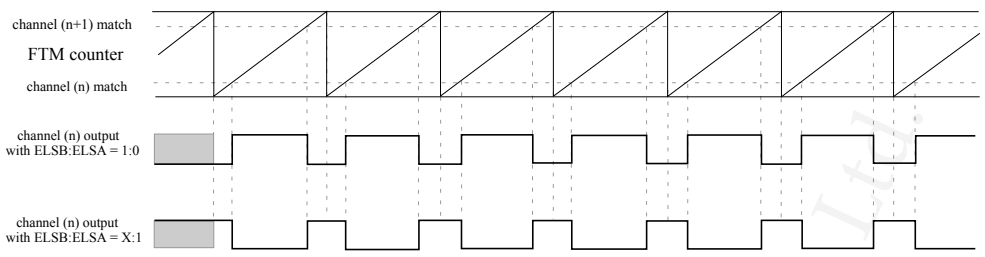


Figure 257. Combine mode

The following figures illustrate the PWM signals generation using Combine mode.

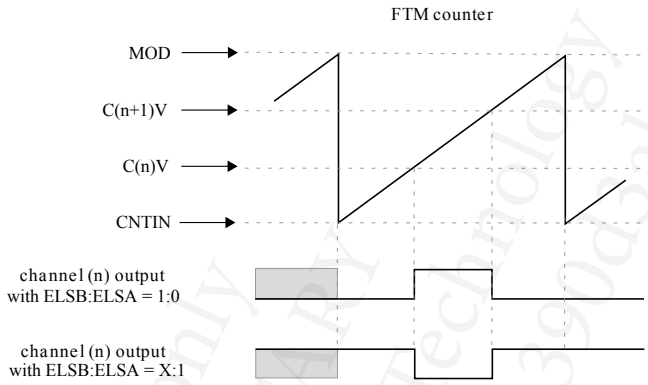


Figure 258. Channel (n) output if (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V < C(n+1)V)

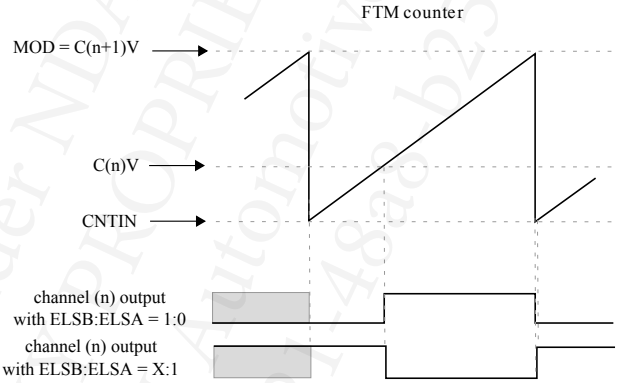


Figure 259. Channel (n) output if (CNTIN < C(n)V < MOD) and (C(n+1)V = MOD)

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4621-48a8-b251-390a5ab624c1

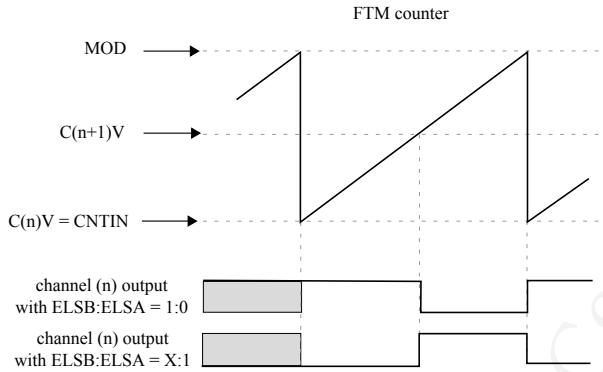


Figure 260. Channel (n) output if (C(n)V = CNTIN) and (CNTIN < C(n+1)V < MOD)

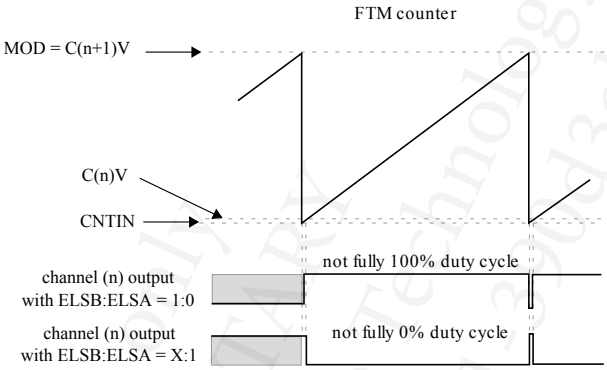


Figure 261. Channel (n) output if (CNTIN < C(n)V < MOD) and (C(n)V is Almost Equal to CNTIN) and (C(n+1)V = MOD)

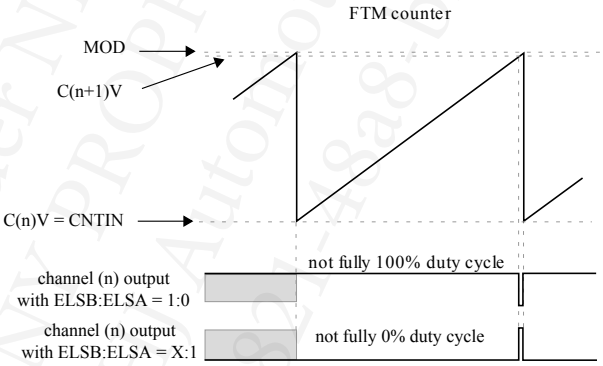


Figure 262. Channel (n) output if (C(n)V = CNTIN) and (CNTIN < C(n+1)V < MOD) and (C(n+1)V is Almost Equal to MOD)

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-8231-39dd3ab624c1

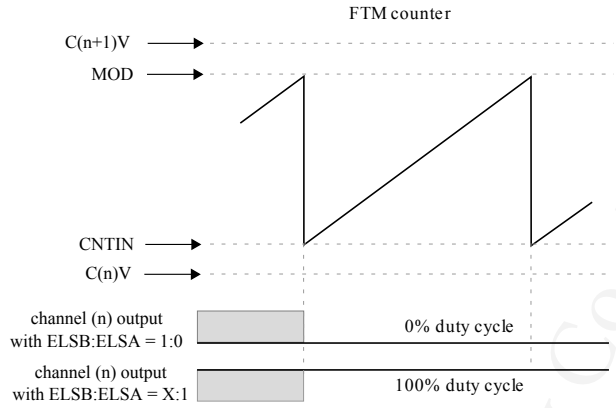


Figure 263. Channel (n) output if C(n)V and C(n+1)V are not between CNTIN and MOD

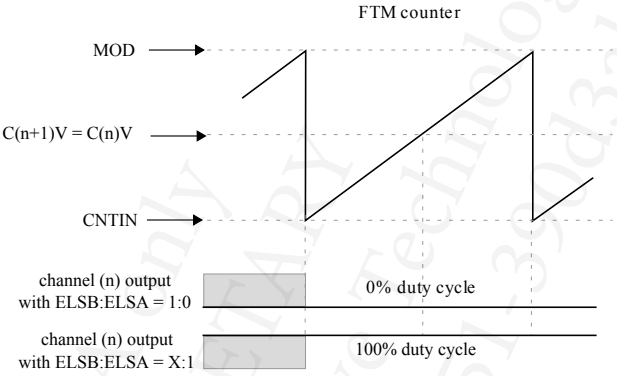


Figure 264. Channel (n) output if (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V = C(n+1)V)

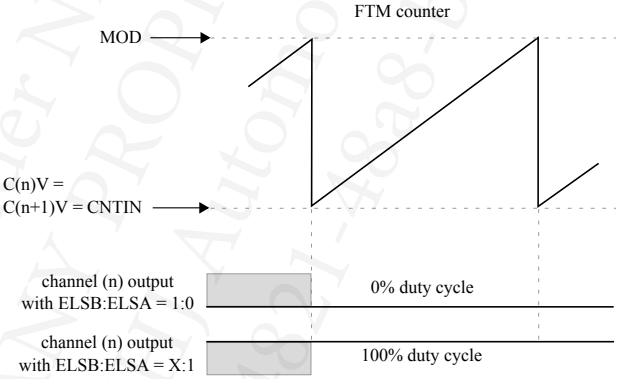


Figure 265. Channel (n) output if (C(n)V = C(n+1)V = CNTIN)

Provided under NXP Only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4801-48a8-b251-39dd3ab624c1

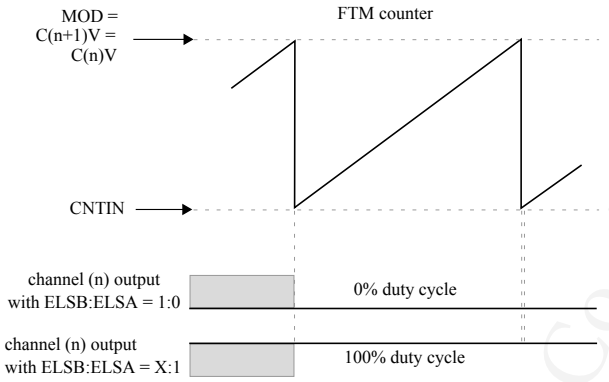


Figure 266. Channel (n) output if (C(n)V = C(n+1)V = MOD)

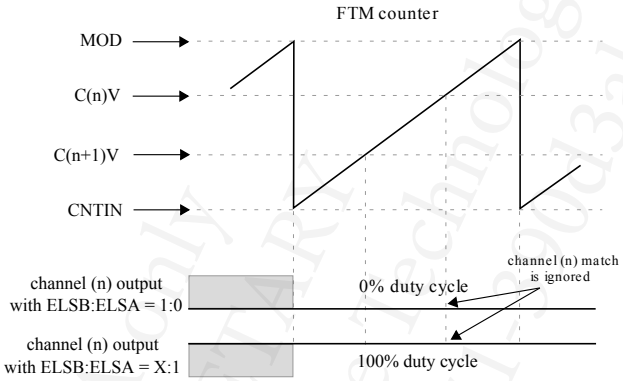


Figure 267. Channel (n) output if (CNTIN < C(n)V < MOD) and (CNTIN < C(n+1)V < MOD) and (C(n)V > C(n+1)V)

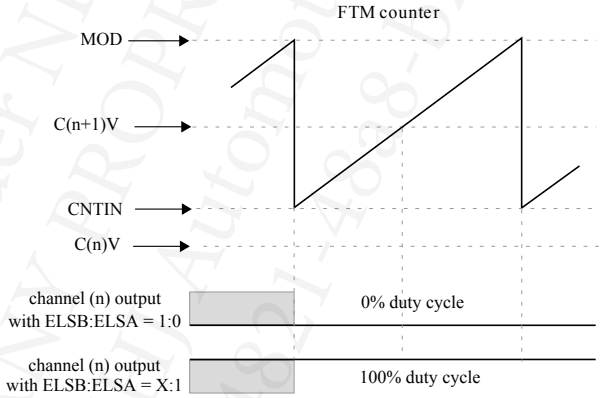


Figure 268. Channel (n) output if (C(n)V < CNTIN) and (CNTIN < C(n+1)V < MOD)

Provided under NDA only
COMPANY PROPRIETARY
Beijing CUBIC Automotive Technology Co., Ltd.
c8c462e5-4821-4888-b2b1-5b0d3ab624c1

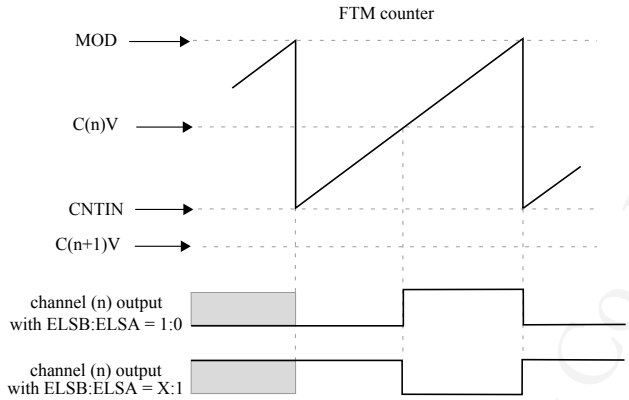


Figure 269. Channel (n) output if (C(n+1)V < CNTIN) and (CNTIN < C(n)V < MOD)

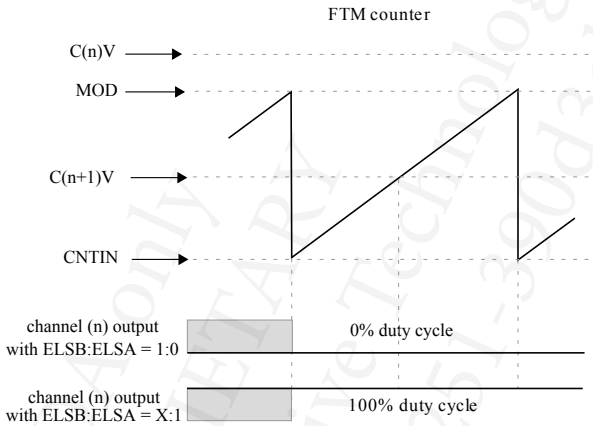


Figure 270. Channel (n) output if (C(n)V > MOD) and (CNTIN < C(n+1)V < MOD)

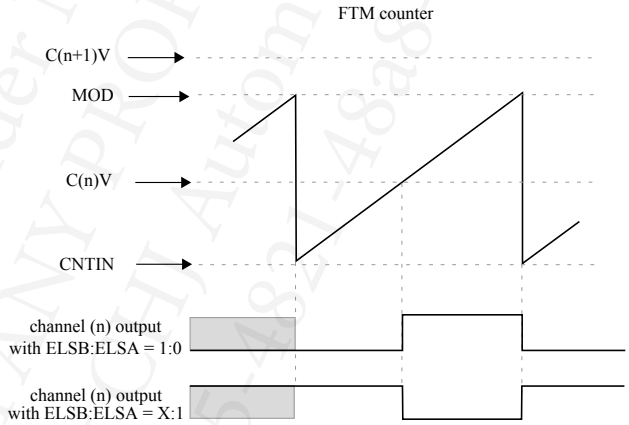


Figure 271. Channel (n) output if (C(n+1)V > MOD) and (CNTIN < C(n)V < MOD)

Provided under NDA only
COMPANY PROPRIETARY
Beijing C-Sky Automotive Technology Co., Ltd.
c8c46255-4821-48a8-b251-3a0d3ab624c1

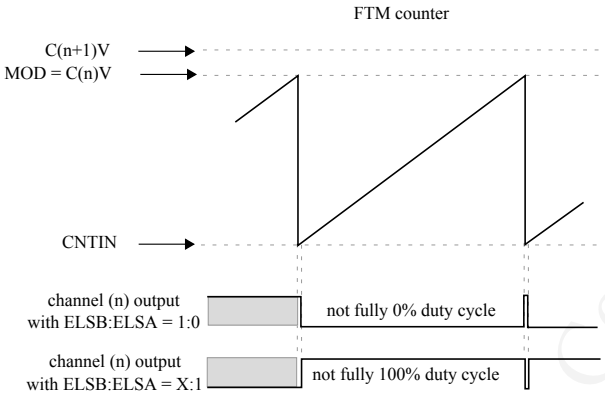


Figure 272. Channel (n) output if (C(n+1)V > MOD) and (CNTIN < C(n)V = MOD)

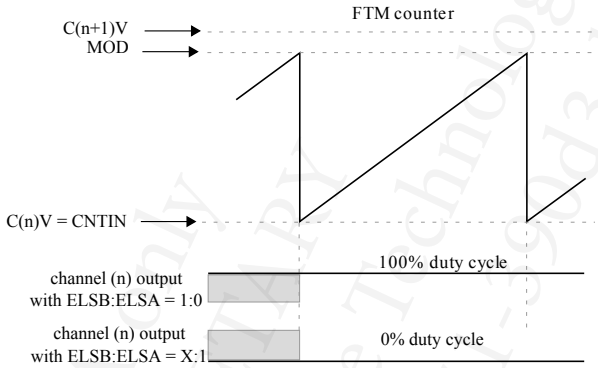


Figure 273. Channel (n) output if (C(n)V = CNTIN) and (C(n+1)V > MOD)

43.5.9.1 Asymmetrical PWM

In Combine mode and Modified Combine PWM Mode, the PWM first edge (channel (n) match: FTM counter = C(n)V) is independent of the PWM second edge (channel (n+1) match: FTM counter = C(n+1)V).

43.5.10 Modified Combine PWM Mode

The Modified Combine PWM mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- MCOMBINE = 1
- COMBINE = 1, and
- CPWMS = 0

The Modified Combine PWM mode is intended to support the generation of PWM signals where the period is not modified while the signal is being generated, but the duty cycle will be varied. In this mode, an even channel (n) and adjacent odd channel (n+1) are combined to generate a PWM signal in the channel (n) output. Thus, the channel (n) match edge is fixed and the channel (n+1) match edge can be varied.

When a pair of channels is in Modified Combine PWM mode, it is recommend that the other pairs also be in Modified Combine PWM mode.

In the Modified Combine PWM mode, assuming that CNTIN ≥ 0, MOD > 0, and CNTIN < MOD:

- The PWM period is determined by (MOD - CNTIN + 0x0001);

- The channel (n) PWM duty cycle is calculated according to the following table.

Table 272. Modified Combine PWM Mode - Duty Cycles

Channel (n) PWM Duty Cycle	Condition
0% duty cycle	For CNTIN ≤ (C(n)V and C(n+1)V) ≤ MOD: C(n)V = C(n+1)V
duty cyle between 0% and 100%	For CNTIN ≤ (C(n)V and C(n+1)V) ≤ MOD: • if (C(n)V < C(n+1)V), then the duty cycle is (C(n+1)V - C(n)V) • if (C(n)V > C(n+1)V), then the duty cycle is [(MOD - C(n)V) + (C(n+1)V - CNTIN) + 1]
100% duty cyle	CNTIN ≤ C(n)V ≤ MOD and C(n+1)V > MOD

The channel (n) CHF bit is set and its interrupt is generated, if channel (n) CHIE = 1, at the channel (n) match (FTM counter = C(n)V). The channel (n+1) CHF bit is set and its interrupt is generated, if channel (n+1) CHIE = 1, at the channel (n+1) match (FTM counter = C(n+1)V).

If channel (n) ELSB:ELSA = 1:0, then the channel (n) output is forced high at the channel (n) match (FTM counter = C(n)V) and it is forced low at the channel (n+1) match (FTM counter = C(n+1)V).

If channel (n) ELSB:ELSA = X:1, then the channel (n) output is forced low at the channel (n) match (FTM counter = C(n)V) and it is forced high at the channel (n+1) match (FTM counter = C(n+1)V).

In Modified Combine PWM mode, the channel (n+1) ELSB:ELSA bits are not used in the generation of the channels (n) and (n+1) output. However, if channel (n) ELSB:ELSA = 0:0, then the channel (n) output is not controlled by FTM, and if channel (n+1) ELSB:ELSA = 0:0, then the channel (n+1) output is not controlled by FTM.

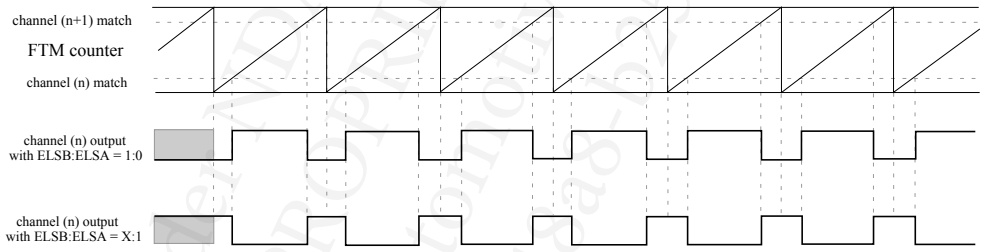


Figure 274. Modified Combine PWM Mode

The Modified Combine PWM mode allows the offset addition of the duty cyle, thus, in some cases, the C(n+1)V match can happen on the next FTM counter period. For CNTIN ≥ 0, MOD > 0, and CNTIN < MOD, this situation happens when C(n)V > C(n+1)V.

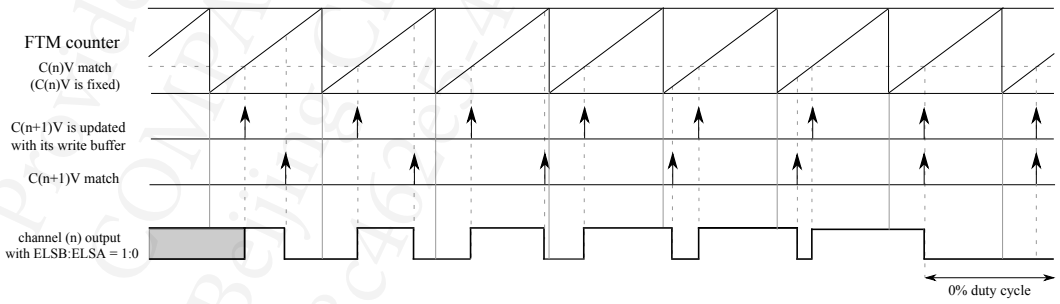


Figure 275. Modified Combine PWM Mode Examples

If more than one pair of channels are configured in Modified Combine PWM Mode, it is possible to fix an offset for the channel (n) match edge of each pair with respect to other pairs. This behavior is useful in the generation of lighting PWM control signals where it is desirable that edges are not coincident with each other pair to help eliminate noise generation. The C(n)V register value is the shift of the PWM pulse with respect to the beginning of FTM counter period (FTM counter = CNTIN).

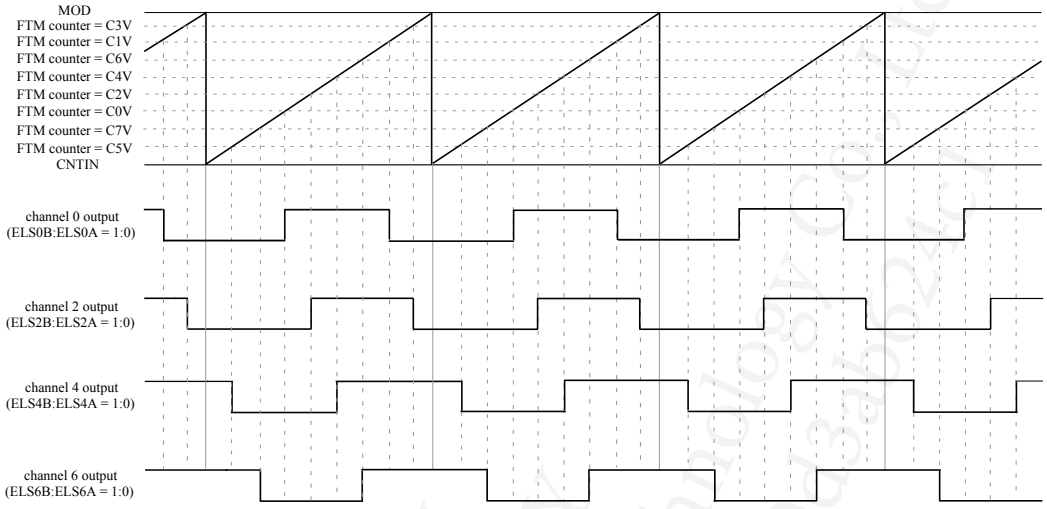


Figure 276. Example of Four Pairs of Channels in Modified Combine PWM Mode

43.5.10.1 Synchronization

In the Modified Combine Mode, the following registers should be updated when the FTM counter clock is disabled (CLKS[1:0] = 0:0).

- CNTIN (CNTIN register update)
- MOD (MOD and HCR registers update)
- C(n)V and C(n+1)V (CnV register update)

In the Modified Combine Mode, if (FTMEN = 1), (CLKS[1:0] ≠ 0:0), and there was a write to the register C(n+1)V, then the register C(n+1)V is updated with its write buffer value on the next channel (n) match (FTM counter = C(n)V). This feature allows to vary the PWM duty cycle value in this mode.

NOTE

In the Modified Combine Mode, the bit SYNCEN(n) should be zero bit for the channels (n) and (n+1). So, the following features are not available for this mode.

- C(n)V and C(n+1)V register synchronization
- Reload Points
- Global Load

43.5.11 Complementary Mode

The Complementary mode is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMP = 1

In Complementary mode, the channel (n+1) output is the inverse of the channel (n) output.

NOTE

The Complementary Mode is not available on Output Compare mode.

The channel (n+1) output is the same as the channel (n) output when:

- QUADEN = 0
- DECAPEN = 0
- COMP = 0
- channels (n) and (n+1) are on Combine Mode or Modified Combine PWM Mode

The channel (n+1) output is independent from channel (n) output when:

- QUADEN = 0
- DECAPEN = 0
- COMP = 0
- channel (n) is on Output Compare Mode, EPWM or CPWM

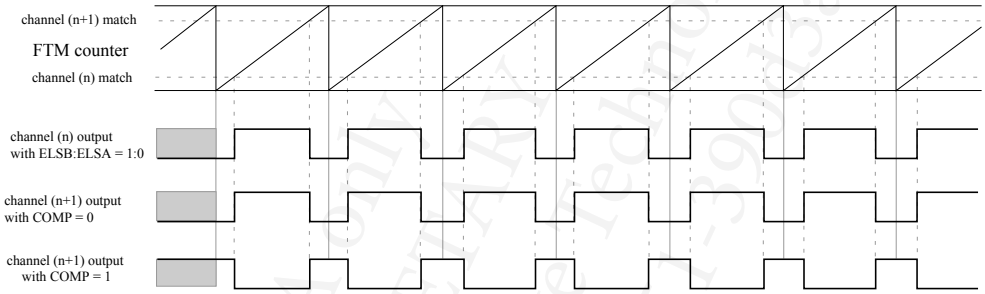


Figure 277. Channel (n+1) output in Complementary mode with (ELSB:ELSA = 1:0)

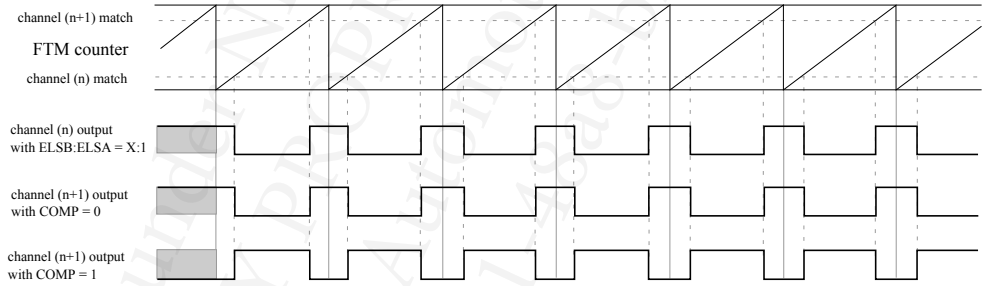


Figure 278. Channel (n+1) output in Complementary mode with (ELSB:ELSA = X:1)

43.5.12 Registers updated from write buffers

43.5.12.1 CNTIN register update

The following table describes when CNTIN register is updated:

Table 273. CNTIN register update

When	Then CNTIN register is updated
CLKS[1:0] = 0:0	When CNTIN register is written, independent of FTMEN bit.
• FTMEN = 0, or	At the next FTM input clock after CNTIN was written.

Table continues on the next page...

Table 273. CNTIN register update (continued)

When	Then CNTIN register is updated
• CNTINC = 0	
• FTMEN = 1, • SYNCMODE = 1, and • CNTINC = 1	By the CNTIN register synchronization.
• CNTINC = 1, and • LDOK = 1	By the Reload Points.

43.5.12.2 MOD and HCR registers update

The following table describes when MOD or HCR registers are updated:

Table 274. MOD and HCR updates

When	Then MOD or HCR is updated
CLKS[1:0] = 0:0	When MOD (or HCR) is written, independent of FTMEN bit.
• CLKS[1:0] ≠ 0:0, and • FTMEN = 0	According to the CPWMS bit, that is: • If the selected mode is not CPWM then MOD (or HCR) is updated after MOD (or HCR) register was written and the FTM counter changes from MOD to CNTIN. If the FTM counter is at free-running counter mode then this update occurs when the FTM counter changes from 0xFFFF to 0x0000. • If the selected mode is CPWM then MOD (or HCR) register is updated after MOD (or HCR) register was written and the FTM counter changes from MOD to (MOD – 0x0001).
• CLKS[1:0] ≠ 0:0, and • FTMEN = 1	By the MOD register synchronization. HCR follows the same procedure of MOD register in this case.
• LDOK = 1	By the Reload Points.

43.5.12.3 CnV register update

The following table describes when CnV register is updated:

Table 275. CnV register update

When	Then CnV register is updated
CLKS[1:0] = 0:0	When CnV register is written, independent of FTMEN bit.
• CLKS[1:0] ≠ 0:0, and • FTMEN = 0	According to the selected mode, that is:

Table continues on the next page...

Table 275. CnV register update (continued)

When	Then CnV register is updated
	• If the selected mode is Output Compare, then CnV register is updated on the next FTM counter change, end of the prescaler counting, after CnV register was written. • If the selected mode is EPWM, then CnV register is updated after CnV register was written and the FTM counter changes from MOD to CNTIN. If the FTM counter is at free-running counter mode then this update occurs when the FTM counter changes from 0xFFFF to 0x0000. • If the selected mode is CPWM, then CnV register is updated after CnV register was written and the FTM counter changes from MOD to (MOD – 0x0001).
• CLKS[1:0] ≠ 0:0, and • FTMEN = 1	According to the selected mode, that is: • If the selected mode is output compare then CnV register is updated according to the SYNCEN bit. If (SYNCEN = 0) then CnV register is updated after CnV register was written at the next change of the FTM counter, the end of the prescaler counting. If (SYNCEN = 1) then CnV register is updated by the C(n)V and C(n+1)V register synchronization. • If the selected mode is not output compare and (SYNCEN = 1) then CnV register is updated by the C(n)V and C(n+1)V register synchronization.
• SYNCEN = 1, and • LDOK = 1	By the Reload Points.

43.5.13 PWM synchronization

The PWM synchronization provides an opportunity to update the MOD, HCR, CNTIN, CnV, OUTMASK, INVCTRL and SWOCTRL registers with their buffered value and force the FTM counter to the CNTIN register value.

NOTE

The legacy PWM synchronization (SYNCMODE = 0) is a subset of the enhanced PWM synchronization (SYNCMODE = 1). Thus, only the enhanced PWM synchronization must be used.

43.5.13.1 Hardware trigger

Three hardware trigger signal inputs of the FTM module are enabled when TRIGn = 1, where n = 0, 1 or 2 corresponding to each one of the input signals, respectively. The hardware trigger input n is synchronized by the FTM input clock. The PWM synchronization with hardware trigger is initiated when a rising edge is detected at the enabled hardware trigger inputs.

If (HWTRIGMODE = 0) then the TRIGn bit is cleared when 0 is written to it or when the trigger n event is detected.

In this case, if two or more hardware triggers are enabled (for example, TRIG0 and TRIG1 = 1) and only trigger 1 event occurs, then only TRIG1 bit is cleared. If a trigger n event occurs together with a write setting TRIGn bit, then the synchronization is initiated, but TRIGn bit remains set due to the write operation.

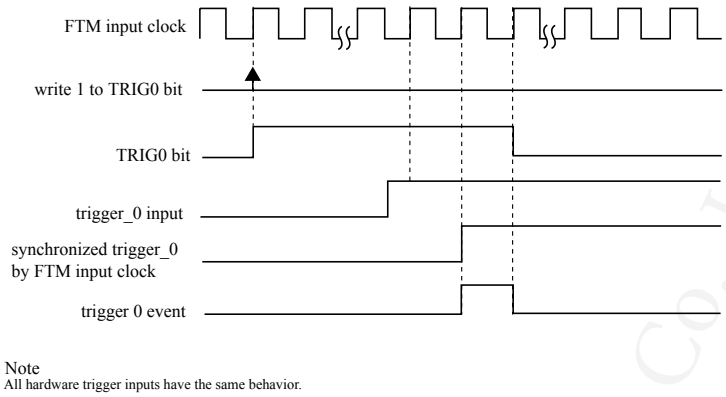


Figure 279. Hardware trigger event with HWTRIGMODE = 0

If HWTRIGMODE = 1, then the TRIGn bit is only cleared when 0 is written to it.

NOTE

The HWTRIGMODE bit must be 1 only with enhanced PWM synchronization (SYNCMODE = 1).

43.5.13.2 Software trigger

A software trigger event occurs when 1 is written to the SYNC[SWSYNC] bit. The SWSYNC bit is cleared when 0 is written to it or when the PWM synchronization, initiated by the software event, is completed.

If another software trigger event occurs (by writing another 1 to the SWSYNC bit) at the same time the PWM synchronization initiated by the previous software trigger event is ending, a new PWM synchronization is started and the SWSYNC bit remains equal to 1.

If SYNCMODE = 0 then the SWSYNC bit is also cleared by FTM according to PWMSYNC and REINIT bits. In this case if (PWMSYNC = 1) or (PWMSYNC = 0 and REINIT = 0) then SWSYNC bit is cleared at the next selected loading point after that the software trigger event occurred; see Synchronization Points and the following figure. If (PWMSYNC = 0) and (REINIT = 1) then SWSYNC bit is cleared when the software trigger event occurs.

If SYNCMODE = 1 then the SWSYNC bit is also cleared by FTM according to the SWRSTCNT bit. If SWRSTCNT = 0 then SWSYNC bit is cleared at the next selected loading point after that the software trigger event occurred; see the following figure. If SWRSTCNT = 1 then SWSYNC bit is cleared when the software trigger event occurs.

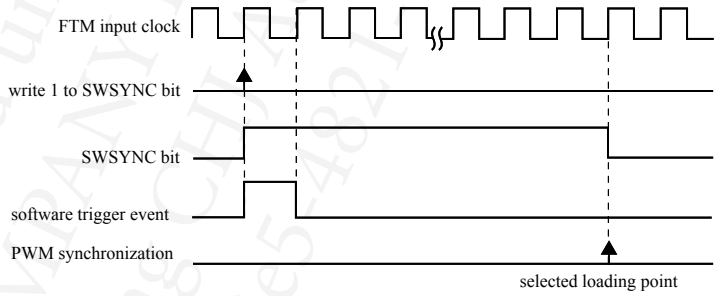


Figure 280. Software trigger event

43.5.13.3 Synchronization Points

The synchronization points are points where the registers can be updated with their write buffer by PWM synchronization. These synchronization points are safe points because guarantee smooth transitions in the generated PWM signals.

In Up counting, the synchronization points are when the FTM counter changes from MOD to CNTIN. In this case, the synchronization points are enabled if (CNTMIN = 1) or (CNTMAX = 1).

In Up-down counting, the synchronization points are:

- if (CNTMAX = 1), when the FTM counter changes from (MOD) to (MOD - 1);
- if (CNTMIN = 1), when the FTM counter changes from (CNTIN) to (CNTIN + 1).

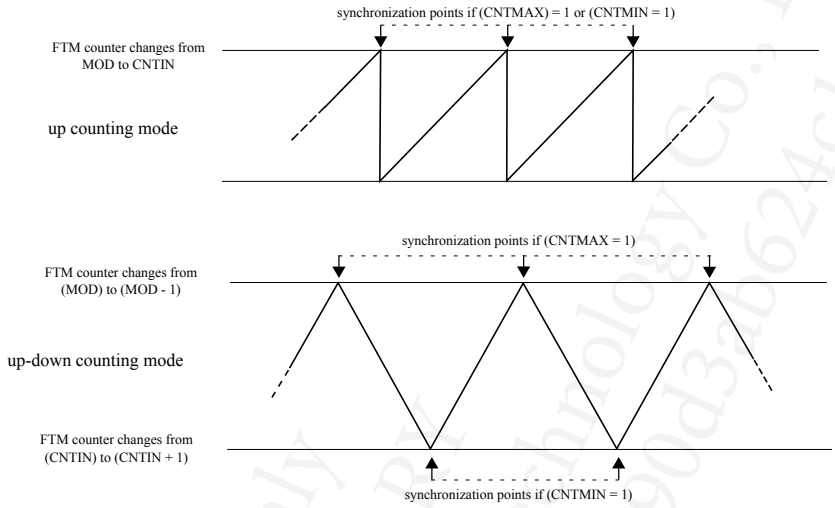


Figure 281. Synchronization Points

43.5.13.4 MOD register synchronization

The MOD register synchronization updates the MOD register with its buffer value. This synchronization is enabled if (FTMEN = 1).

The MOD register synchronization can be done by either the enhanced PWM synchronization (SYNCMODE = 1) or the legacy PWM synchronization (SYNCMODE = 0). However, it is expected that the MOD register be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the MOD register synchronization depends on SWWRBUF, SWRSTCNT, HWWRBUF, and HWRSTCNT bits according to this flowchart:

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-9251-390d3a624c1

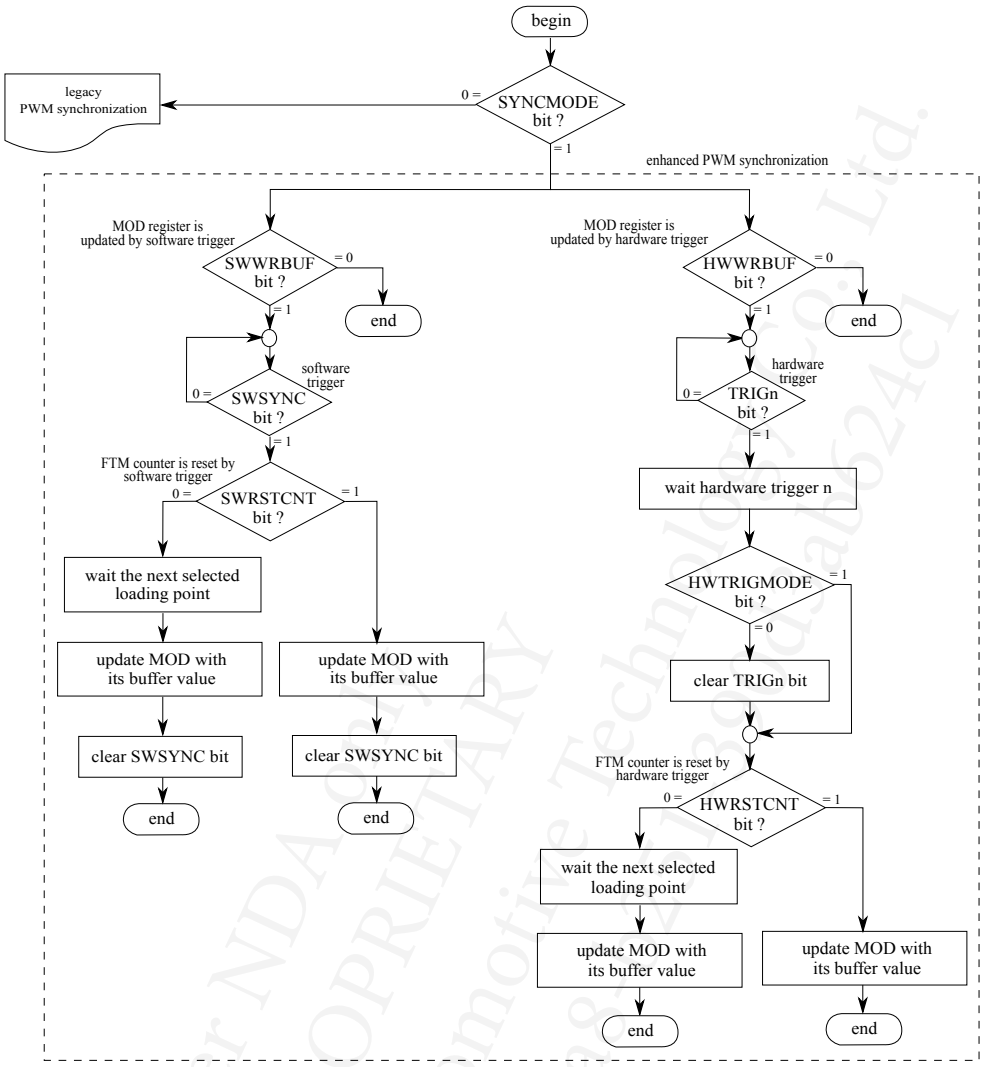


Figure 282. MOD register synchronization flowchart

In the case of legacy PWM synchronization, the MOD register synchronization depends on PWMSYNC and REINIT bits according to the following description.

If (SYNCMODE = 0), (PWMSYNC = 0), and (REINIT = 0), then this synchronization is made on the next selected loading point after an enabled trigger event takes place. If the trigger event was a software trigger, then the SWSYNC bit is cleared on the next selected loading point. If the trigger event was a hardware trigger, then the trigger enable bit (TRIGn) is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

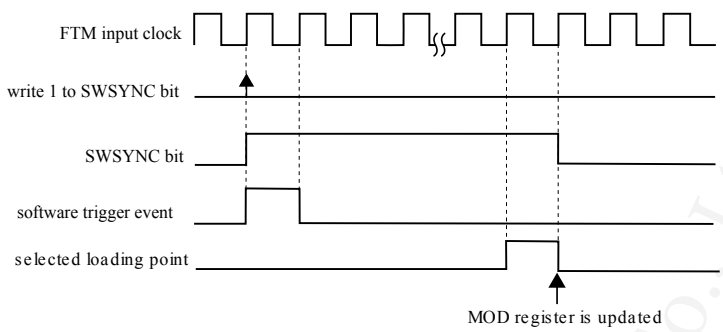


Figure 283. MOD synchronization with (SYNCMODE = 0), (PWMSYNC = 0), (REINIT = 0), and software trigger was used

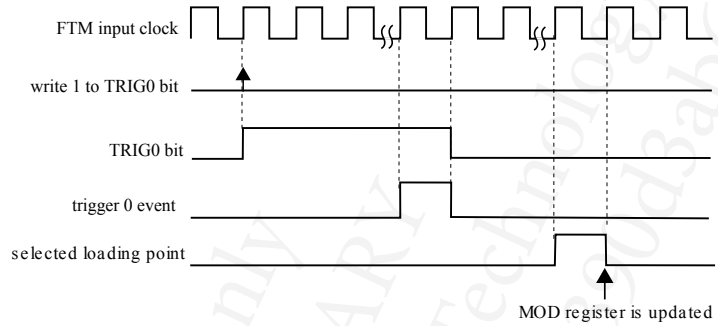


Figure 284. MOD synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (PWMSYNC = 0), (REINIT = 0), and a hardware trigger was used

If (SYNCMODE = 0), (PWMSYNC = 0), and (REINIT = 1), then this synchronization is made on the next enabled trigger event. If the trigger event was a software trigger, then the SWSYNC bit is cleared according to the following example. If the trigger event was a hardware trigger, then the TRIGn bit is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

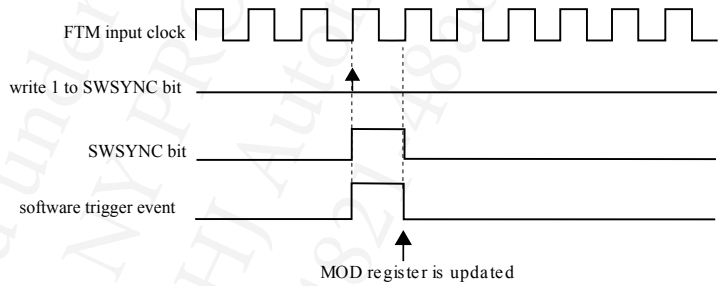


Figure 285. MOD synchronization with (SYNCMODE = 0), (PWMSYNC = 0), (REINIT = 1), and software trigger was used

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4827-4884-a513-50d3ab624c1

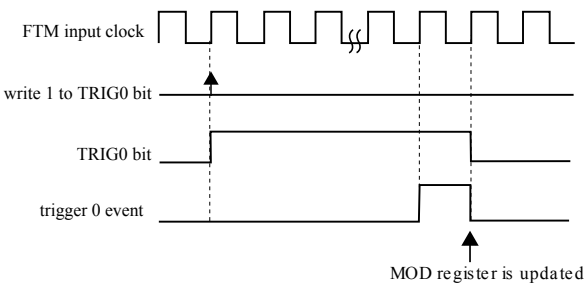


Figure 286. MOD synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (PWMSYNC = 0), (REINIT = 1), and a hardware trigger was used

If (SYNCMODE = 0) and (PWMSYNC = 1), then this synchronization is made on the next selected loading point after the software trigger event takes place. The SWSYNC bit is cleared on the next selected loading point:

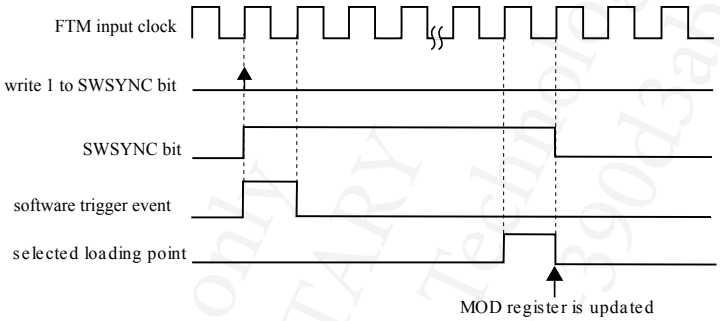


Figure 287. MOD synchronization with (SYNCMODE = 0) and (PWMSYNC = 1)

43.5.13.5 CNTIN register synchronization

The CNTIN register synchronization updates the CNTIN register with its buffer value.

This synchronization is enabled if (FTMEN = 1), (SYNCMODE = 1), and (CNTINC = 1). The CNTIN register synchronization can be done only by the enhanced PWM synchronization (SYNCMODE = 1). The synchronization mechanism is the same as the MOD register synchronization done by the enhanced PWM synchronization; see MOD register synchronization.

43.5.13.6 C(n)V and C(n+1)V register synchronization

The C(n)V and C(n+1)V registers synchronization updates the C(n)V and C(n+1)V registers with their buffer values.

This synchronization is enabled if (FTMEN = 1) and (SYNCEN = 1). The synchronization mechanism is the same as the MOD register synchronization. However, it is expected that the C(n)V and C(n+1)V registers be synchronized only by the enhanced PWM synchronization (SYNCMODE = 1).

43.5.13.7 OUTMASK register synchronization

The OUTMASK register synchronization updates the OUTMASK register with its buffer value.

The OUTMASK register can be updated at each rising edge of FTM input clock (SYNCHOM = 0), by the enhanced PWM synchronization (SYNCHOM = 1 and SYNCMODE = 1) or by the legacy PWM synchronization (SYNCHOM = 1 and SYNCMODE = 0). However, it is expected that the OUTMASK register be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the OUTMASK register synchronization depends on SWOM and HWOM bits. See the following flowchart:

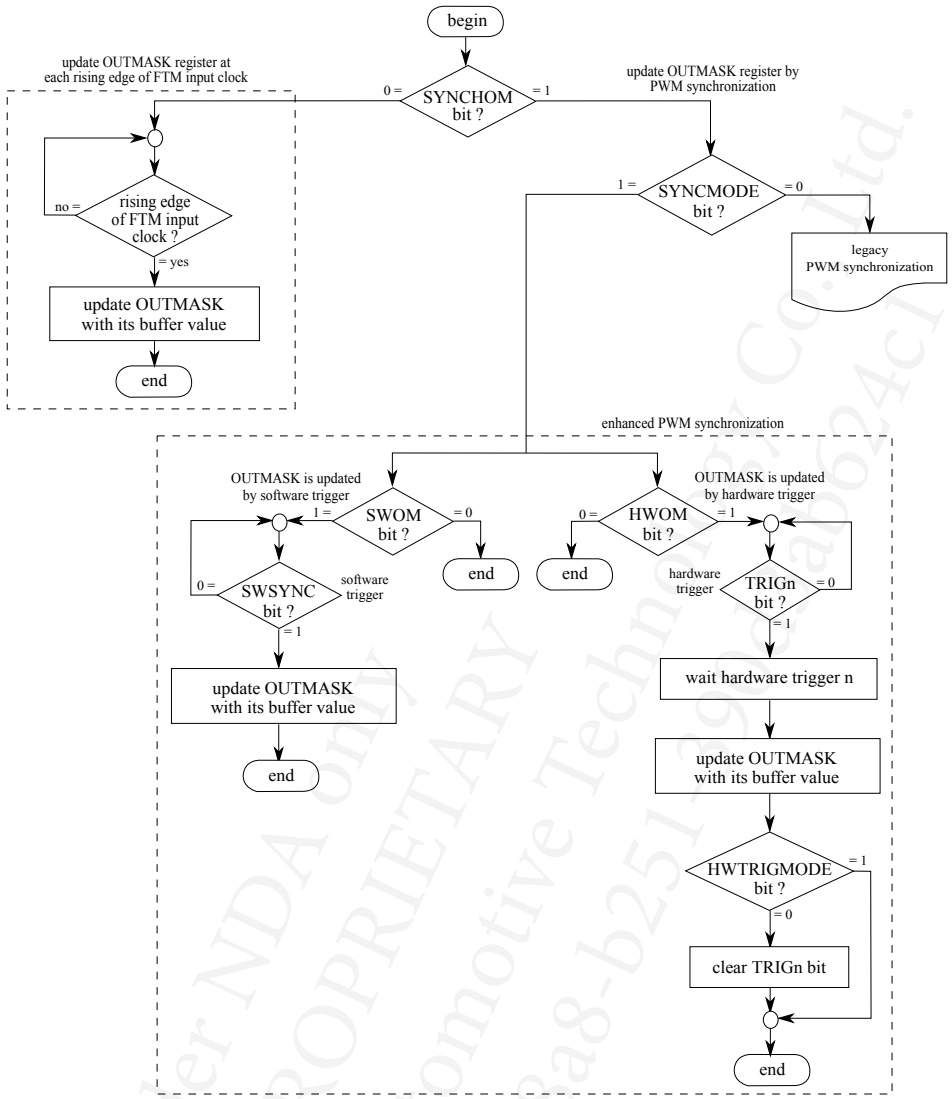


Figure 288. OUTMASK register synchronization flowchart

In the case of legacy PWM synchronization, the OUTMASK register synchronization depends on PWMSYNC bit according to the following description.

If (SYNCMODE = 0), (SYNCHOM = 1), and (PWMSYNC = 0), then this synchronization is done on the next enabled trigger event. If the trigger event was a software trigger, then the SWSYNC bit is cleared on the next selected loading point. If the trigger event was a hardware trigger, then the TRIGn bit is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CCI Automotive Technology Co., Ltd.
c8c462e5-021a-48a8-b251-990eab624c1

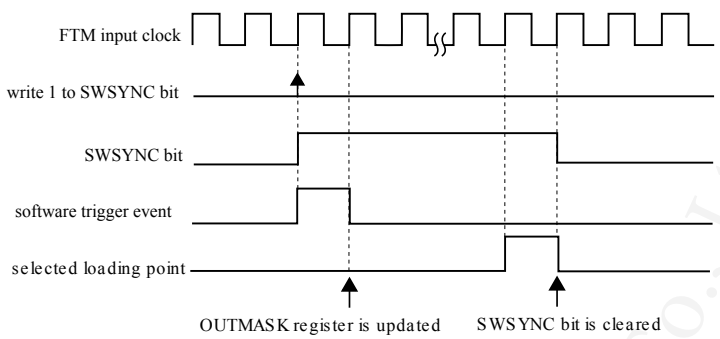


Figure 289. OUTMASK synchronization with (SYNCMODE = 0), (SYNCHOM = 1), (PWMSYNC = 0) and software trigger was used

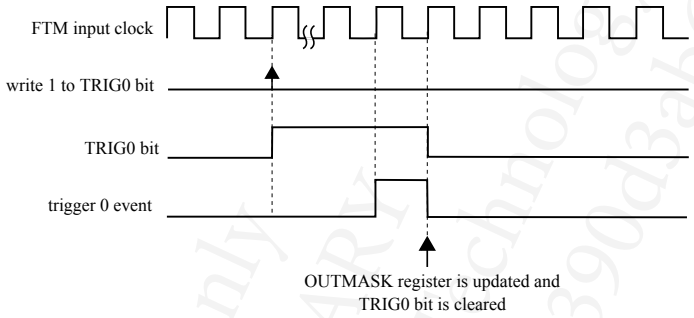


Figure 290. OUTMASK synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (SYNCHOM = 1), (PWMSYNC = 0), and a hardware trigger was used

If (SYNCMODE = 0), (SYNCHOM = 1), and (PWMSYNC = 1), then this synchronization is made on the next enabled hardware trigger. The TRIGn bit is cleared according to Hardware trigger. An example with a hardware trigger follows.

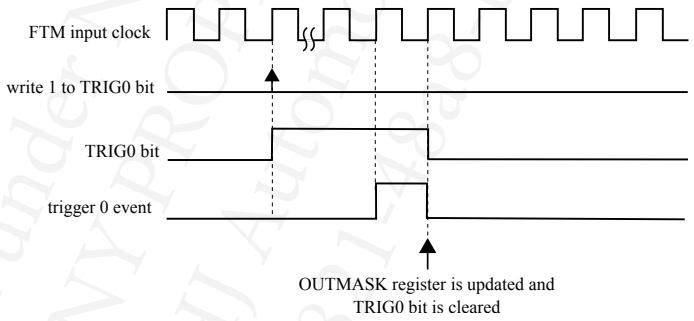


Figure 291. OUTMASK synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (SYNCHOM = 1), (PWMSYNC = 1), and a hardware trigger was used

43.5.13.8 INVCTRL register synchronization

The INVCTRL register synchronization updates the INVCTRL register with its buffer value.

The INVCTRL register can be updated at each rising edge of FTM input clock (INVC = 0) or by the enhanced PWM synchronization (INVC = 1 and SYNCMODE = 1) according to the following flowchart.

In the case of enhanced PWM synchronization, the INVCTRL register synchronization depends on SWINVC and HWINVC bits.

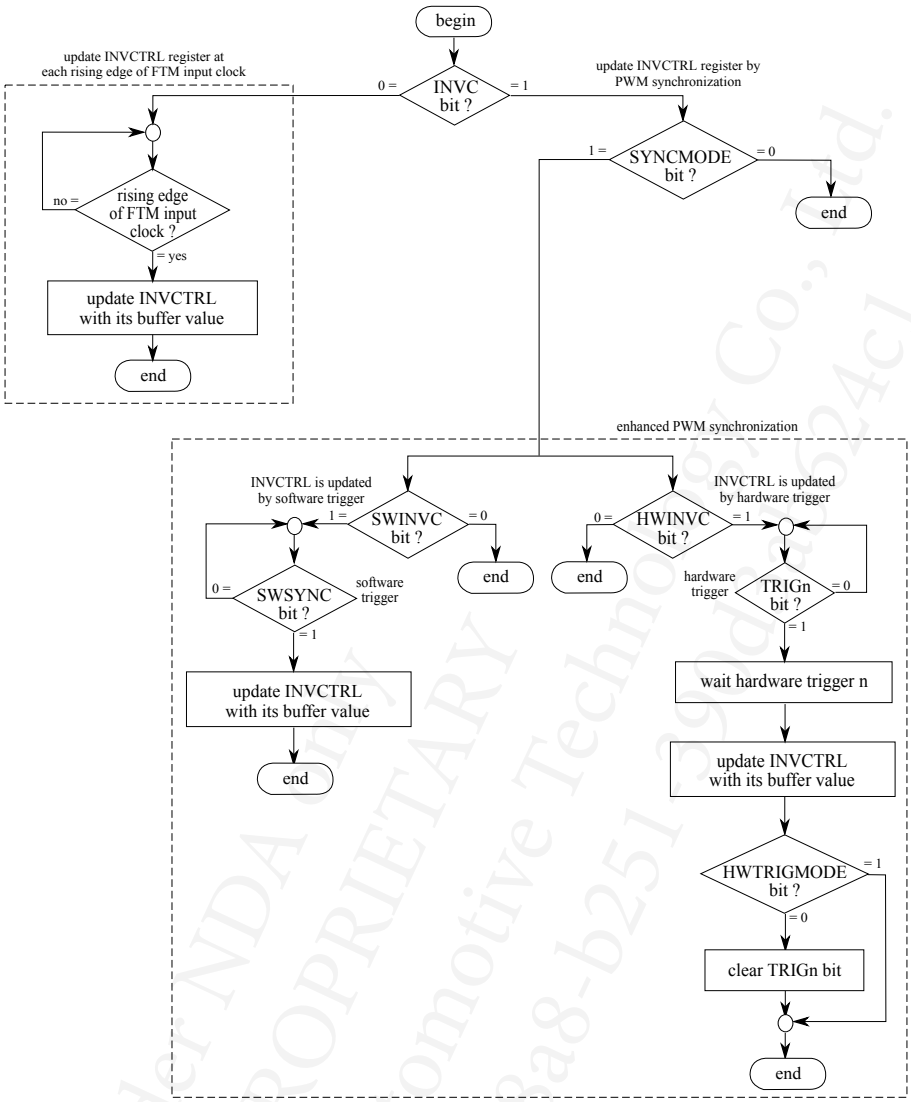


Figure 292. INVCTRL register synchronization flowchart

43.5.13.9 SWOCTRL register synchronization

The SWOCTRL register synchronization updates the SWOCTRL register with its buffer value.

The SWOCTRL register can be updated at each rising edge of FTM input clock (SWOC = 0) or by the enhanced PWM synchronization (SWOC = 1 and SYNCMODE = 1) according to the following flowchart.

In the case of enhanced PWM synchronization, the SWOCTRL register synchronization depends on SWSOC and HWSOC bits.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CAN Automotive Technology Co., Ltd.
c8c462e3-921-48a8-b251-29dd1624c1

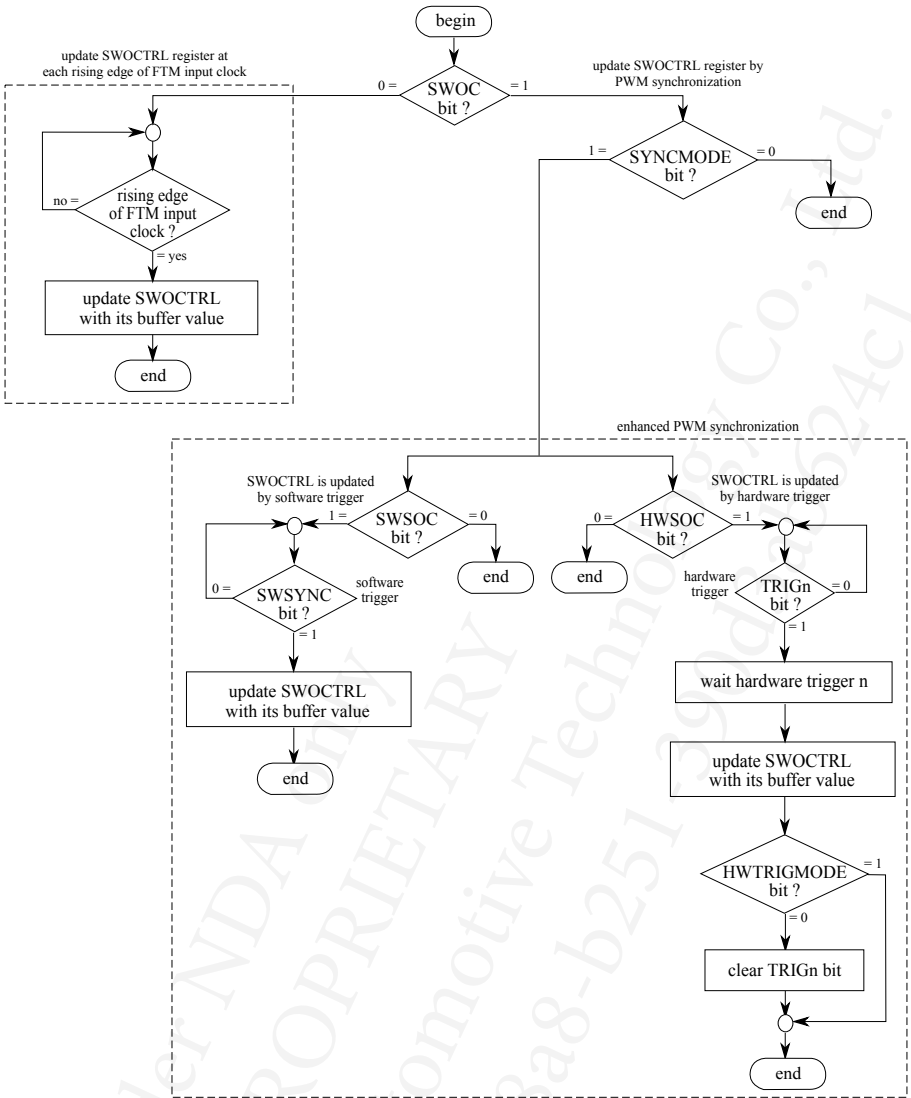


Figure 293. SWOCTRL register synchronization flowchart

43.5.13.10 FTM counter synchronization

The FTM counter synchronization is a mechanism that allows the FTM to restart the PWM generation at a certain point in the PWM period. The channels outputs are forced to their initial value, except for channels in Output Compare mode, and the FTM counter is forced to its initial counting value defined by CNTIN register.

The following figure shows the FTM counter synchronization. Note that after the synchronization event occurs, the channel (n) is set to its initial value and the channel (n+1) is not set to its initial value due to a specific timing of this figure in which the deadtime insertion prevents this channel output from transitioning to 1. If no deadtime insertion is selected, then the channel (n+1) transitions to logical value 1 immediately after the synchronization event occurs.

Preliminary NDA Only
COMPANY PROPRIETARY
Beijing CT Automotive Technology Co., Ltd.
c8c46a21-48a8-b251-1908-6924c1

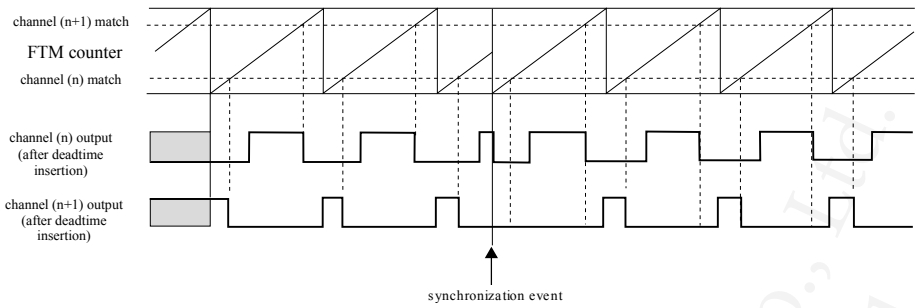


Figure 294. FTM counter synchronization

The FTM counter synchronization can be done by either the enhanced PWM synchronization (SYNCMODE = 1) or the legacy PWM synchronization (SYNCMODE = 0). However, the FTM counter must be synchronized only by the enhanced PWM synchronization.

In the case of enhanced PWM synchronization, the FTM counter synchronization depends on SWRSTCNT and HWRSTCNT bits according to the following flowchart.

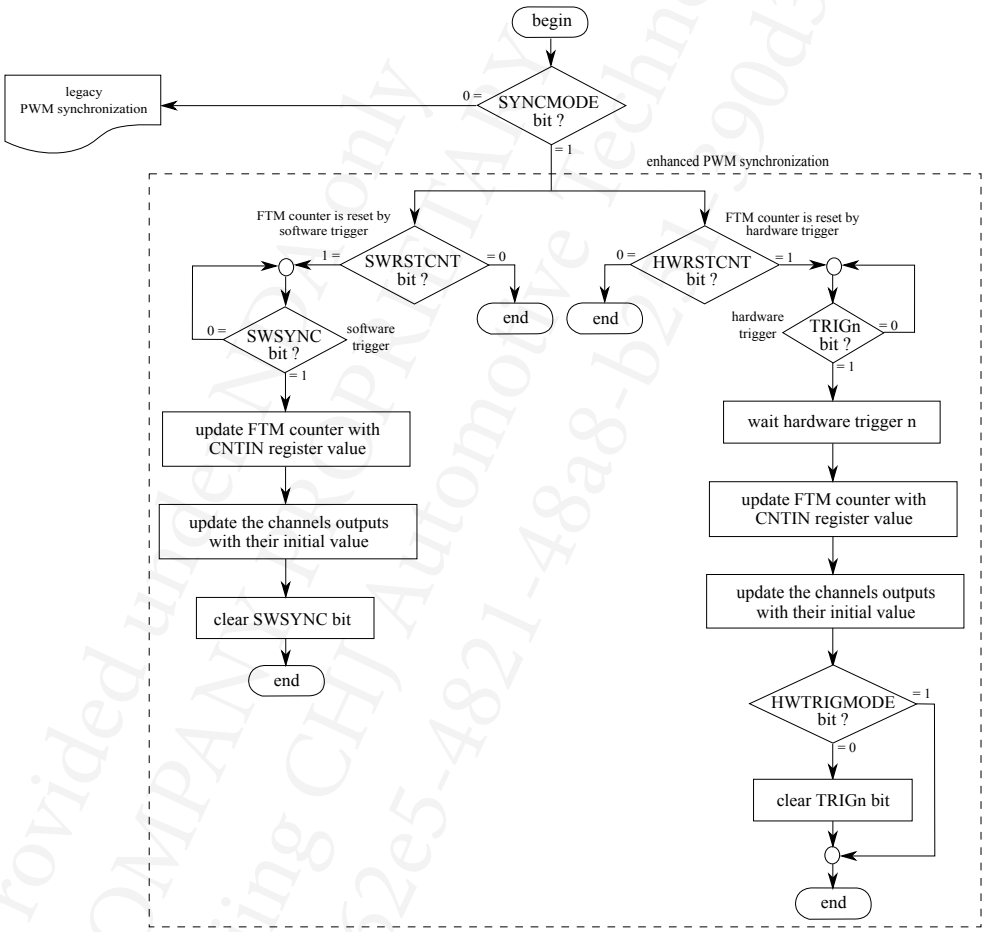


Figure 295. FTM counter synchronization flowchart

In the case of legacy PWM synchronization, the FTM counter synchronization depends on REINIT and PWMSYNC bits according to the following description.

If (SYNCMODE = 0), (REINIT = 1), and (PWMSYNC = 0) then this synchronization is made on the next enabled trigger event. If the trigger event was a software trigger then the SWSYNC bit is cleared according to the following example. If the trigger

event was a hardware trigger then the TRIGn bit is cleared according to Hardware trigger. Examples with software and hardware triggers follow.

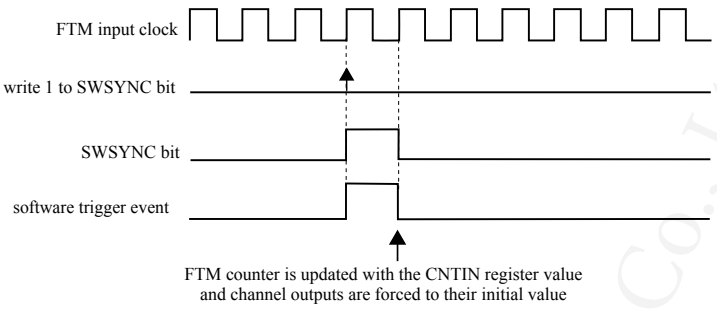


Figure 296. FTM counter synchronization with (SYNCMODE = 0), (REINIT = 1), (PWMSYNC = 0), and software trigger was used

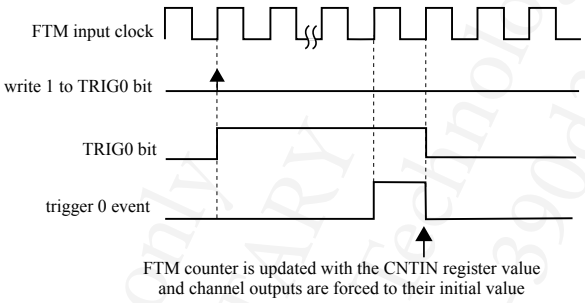


Figure 297. FTM counter synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (REINIT = 1), (PWMSYNC = 0), and a hardware trigger was used

If (SYNCMODE = 0), (REINIT = 1), and (PWMSYNC = 1) then this synchronization is made on the next enabled hardware trigger. The TRIGn bit is cleared according to Hardware trigger.

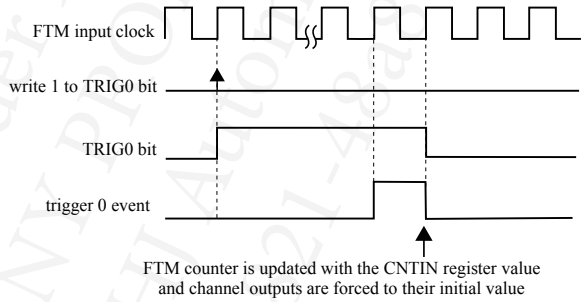


Figure 298. FTM counter synchronization with (SYNCMODE = 0), (HWTRIGMODE = 0), (REINIT = 1), (PWMSYNC = 1), and a hardware trigger was used

43.5.14 Inverting

The invert functionality swaps the signals between channel (n) and channel (n+1) outputs. The inverting operation is selected when:

- QUADEN = 0
- DECAPEN = 0
- COMP = 1, and
- INVm = 1 (where m represents a channel pair)

The INVm bit in INVCTRL register is updated with its buffer value according to INVCTRL register synchronization.

In combine mode with channel (n) ELSB:ELSA = 1:0, the channel (n) output is forced low at the beginning of the period (FTM counter = CNTIN), forced high at the channel (n) match and forced low at the channel (n+1) match. If the inverting is selected, the channel (n) output behavior is changed to force high at the beginning of the PWM period, force low at the channel (n) match and force high at the channel (n+1) match. See the following figure.

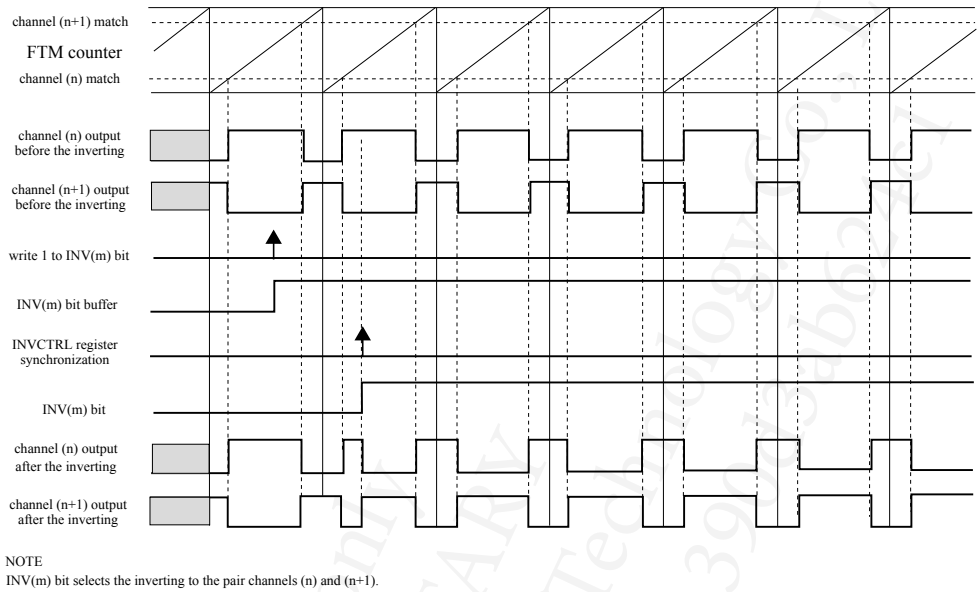


Figure 299. Channels (n) and (n+1) outputs after the inverting in combine mode with channel (n) ELSB:ELSA = 1:0

Note that the channel (n) ELSB:ELSA bits should be considered because they define the active state of the channels outputs. In combine mode with channel (n) ELSB:ELSA = X:1, the channel (n) output is forced high at the beginning of the period, forced low at the channel (n) match and forced high at the channel (n+1) match. When inverting is selected, the channels (n) and (n+1) present waveforms as shown in the following figure.

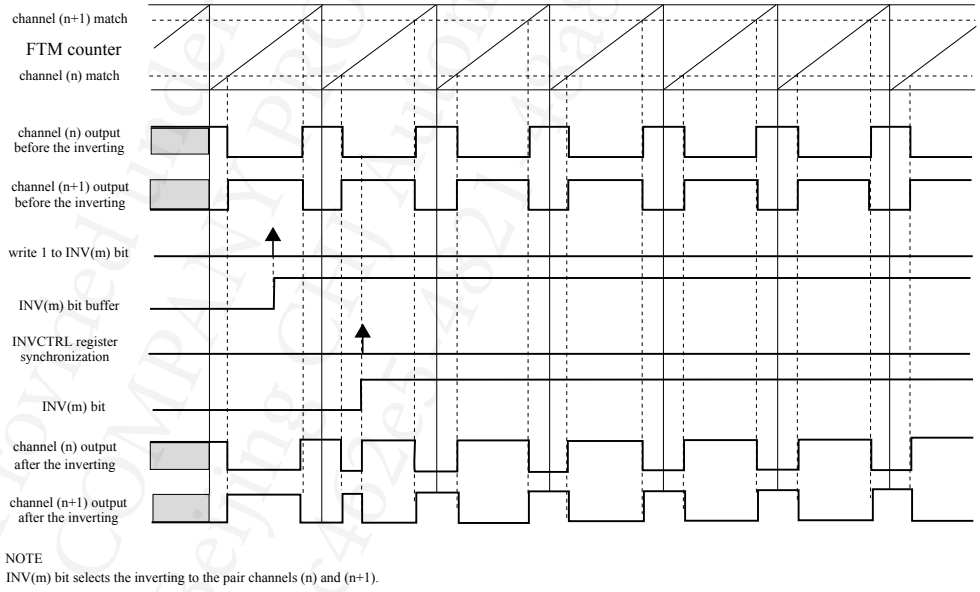


Figure 300. Channels (n) and (n+1) outputs after the inverting in combine mode with channel (n) ELSB:ELSA = X:1

NOTE

The Inverting is not available in Output Compare mode and Modified Combine PWM Mode.

43.5.15 Software Output Control Mode

The software output control forces the channel output according to software defined values at a specific time in the PWM generation.

The software output control is selected when:

- QUADEN = 0
- DECAPEN = 0, and
- CH(n)OC = 1

The CH(n)OC bit enables the software output control for a specific channel output and the CH(n)OCV selects the value that is forced to this channel output.

Both CH(n)OC and CH(n)OCV bits in SWOCTRL register are buffered and updated with their buffer value according to SWOCTRL register synchronization.

The following figure shows the channels (n) and (n+1) outputs signals when the software output control is used. In this case the channels (n) and (n+1) are set to Combine and Complementary mode.

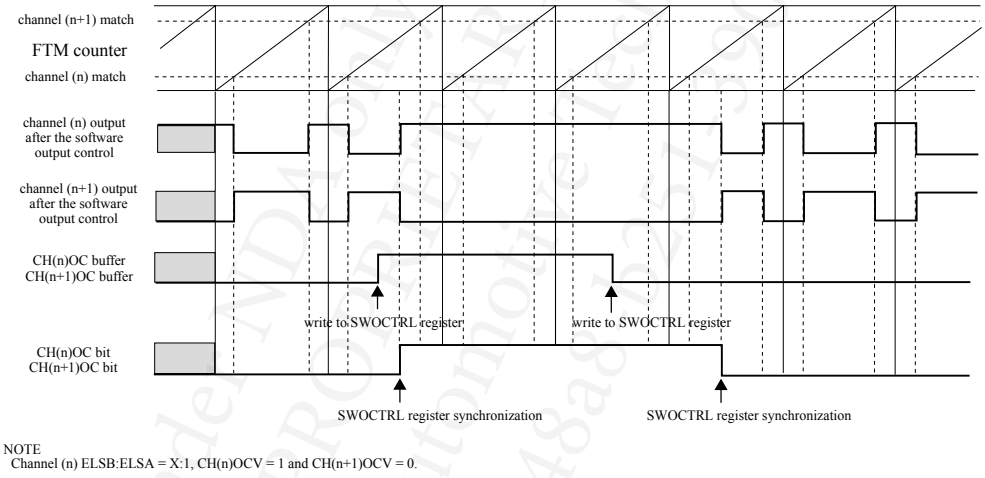


Figure 301. Example of software output control in Combine and Complementary mode

Software output control forces the following values on channels (n) and (n+1) when the COMP bit is zero.

Table 276. Software ouput control behavior when (COMP = 0)

CH(n)OC	CH(n+1)OC	CH(n)OCV	CH(n+1)OCV	Channel (n) Output	Channel (n+1) Output
0	0	X	X	is not modified by SWOC	is not modified by SWOC
1	1	0	0	is forced to zero	is forced to zero
1	1	0	1	is forced to zero	is forced to one
1	1	1	0	is forced to one	is forced to zero
1	1	1	1	is forced to one	is forced to one

Software output control forces the following values on channels (n) and (n+1) when the COMP bit is one.

Table 277. Software ouput control behavior when (COMP = 1)

CH(n)OC	CH(n+1)OC	CH(n)OCV	CH(n+1)OCV	Channel (n) Output	Channel (n+1) Output
0	0	X	X	is not modified by SWOC	is not modified by SWOC
1	1	0	0	is forced to zero	is forced to zero
1	1	0	1	is forced to zero	is forced to one
1	1	1	0	is forced to one	is forced to zero
1	1	1	1	is forced to one	is forced to zero

NOTE

- The CH(n)OC and CH(n+1)OC bits should be equal.
- The COMP bit must not be modified when software output control is enabled, that is, CH(n)OC = 1 and/or CH(n+1)OC = 1.
- Software output control has the same behavior with disabled or enabled FTM counter (see the CLKS field description in the Status and Control register).

43.5.16 Deadtime insertion

The deadtime insertion is enabled when DTEN is set and the concatenation {DTVALEX[3:0], DTVAL[5:0]} is non-zero.

DEADTIME register defines the deadtime delay that can be used for all FTM channels. The clock for the DEADTIME delay is the FTM input clock divided by DTPS bits, and the {DTVALEX[3:0], DTVAL[5:0]} bits define the deadtime modulo, that is, the number of the deadtime prescaler clocks.

The deadtime delay insertion ensures that no two complementary signals (channels (n) and (n+1)) drive the active state at the same time.

If POL(n) = 0, POL(n+1) = 0, and the deadtime is enabled, then when the channel (n) match (FTM counter = C(n)V) occurs, the channel (n) output remains at the low value until the end of the deadtime delay when the channel (n) output is set. Similarly, when the channel (n+1) match (FTM counter = C(n+1)V) occurs, the channel (n+1) output remains at the low value until the end of the deadtime delay when the channel (n+1) output is set. See the following figures.

If POL(n) = 1, POL(n+1) = 1, and the deadtime is enabled, then when the channel (n) match (FTM counter = C(n)V) occurs, the channel (n) output remains at the high value until the end of the deadtime delay when the channel (n) output is cleared. Similarly, when the channel (n+1) match (FTM counter = C(n+1)V) occurs, the channel (n+1) output remains at the high value until the end of the deadtime delay when the channel (n+1) output is cleared.

Provided by IHS Markit
COMPANY CONFIDENTIAL
Beijing CEC Computer Technology Co., Ltd.
c8c462e5-4c1f-4e8b-8b25-390d3ab624c1

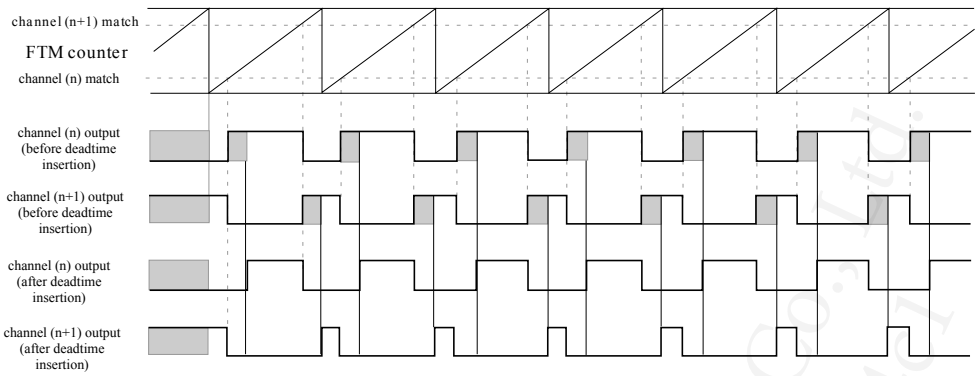


Figure 302. Deadtime insertion with ELSB:ELSA = 1:0, POL(n) = 0, and POL(n+1) = 0

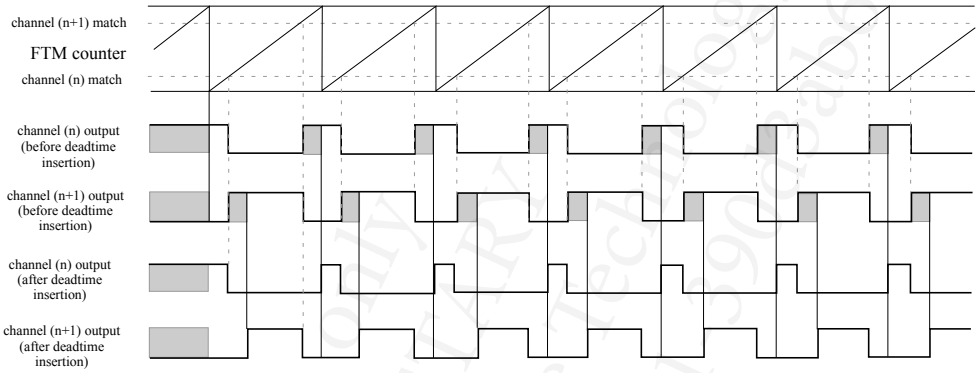


Figure 303. Deadtime insertion with ELSB:ELSA = X:1, POL(n) = 0, and POL(n+1) = 0

NOTE

- The deadtime feature must be used only in Complementary mode.
- The deadtime feature is not available in Output Compare mode.

43.5.16.1 Deadtime insertion corner cases

If (PS[2:0] is cleared), (DTPS[1:0] = 0:0 or DTPS[1:0] = 0:1):

- and the deadtime delay is greater than or equal to the channel (n) duty cycle ((C(n+1)V – C(n)V) × FTM input clock), then the channel (n) output is always the inactive value (POL(n) bit value).
- and the deadtime delay is greater than or equal to the channel (n+1) duty cycle ((MOD – CNTIN + 1 – (C(n+1)V – C(n)V)) × FTM input clock), then the channel (n+1) output is always the inactive value (POL(n+1) bit value).

Although, in most cases the deadtime delay is not comparable to channels (n) and (n+1) duty cycle, the following figures show examples where the deadtime delay is comparable to the duty cycle.

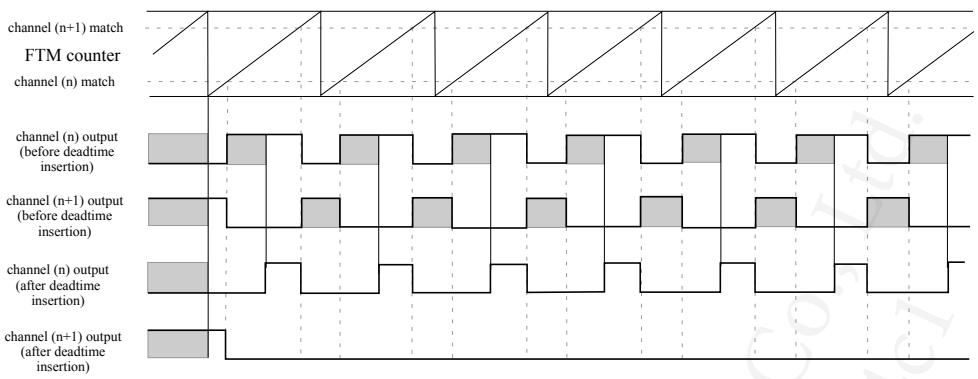


Figure 304. Example of the deadtime insertion (channel (n) ELSB:ELSA = 1:0, POL(n) = 0, and POL(n+1) = 0) when the deadtime delay is comparable to channel (n+1) duty cycle

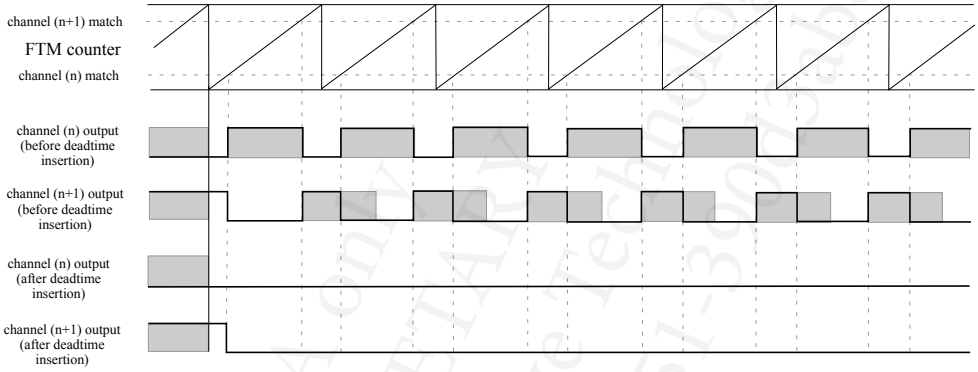


Figure 305. Example of the deadtime insertion (channel (n) ELSB:ELSA = 1:0, POL(n) = 0, and POL(n+1) = 0) when the deadtime delay is comparable to channels (n) and (n+1) duty cycle

43.5.17 Output mask

The output mask can be used to force channels output to their inactive state through software. For example: to control a BLDC motor.

Any write to the OUTMASK register updates its write buffer. The OUTMASK register is updated with its buffer value by PWM synchronization; see OUTMASK register synchronization.

If CH(n)OM = 1, then the channel (n) output is forced to its inactive state (POLn bit value). If CH(n)OM = 0, then the channel (n) output is unaffected by the output mask. See the following figure.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4888-b151-3800346246c1

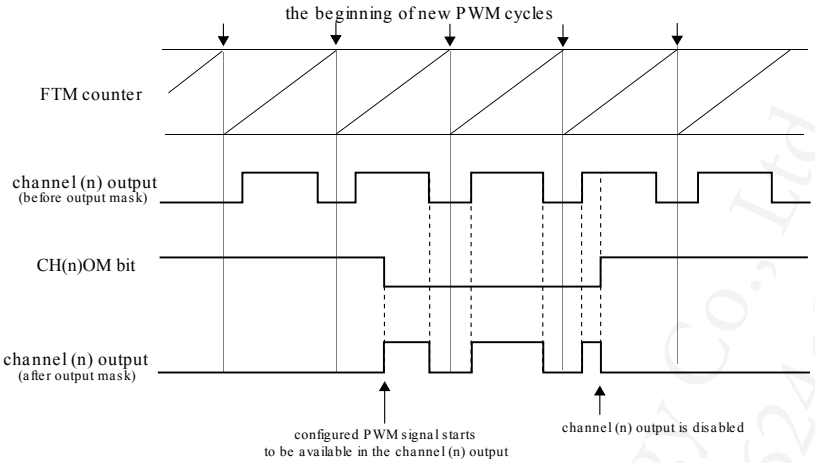


Figure 306. Output mask with POLn = 0

The following table shows the output mask result before the polarity control.

Table 278. Output mask result for channel (n) before the polarity control

CH(n)OM	Output Mask Input	Output Mask Result
0	inactive state	inactive state
	active state	active state
1	inactive state	inactive state
	active state	

43.5.18 Polarity Control

The POLn bit selects the channel (n) output polarity:

- If POLn = 0, the channel (n) output polarity is high, so the logical one is the active state and the logical zero is the inactive state.
- If POLn = 1, the channel (n) output polarity is low, so the logical zero is the active state and the logical one is the inactive state.

43.5.19 Initialization

The initialization forces the CH(n)OI bit value to the channel (n) output when 1 is written to the INIT bit.

The initialization depends on COMP and DTEN bits. The following table shows the values that channels (n) and (n+1) are forced by initialization when the COMP and DTEN bits are zero.

Table 279. Initialization behavior when (COMP = 0 and DTEN = 0)

CH(n)OI	CH(n+1)OI	Channel (n) Output	Channel (n+1) Output
0	0	is forced to zero	is forced to zero
0	1	is forced to zero	is forced to one
1	0	is forced to one	is forced to zero
1	1	is forced to one	is forced to one

The following table shows the values that channels (n) and (n+1) are forced by initialization when (COMP = 1) or (DTEN = 1).

Table 280. Initialization behavior when (COMP = 1 or DTEN = 1)

CH(n)OI	CH(n+1)OI	Channel (n) Output	Channel (n+1) Output
0	X	is forced to zero	is forced to one
1	X	is forced to one	is forced to zero

NOTE

The initialization feature must be used only with disabled FTM counter. See the description of the CLKS field in the Status and Control register.

43.5.20 Features Priority

The following figure shows the priority of the features used at the generation of channels (n) and (n+1) outputs signals.

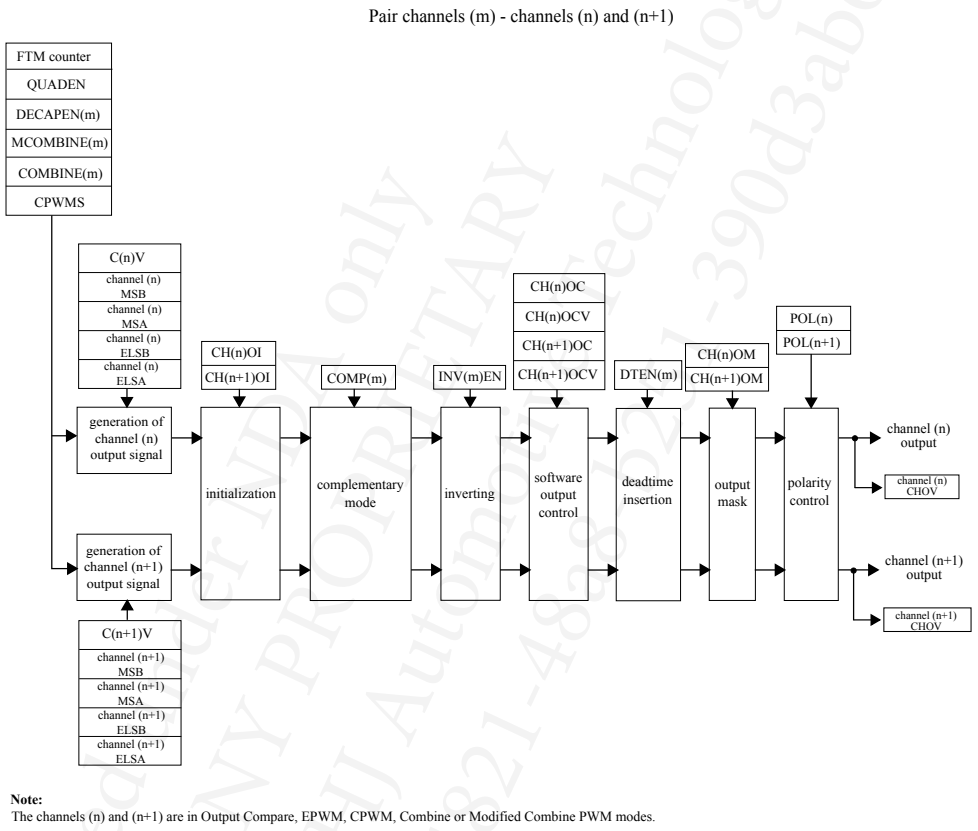


Figure 307. Priority of the features usead at the generation of channels (n) and (n+1) output

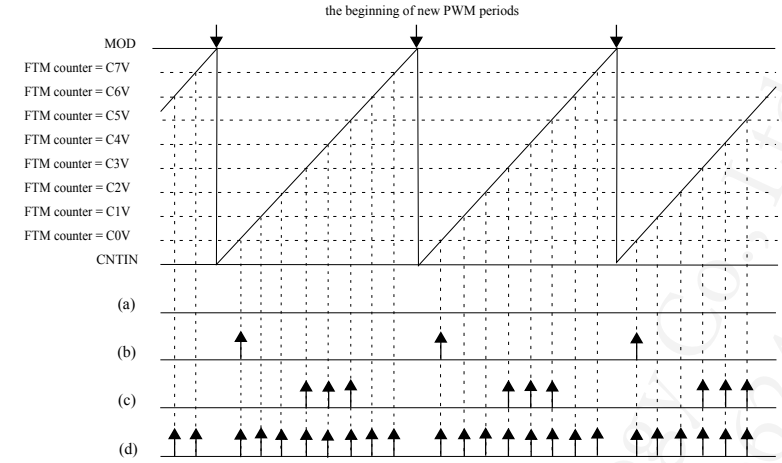
NOTE

The Initialization must not be used with Inverting and Software Output Control Mode.

43.5.21 External Trigger

If the CH(n)TRIG bit (register EXTTRIG) is set, where n = 0, 1, 2, 3, 4, 5, 6 or 7, then the FTM generates a trigger when the channel (n) match occurs (FTM counter = C(n)V) at the FTM external trigger output.

The width of a channel (n) trigger is one FTM input clock and the FTM is able to generate multiple triggers in one PWM period. See the figure below.



Note
(a) CH0TRIG = 0, CH1TRIG = 0, CH2TRIG = 0, CH3TRIG = 0, CH4TRIG = 0, CH5TRIG = 0, CH6TRIG = 0, CH7TRIG = 0
(b) CH0TRIG = 1, CH1TRIG = 0, CH2TRIG = 0, CH3TRIG = 0, CH4TRIG = 0, CH5TRIG = 0, CH6TRIG = 0, CH7TRIG = 0
(c) CH0TRIG = 0, CH1TRIG = 0, CH2TRIG = 0, CH3TRIG = 1, CH4TRIG = 1, CH5TRIG = 1, CH6TRIG = 0, CH7TRIG = 0
(d) CH0TRIG = 1, CH1TRIG = 1, CH2TRIG = 1, CH3TRIG = 1, CH4TRIG = 1, CH5TRIG = 1, CH6TRIG = 1, CH7TRIG = 1

Figure 308. External Trigger

43.5.22 Initialization Trigger

Initialization trigger allows FTM to generate a trigger in some specific points of FTM counter cycle. This feature is controlled by the bits INITTRIGEN and ITRIGR. The INITTRIGEN bit enables the initialization trigger generation and the ITRIGR bit selects when the initialization trigger is generated.

If INITTRIGEN = 1 and ITRIGR = 1, then the initialization trigger is generated when FTM counter reaches a reload point according to the frequency of the reload opportunities (Reload Points).

NOTE

For this configuration of initilization trigger and in CPWM mode, the bits CNTMAX and CNTMIN select where the initialization trigger is generated.

If INITTRIGEN = 1 and ITRIGR = 0, then the initialization trigger is generated when FTM counter is updated with the CNTIN register value. See the cases below.

1. When FTM counter is updated with CNTIN register value automatically.

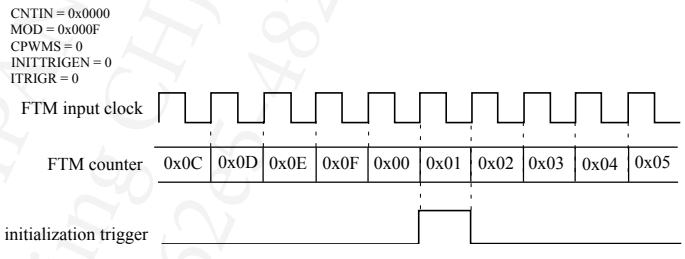


Figure 309. Example of the generation of the initialization trigger in the case 1.

2. When there is a write to CNT register.

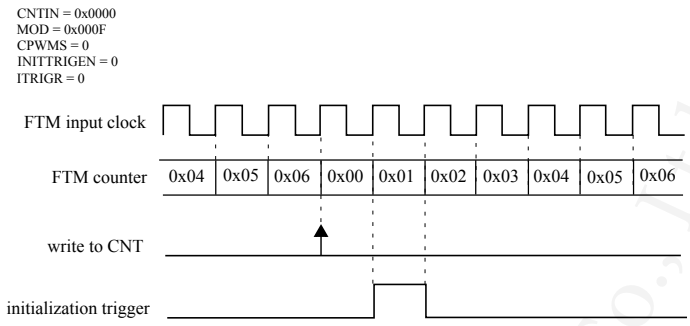


Figure 310. Example of the generation of the initialization trigger in the case 2.

NOTE

This behavior is not available in CPWM mode.

3. When there is the FTM counter synchronization.

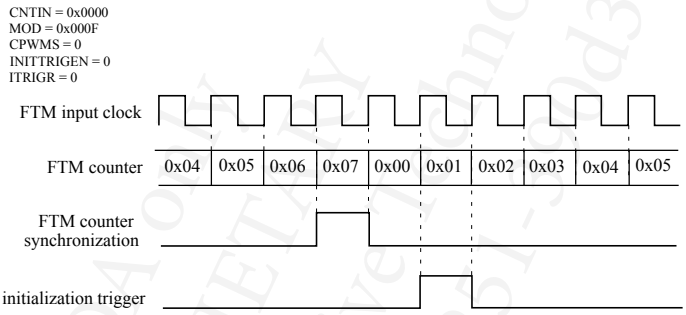


Figure 311. Example of the generation of the initialization trigger in the case 3.

NOTE

This behavior is not available in CPWM mode.

4. If (CNT = CNTIN), (CLKS[1:0] = 0:0), and a value different from zero is written to CLKS[1:0] bits.

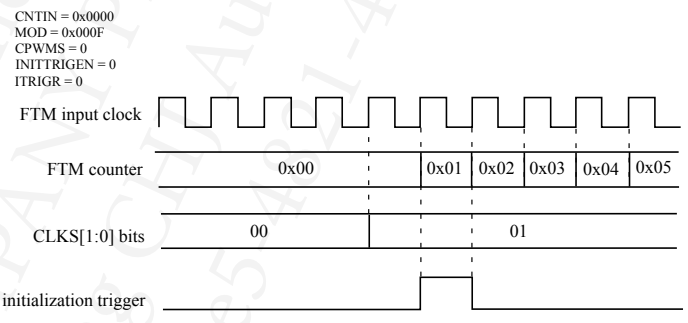


Figure 312. Example of the generation of the initialization trigger in the case 4.

NOTE

This behavior is not available in CPWM mode.

5. If the channel (n) is in Input Capture mode, (ICRST = 1) and the selected input capture event occurs in the channel (n) input.

Provided under NDA of COMPANY PROPRIETARY
Beijing CIP Automotive Technology Co., Ltd.
c8462854-21-4ca2-b251-3ad3ab624c1

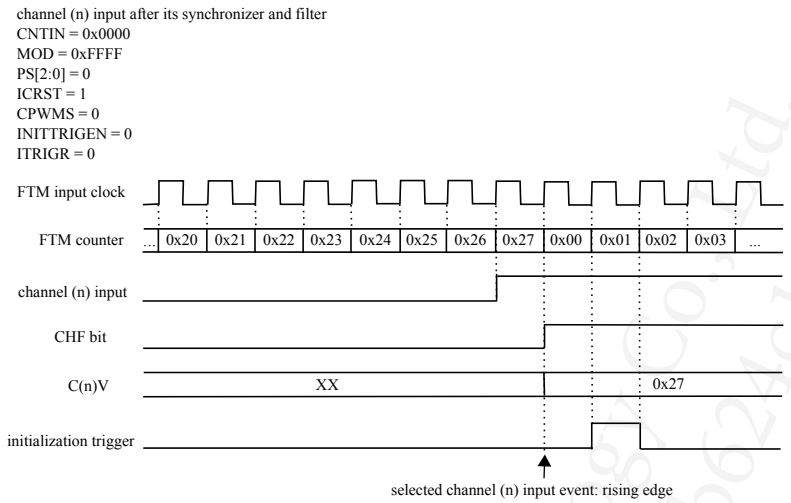


Figure 313. Example of the generation of the initialization trigger in the case 5.

43.5.23 Capture Test Mode

The Capture Test mode allows to test the CnV registers, the FTM counter and the interconnection logic between the FTM counter and CnV registers.

In this test mode, all channels must be configured for Input Capture Mode and FTM counter must be configured to the Up counting.

When the Capture Test mode is enabled (CAPTEST = 1), the FTM counter is frozen and any write to CNT register updates directly the FTM counter; see the following figure. After it was written, all CnV registers are updated with the written value to CNT register and CHF bits are set. Therefore, the FTM counter is updated with its next value according to its configuration. Its next value depends on CNTIN, MOD, and the written value to FTM counter.

The next reads of CnV registers return the written value to the FTM counter and the next reads of CNT register return FTM counter next value.

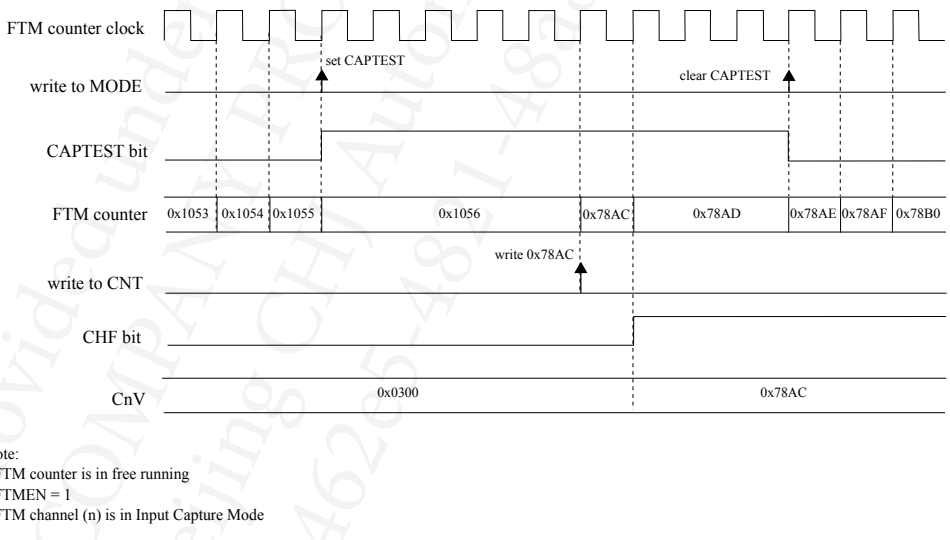


Figure 314. Capture Test Mode

43.5.24 DMA

The channel generates a DMA transfer request according to DMA and CHIE bits. See the following table.

Table 281. Channel DMA transfer request

DMA	CHIE	Channel DMA Transfer Request	Channel Interrupt
0	0	The channel DMA transfer request is not generated.	The channel interrupt is not generated.
0	1	The channel DMA transfer request is not generated.	The channel interrupt is generated if (CHF = 1).
1	0	The channel DMA transfer request is not generated.	The channel interrupt is not generated.
1	1	The channel DMA transfer request is generated if (CHF = 1).	The channel interrupt is not generated.

If DMA = 1, the CHF bit is cleared either by channel DMA transfer done or reading CnSC while CHF is set and then writing a zero to CHF bit according to CHIE bit. See the following table.

Table 282. Clear CHF bit when DMA = 1

CHIE	How CHF Bit Can Be Cleared
0	CHF bit is cleared either when the channel DMA transfer is done or by reading CnSC while CHF is set and then writing a 0 to CHF bit.
1	CHF bit is cleared when the channel DMA transfer is done.

43.5.25 Dual Edge Capture Mode

The dual edge capture mode is enabled if DECAPEN = 1. This mode allows to measure a pulse width or period of the channel (n) input where n = 0, 2, 4 or 6. The channel (n) filter can be enabled for n = 0 or 2.

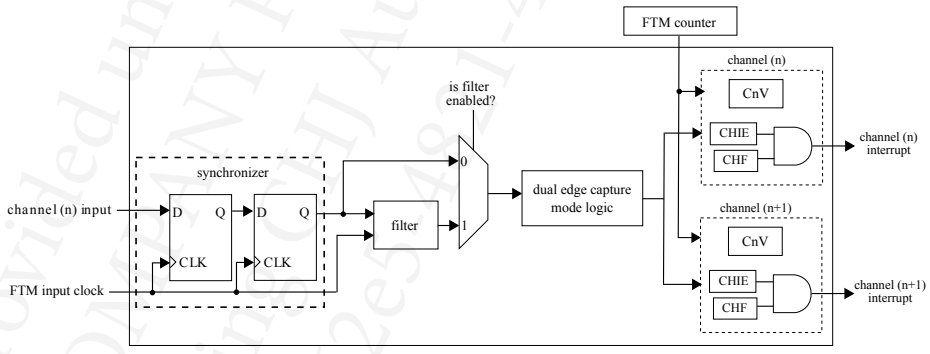


Figure 315. Diagram for Dual Edge Capture Mode when FLTPS[3:0] = 0

provided under NDA only
COMPANY PROPRIETARY
Beijing eGTJ Automotive Technology Co., Ltd.
c8c62e54821-4848-b25f-590c3a624d

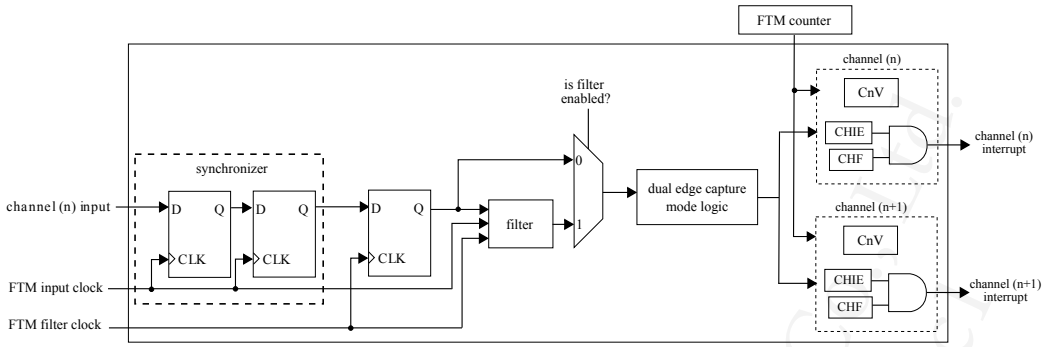


Figure 316. Diagram for Dual Edge Capture Mode when FLTPS[3:0] ≠ 0

The channel (n) MSA bit defines if the dual edge capture mode is one-shot or continuous.

The channel (n) ELSB:ELSA bits select the edge that is captured by channel (n), and channel (n+1) ELSB:ELSA bits select the edge that is captured by channel (n+1). If both channel (n) ELSB:ELSA and channel (n+1) ELSB:ELSA bits select the same edge, then it is the period measurement. If these bits select different edges, then it is a pulse width measurement.

In the dual edge capture mode, only channel (n) input is used and channel (n+1) input is ignored.

If the selected edge by channel (n) bits is detected at channel (n) input, then channel (n) CHF bit is set and the channel (n) interrupt is generated (if channel (n) CHIE = 1). If the selected edge by channel (n+1) bits is detected at channel (n) input and (channel (n) CHF = 1), then channel (n+1) CHF bit is set and the channel (n+1) interrupt is generated (if channel (n+1) CHIE = 1).

The C(n)V register stores the FTM counter value when the selected edge by channel (n) is detected at channel (n) input. The C(n+1)V register stores the FTM counter value when the selected edge by channel (n+1) is detected at channel (n) input.

In this mode, a coherency mechanism (for channels (n) and (n+1)) ensures coherent data when the C(n)V and C(n+1)V registers are read. The only requirement is that C(n)V must be read before C(n+1)V.

NOTE

- The dual edge capture mode must be used with channel (n) ELSB:ELSA = 0:1 or 1:0, channel (n+1) ELSB:ELSA = 0:1 or 1:0 and the FTM counter in Free running counter.

43.5.25.1 One-Shot Capture mode

The One-Shot Capture mode is selected when (DECAPEN = 1), and (channel (n) MSA = 0). In this capture mode, only one pair of edges at the channel (n) input is captured. The channel (n) ELSB:ELSA bits select the first edge to be captured, and channel (n+1) ELSB:ELSA bits select the second edge to be captured.

The edge captures are enabled while DECAP bit is set. For each new measurement in One-Shot Capture mode, first the channel (n) CHF and channel (n+1) CHF bits must be cleared, and then the DECAP bit must be set.

In this mode, the DECAP bit is automatically cleared by FTM when the edge selected by channel (n+1) is captured. Therefore, while DECAP bit is set, the one-shot capture is in process. When this bit is cleared, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers.

Similarly, when the channel (n+1) CHF bit is set, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers.

43.5.25.2 Continuous Capture mode

The Continuous Capture mode is selected when (DECAPEN = 1), and (channel (n) MSA = 1). In this capture mode, the edges at the channel (n) input are captured continuously. The channel (n) ELSB:ELSA bits select the initial edge to be captured, and channel (n+1) ELSB:ELSA bits select the final edge to be captured.

The edge captures are enabled while DECAP bit is set. For the initial use, first the channel (n) CHF and channel (n+1) CHF bits must be cleared, and then DECAP bit must be set to start the continuous measurements.

When the channel (n+1) CHF bit is set, both edges were captured and the captured values are ready for reading in the C(n)V and C(n+1)V registers. The latest captured values are always available in these registers even after the DECAP bit is cleared.

In this mode, it is possible to clear only the channel (n+1) CHF bit. Therefore, when the channel (n+1) CHF bit is set again, the latest captured values are available in C(n)V and C(n+1)V registers.

For a new sequence of the measurements in the Dual Edge Capture – Continuous mode, clear the channel (n) CHF and channel (n+1) CHF bits to start new measurements.

43.5.25.3 Pulse width measurement

If the channel (n) is configured to capture rising edges (channel (n) ELSB:ELSA = 0:1) and the channel (n+1) to capture falling edges (channel (n+1) ELSB:ELSA = 1:0), then the positive polarity pulse width is measured. If the channel (n) is configured to capture falling edges (channel (n) ELSB:ELSA = 1:0) and the channel (n+1) to capture rising edges (channel (n+1) ELSB:ELSA = 0:1), then the negative polarity pulse width is measured.

The pulse width measurement can be made in One-Shot Capture mode or Continuous Capture mode.

The following figure shows an example of the Dual Edge Capture – One-Shot mode used to measure the positive polarity pulse width. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. The DECAP bit is set to enable the measurement of next positive polarity pulse width. The channel (n) CHF bit is set when the first edge of this pulse is detected, that is, the edge selected by channel (n) ELSB:ELSA bits. The channel (n+1) CHF bit is set and DECAP bit is cleared when the second edge of this pulse is detected, that is, the edge selected by channel (n+1) ELSB:ELSA bits. Both DECAP and channel (n+1) CHF bits indicate when two edges of the pulse were captured and the C(n)V and C(n+1)V registers are ready for reading.

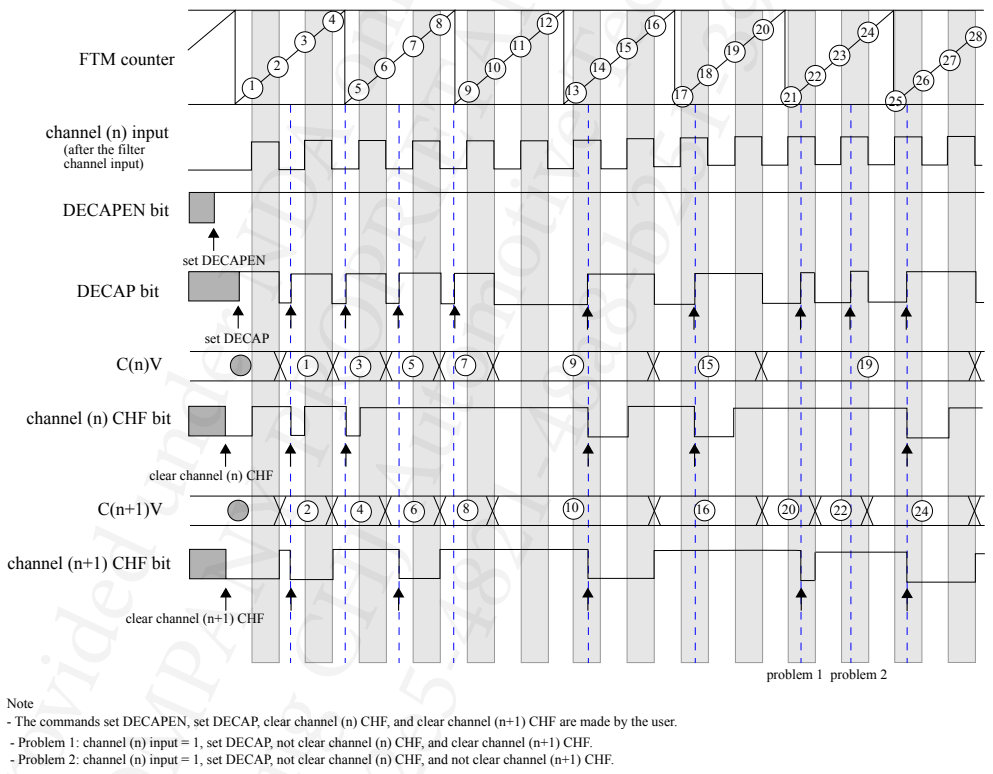


Figure 317. Dual Edge Capture – One-Shot mode for positive polarity pulse width measurement

The following figure shows an example of the Dual Edge Capture – Continuous mode used to measure the positive polarity pulse width. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. While the DECAP bit is set the configured measurements are made. The channel (n) CHF bit is set when the first edge of the positive polarity pulse is detected, that is, the edge selected by channel (n) ELSB:ELSA bits. The channel (n+1) CHF bit is set when the second edge of this pulse is detected,

that is, the edge selected by channel (n+1) ELSB:ELSA bits. The channel (n+1) CHF bit indicates when two edges of the pulse were captured and the C(n)V and C(n+1)V registers are ready for reading.

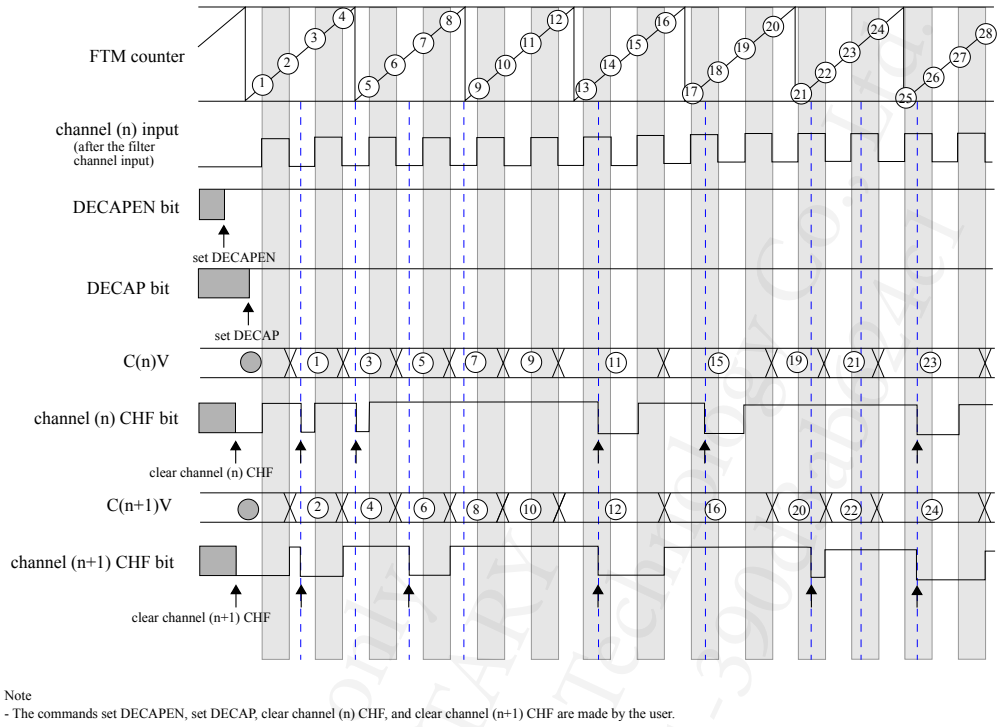


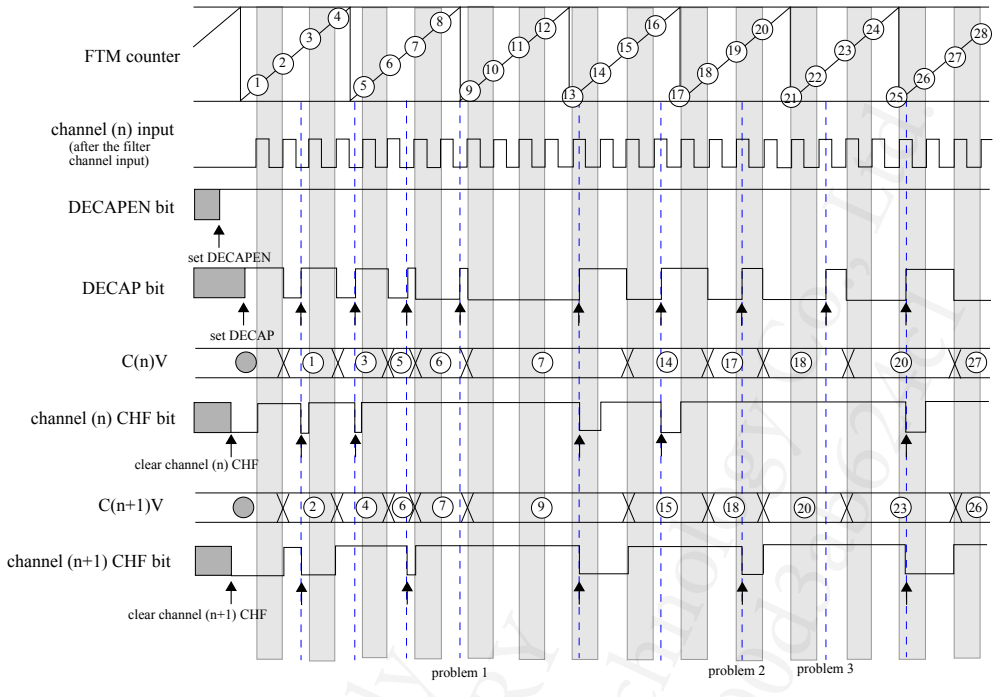
Figure 318. Dual Edge Capture – Continuous mode for positive polarity pulse width measurement

43.5.25.4 Period measurement

If the channels (n) and (n+1) are configured to capture consecutive edges of the same polarity, then the period of the channel (n) input signal is measured. If both channels (n) and (n+1) are configured to capture rising edges (channel (n) ELSB:ELSA = 0:1 and channel (n+1) ELSB:ELSA = 0:1), then the period between two consecutive rising edges is measured. If both channels (n) and (n+1) are configured to capture falling edges (channel (n) ELSB:ELSA = 1:0 and channel (n+1) ELSB:ELSA = 1:0), then the period between two consecutive falling edges is measured.

The period measurement can be made in One-Shot Capture mode or Continuous Capture mode.

The following figure shows an example of the Dual Edge Capture – One-Shot mode used to measure the period between two consecutive rising edges. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. The DECAP bit is set to enable the measurement of next period. The channel (n) CHF bit is set when the first rising edge is detected, that is, the edge selected by channel (n) ELSB:ELSA bits. The channel (n+1) CHF bit is set and DECAP bit is cleared when the second rising edge is detected, that is, the edge selected by channel (n+1) ELSB:ELSA bits. Both DECAP and channel (n+1) CHF bits indicate when two selected edges were captured and the C(n)V and C(n+1)V registers are ready for reading.



Note
- The commands set DECAPEN, set DECAP, clear channel (n) CHF, and clear channel (n+1) CHF are made by the user.
- Problem 1: channel (n) input = 0, set DECAP, not clear channel (n) CHF, and not clear channel (n+1) CHF.
- Problem 2: channel (n) input = 1, set DECAP, not clear channel (n) CHF, and clear channel (n+1) CHF.
- Problem 3: channel (n) input = 1, set DECAP, not clear channel (n) CHF, and not clear channel (n+1) CHF.

Figure 319. Dual Edge Capture – One-Shot mode to measure of the period between two consecutive rising edges

The following figure shows an example of the Dual Edge Capture – Continuous mode used to measure the period between two consecutive rising edges. The DECAPEN bit selects the Dual Edge Capture mode, so it remains set. While the DECAP bit is set the configured measurements are made. The channel (n) CHF bit is set when the first rising edge is detected, that is, the edge selected by channel (n) ELSB:ELSA bits. The channel (n+1) CHF bit is set when the second rising edge is detected, that is, the edge selected by channel (n+1) ELSB:ELSA bits. The channel (n+1) CHF bit indicates when two edges of the period were captured and the C(n)V and C(n+1)V registers are ready for reading.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a2-9e51-390d3e6b6d1a

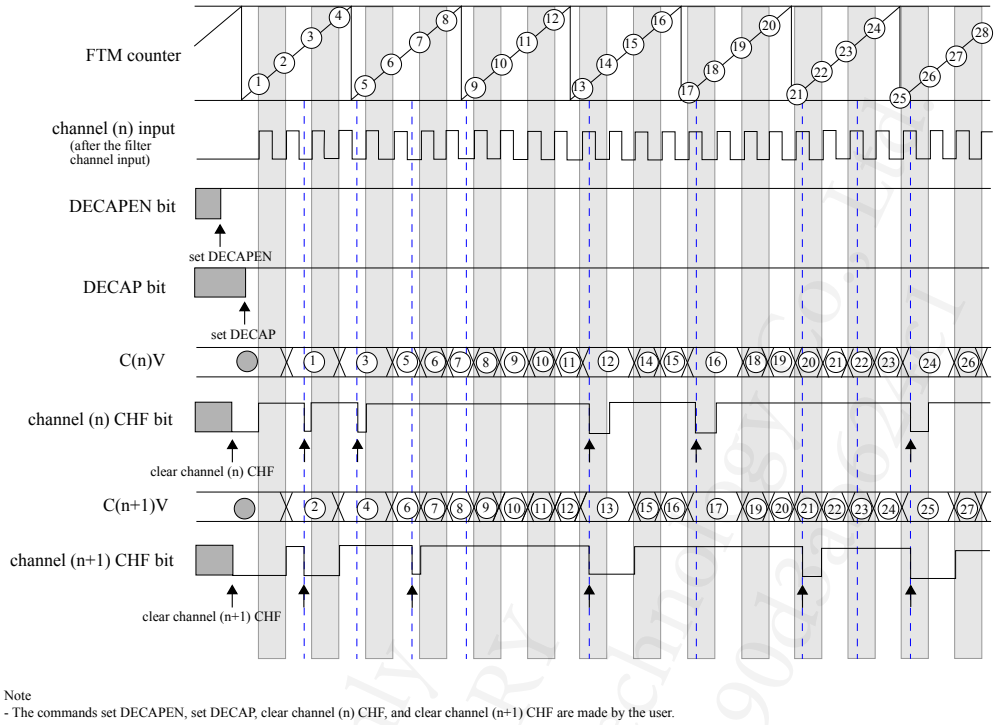


Figure 320. Dual Edge Capture – Continuous mode to measure of the period between two consecutive rising edges

43.5.25.5 Read coherency mechanism

The Dual Edge Capture mode implements a read coherency mechanism between the FTM counter value captured in C(n)V and C(n+1)V registers. The read coherency mechanism is illustrated in the following figure. In this example, the channels (n) and (n+1) are in Dual Edge Capture – Continuous mode for positive polarity pulse width measurement. Thus, the channel (n) is configured to capture the FTM counter value when there is a rising edge at channel (n) input signal, and channel (n+1) to capture the FTM counter value when there is a falling edge at channel (n) input signal.

When a rising edge occurs in the channel (n) input signal, the FTM counter value is captured into channel (n) capture buffer. The channel (n) capture buffer value is transferred to C(n)V register when a falling edge occurs in the channel (n) input signal. C(n)V register has the FTM counter value when the previous rising edge occurred, and the channel (n) capture buffer has the FTM counter value when the last rising edge occurred.

When a falling edge occurs in the channel (n) input signal, the FTM counter value is captured into channel (n+1) capture buffer. The channel (n+1) capture buffer value is transferred to C(n+1)V register when the C(n)V register is read.

In the following figure, the read of C(n)V returns the FTM counter value when the event 1 occurred and the read of C(n+1)V returns the FTM counter value when the event 2 occurred.

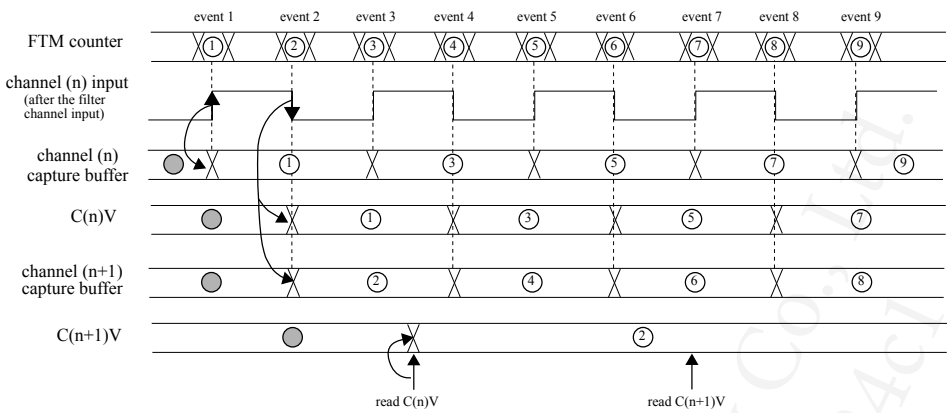


Figure 321. Dual Edge Capture mode read coherency mechanism

C(n)V register must be read prior to C(n+1)V register in dual edge capture one-shot and continuous modes for the read coherency mechanism works properly.

43.5.26 Quadrature Decoder Mode

The quadrature decoder mode is enabled if QUADEN = 1. The quadrature decoder mode uses the input signals phase A and B to control the FTM counter increment and decrement.

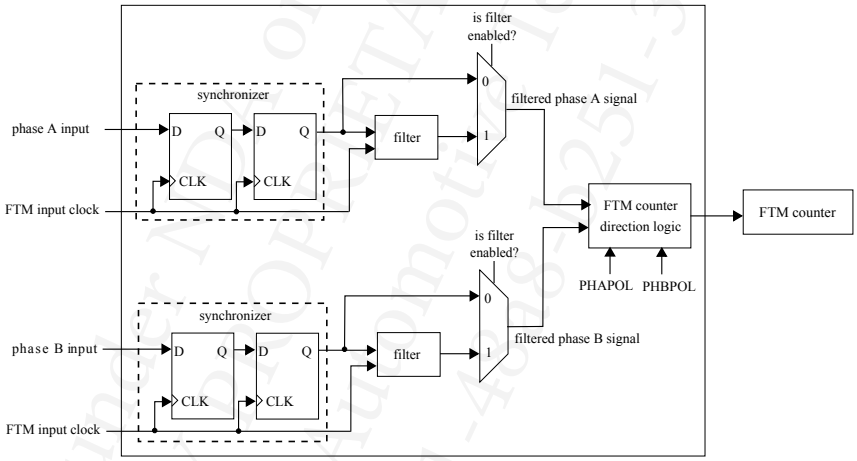


Figure 322. Diagram for Quadrature Decoder when FLTPS[3:0] = 0

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHIPTOP Technology Co., Ltd.
c8c462e5-4821-4819-b251-390d3a1624c1

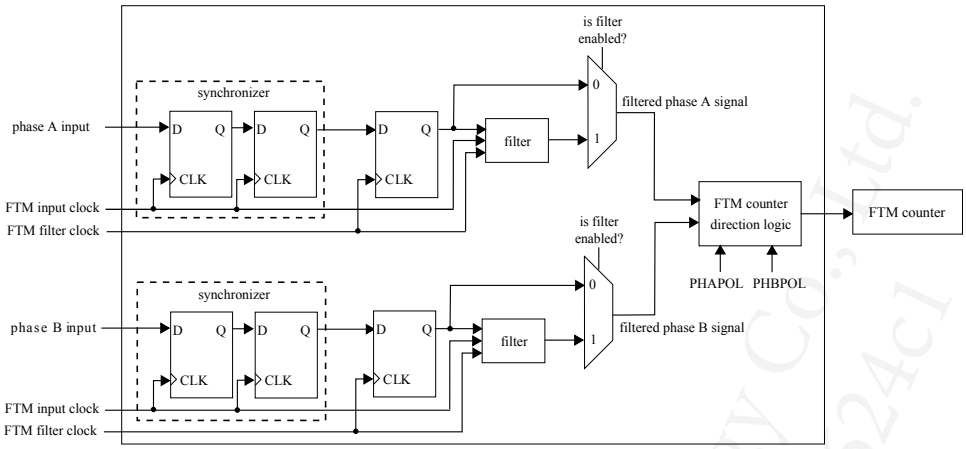


Figure 323. Diagram for Quadrature Decoder when FLTPS[3:0] ≠ 0

Each one of input signals phase A and B has a filter that is equivalent to the channel input filter (Filter for Input Capture Mode). The phase A input filter is enabled by PHAFLTREN bit and its value is defined by CH0FVAL[3:0] bits. The phase B input filter is enabled by PHBFLTREN bit and its value is defined by CH1FVAL[3:0] bits.

Except for CH0FVAL[3:0] and CH1FVAL[3:0] bits, no channel logic is used in quadrature decoder mode.

NOTE

The FTM counter is clocked by the phase A and B input signals when quadrature decoder mode is enabled. Therefore it is expected that the quadrature decoder mode be used only with the FTM channels in input capture or output compare modes.

NOTE

An edge at phase A must not occur together an edge at phase B and vice-versa.

The PHAPOL bit selects the polarity of the phase A input, and the PHBPOL bit selects the polarity of the phase B input.

The QUADMODE selects the encoding mode used in the quadrature decoder mode. If QUADMODE = 1, then the count and direction encoding mode is enabled; see the following figure. In this mode, the phase B input value indicates the counting direction, and the phase A input defines the counting rate. The FTM counter is updated when there is a rising edge at phase A input signal.

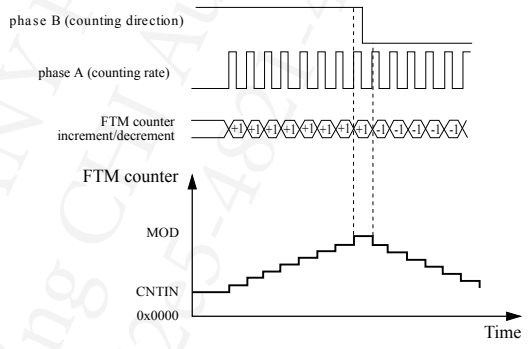


Figure 324. Quadrature Decoder – Count and Direction Encoding mode

If QUADMODE = 0, then the phase A and phase B Encoding mode is enabled; see the following figure. In this mode, the relationship between phase A and B signals indicates the counting direction, and phase A and B signals define the counting rate. The FTM counter is updated when there is an edge either at the phase A or phase B signals.

If PHAPOL = 0 and PHBPOL = 0, then the FTM counter increment happens when:

- there is a rising edge at phase A signal and phase B signal is at logic zero;
- there is a rising edge at phase B signal and phase A signal is at logic one;
- there is a falling edge at phase B signal and phase A signal is at logic zero;
- there is a falling edge at phase A signal and phase B signal is at logic one;

and the FTM counter decrement happens when:

- there is a falling edge at phase A signal and phase B signal is at logic zero;
- there is a falling edge at phase B signal and phase A signal is at logic one;
- there is a rising edge at phase B signal and phase A signal is at logic zero;
- there is a rising edge at phase A signal and phase B signal is at logic one.

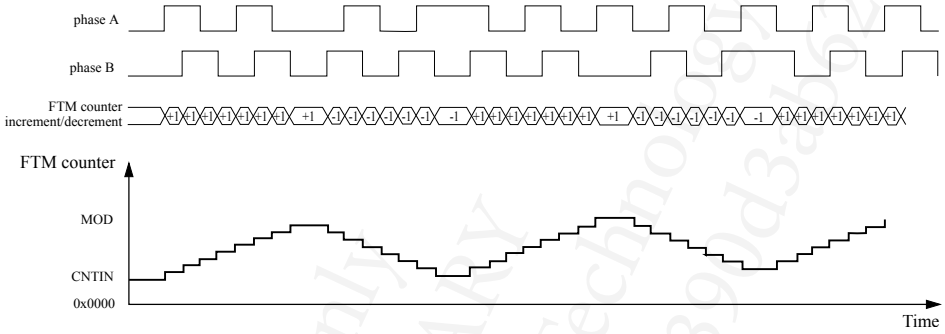


Figure 325. Quadrature Decoder – Phase A and Phase B Encoding Mode

The following figure shows the FTM counter overflow in up counting. In this case, when the FTM counter changes from MOD to CNTIN, TOF and TOFDIR bits are set. TOF bit indicates the FTM counter overflow occurred. TOFDIR indicates the counting was up when the FTM counter overflow occurred.

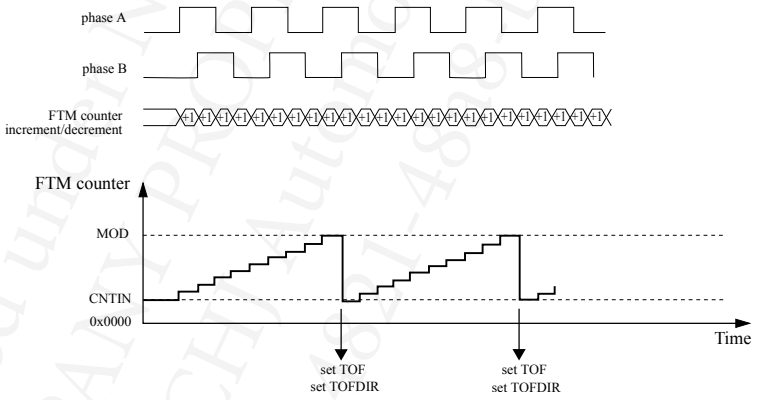


Figure 326. FTM Counter Overflow in Up Counting for Quadrature Decoder Mode

The following figure shows the FTM counter overflow in down counting. In this case, when the FTM counter changes from CNTIN to MOD, TOF bit is set and TOFDIR bit is cleared. TOF bit indicates the FTM counter overflow occurred. TOFDIR indicates the counting was down when the FTM counter overflow occurred.

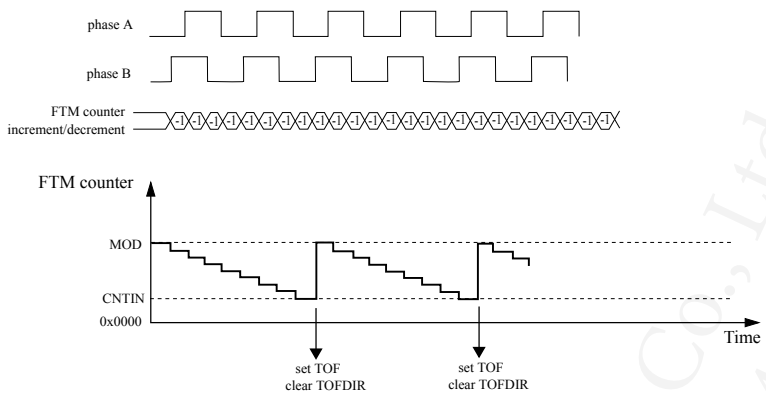


Figure 327. FTM Counter Overflow in Down Counting for Quadrature Decoder Mode

43.5.26.1 Quadrature Decoder boundary conditions

The following figures show the FTM counter responding to motor jittering typical in motor position control applications.

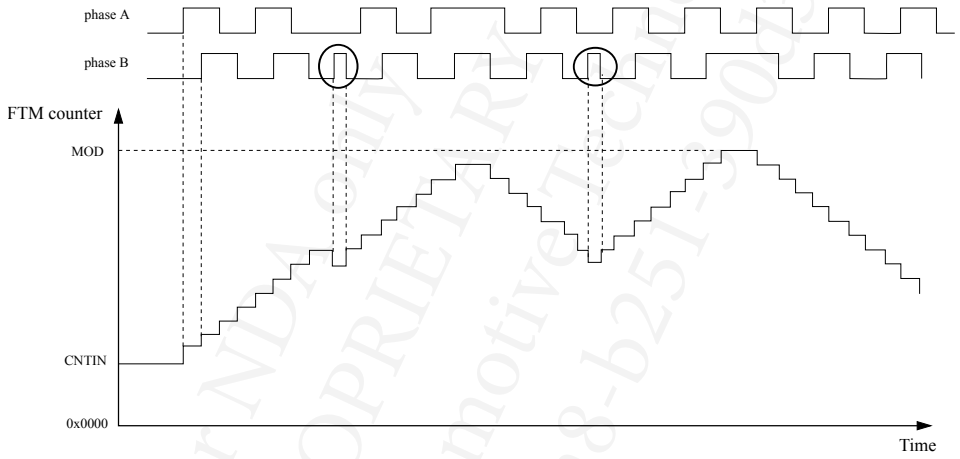


Figure 328. Motor position jittering in a mid count value

The following figure shows motor jittering produced by the phase B and A pulses respectively:

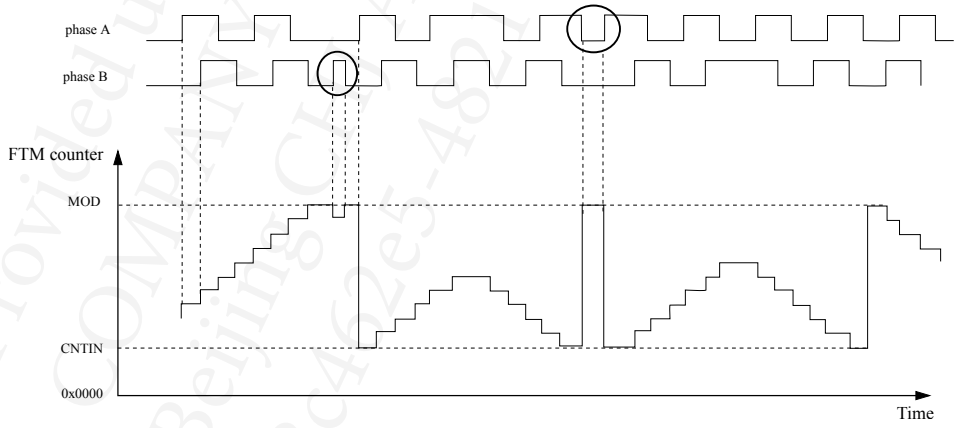


Figure 329. Motor position jittering near maximum and minimum count value

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIAT Automotive Technology Co., Ltd.
8c42e3-482-18a8-b251-90d3b624c1

The first highlighted transition causes a jitter on the FTM counter value near the maximum count value (MOD). The second indicated transition occurs on phase A and causes the FTM counter transition between the maximum and minimum count values which are defined by MOD and CNTIN registers.

The appropriate settings of the phase A and phase B input filters are important to avoid glitches that may cause oscillation on the FTM counter value. The preceding figures show examples of oscillations that can be caused by poor input filter setup. Thus, it is important to guarantee a minimum pulse width to avoid these oscillations.

43.5.27 Debug mode

When the chip is in Debug mode, the BDMMODE[1:0] bits select the behavior of the FTM counter, the channel (n) CHF bit, the channels output, and the writes to the MOD, CNTIN, and C(n)V registers according to the following table.

Table 283. FTM behavior when the chip Is in Debug mode

BDMMODE	FTM Counter	channel (n) CHF bit	FTM Channels Output	Writes to MOD, CNTIN, and C(n)V Registers
00	Stopped	can be set	Functional mode	Writes to these registers bypass the registers buffers
01	Stopped	is not set	The channels outputs are forced to their safe value according to POLn bit	Writes to these registers bypass the registers buffers
10	Stopped	is not set	The channels outputs are frozen when the chip enters in Debug mode	Writes to these registers bypass the registers buffers
11	Functional mode	can be set	Functional mode	Functional mode

Note that if BDMMODE[1:0] = 2’b00 then the channels outputs remain at the value when the chip enters in Debug mode, because the FTM counter is stopped. However, the following situations modify the channels outputs in this Debug mode.

- Write any value to CNT register; see Counter reset. In this case, the FTM counter is updated with the CNTIN register value and the channels outputs are updated to the initial value – except for those channels set to Output Compare mode.
- FTM counter is reset by PWM Synchronization mode; see FTM counter synchronization. In this case, the FTM counter is updated with the CNTIN register value and the channels outputs are updated to the initial value – except for channels in Output Compare mode.
- In the channels outputs initialization, the channel (n) output is forced to the CH(n)OI bit value when the value 1 is written to INIT bit. See Initialization.

NOTE

If CLKS[1:0] = 2'b00 in Debug, a non-zero value is written to CLKS in Debug, and CnV = CNTIN when the Debug is disabled, then the CHF bit is set (since if the channel is a 0% EPWM signal) when the Debug is disabled.

43.5.28 Reload Points

This feature allows to update the registers CNTIN, HCR, MOD and C(n)V with the value of their write buffer at the selected reload point.

NOTE

- This feature is independent of the PWM synchronization.
- At these reload points neither the channels outputs nor the FTM counter are changed. Software must select these reload points at the safe points in time.

43.5.28.1 Reload Opportunities

The reload opportunities are:

1. At the half cycle

   This reload opportunity is enabled if (HCSEL = 1) and it happens at the half cycle (FTM counter = HCR register). The software should calculate the half cycle value according to the FTM counter configuration, then writes this value to the register HCR.

2. At the channel (n) match

   This reload opportunity is enabled if (CH(n)SEL = 1) and it happens at the channel (n) match (FTM counter = C(n)V).

3. When the FTM counter is an up counter

   This reload opportunity is when the FTM counter changes from (MOD) to (CNTIN - 1) and it is always enabled.

   The following figure shows an example of the reload opportunities at the half cycle, at the channels match, and when the FTM counter is an up counter.

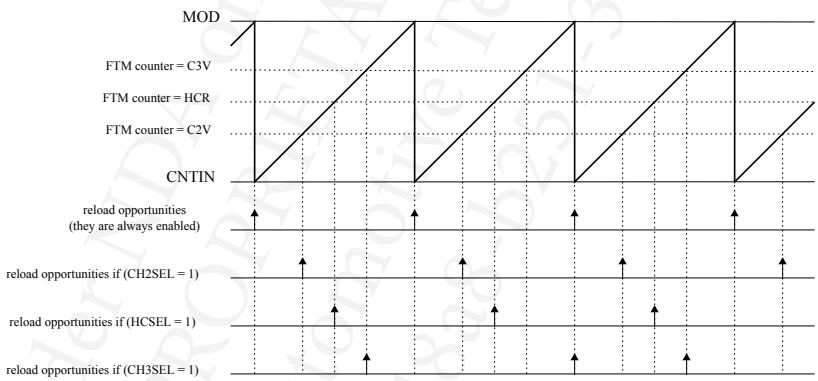


Figure 330. Reload opportunities when the FTM counter is an up counter

4. When the FTM counter is an up-down counter

   In this case, the reload opportunities are enabled by the bits CNTMAX and CNTMIN according to Table 284.

Table 284. Reload opportunities enabled by the bits CNTMAX and CNTMIN when the FTM counter is up-down counter

CNTMAX	CNTMIN	Reload Opportunities
0	0	when the FTM counter changes from (MOD) to (MOD - 1)
0	1	when the FTM counter changes from (CNTMIN) to (CNTMIN + 1)
1	0	when the FTM counter changes from (MOD) to (MOD - 1)
1	1	• when the FTM counter changes from (MOD) to (MOD - 1), and • when the FTM counter changes from (CNTMIN) to (CNTMIN + 1)

The following figure shows an example of the reload opportunities at the half cycle, at the channels match, and when the FTM counter is an up-down counter.

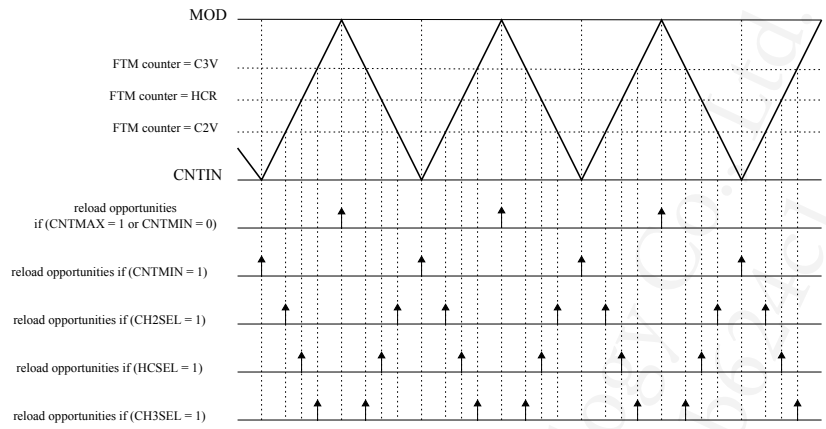


Figure 331. Reload opportunities when the FTM counter is an up-down counter

43.5.28.2 Frequency of Reload Opportunities

The LDFQ[4:0] bits define the number of enabled reload opportunities should happen until an enabled reload opportunity becomes a reload point. The following figure shows an example when the LDFQ[4:0] = 4.

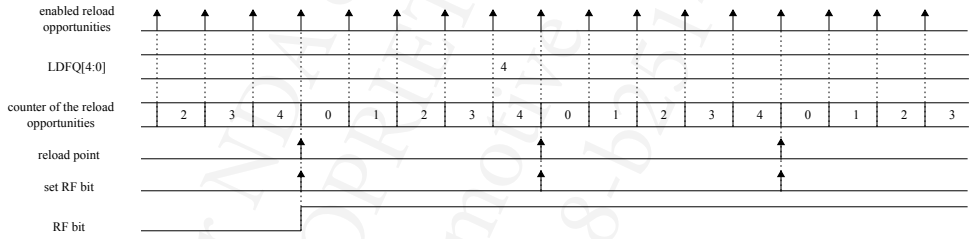


Figure 332. Frequency of Reload Opportunities with LDFQ[4:0] = 4

If LDFQ[4:0] = 0, then all reload opportunities are reload points.

The counter of the reload opportunities is reset when there is a write to the register CNT.

The RF bit is set at each reload point (see the figure above) independent of LDOK bit value. The reload point interrupt is generated when (RF = 1) and (RIE = 1).

43.5.28.3 Update of the Registers

After writing new value to the registers with write buffer, selecting which of them will be updated (according to Table 285), selecting the reload opportunities, selecting the frequency of the reload opportunities, thus the LDOK bit should be set to enable the update of these registers at the next reload point.

Table 285. Additional conditions to update the registers

Register	Additional Condition
CNTIN	CNTINC = 1
HCR	-

Table continues on the next page...

Table 285. Additional conditions to update the registers (continued)

Register	Additional Condition
MOD	-
C(n)V and C(n+1)V	SYNCENm = 1, where m is the pair of the channels (n) and (n+1)

43.5.29 Global Load

The global load mechanism allows several modules to have their double buffered registers synchronously reloaded after a synchronization event if a write to one operation is performed in the global load OK (GLDOK) bit in the PWMLOAD register. Global load may be enabled or disabled configuring the global load enable (GLEN) bit in the PWMLOAD register. Writing one in the GLDOK bit with GLEN enabled has the same effect of writing one in the LDOK bit. Refer to SoC specific information about global load connections.

Global load mechanism allows MOD, HCR, CNTIN, and C(n)V registers to be updated with the content of the register buffer at configurable reload point. The figure below shows an example of connection between FTM global load inputs and outputs considering that GLDOK bit is implemented outside from FTM module.

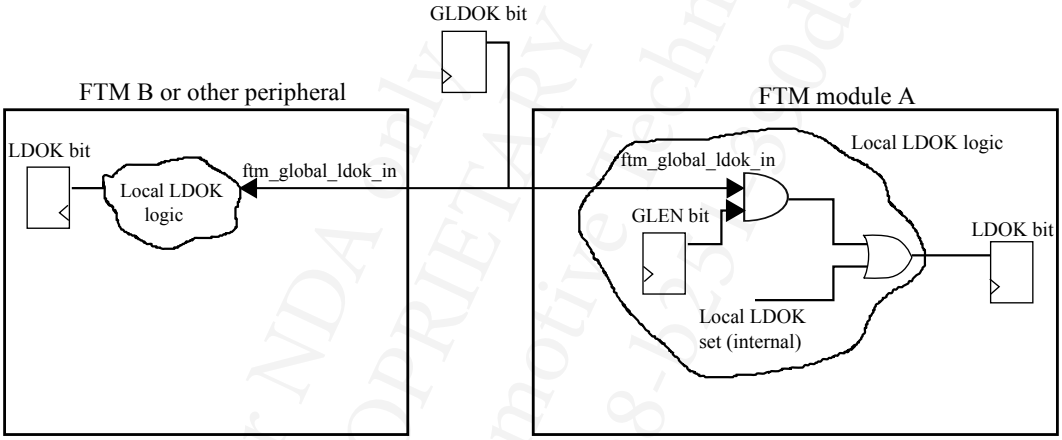


Figure 333. Global load logic

43.5.30 Global time base (GTB)

The global time base (GTB) is a FTM function that allows the synchronization of multiple FTM modules on a chip. The following figure shows an example of the GTB feature used to synchronize two FTM modules. In this case, the FTM A and B channels can behave as if just one FTM module was used, that is, a global time base.

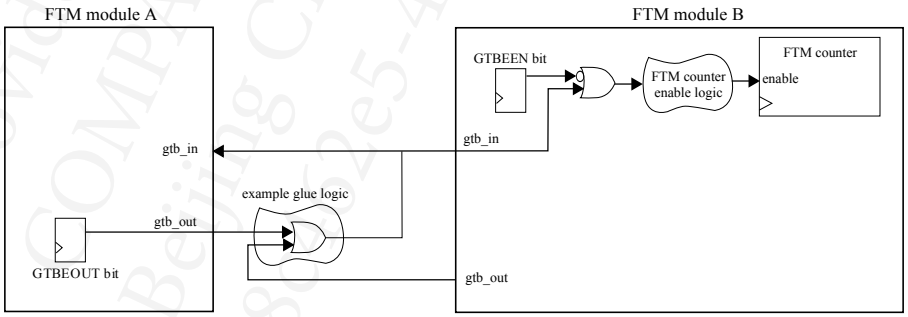


Figure 334. Global time base (GTB) block diagram

The GTB functionality is implemented by the GTBEEN and GTBEOUT bits in the CONF register, the input signal *gtb_in*, and the output signal *gtb_out*. The GTBEEN bit enables gtb_in to control the FTM counter enable signal:

- If GTBEEN = 0, each one of FTM modules works independently according to their configured mode.
- If GTBEEN = 1, the FTM counter update is enabled only when gtb_in is 1.

In the configuration described in the preceding figure, FTM modules A and B have their FTM counters enabled if at least one of the gtb_out signals from one of the FTM modules is 1. There are several possible configurations for the interconnection of the gtb_in and gtb_out signals, represented by the example glue logic shown in the figure. Note that these configurations are chip-dependent and implemented outside of the FTM modules. See the chip-specific FTM information for the chip's specific implementation.

NOTE

- In order to use the GTB signals to synchronize the FTM counter of different FTM modules, the configuration of each FTM module should guarantee that its FTM counter starts counting as soon as the gtb_in signal is 1.
- The GTB feature does not provide continuous synchronization of FTM counters, meaning that the FTM counters may lose synchronization during FTM operation. The GTB feature only allows the FTM counters to *start* their operation synchronously.

43.5.30.1 Enabling the global time base (GTB)

To enable the GTB feature, follow these steps for each participating FTM module:

1. Stop the FTM counter: Write 00b to SC[CLKS].
2. Program the FTM to the intended configuration. The FTM counter mode needs to be consistent across all participating modules.
3. Write 1 to CONF[GTBEEN] and write 0 to CONF[GTBEOUT] at the same time.
4. Select the intended FTM counter clock source in SC[CLKS]. The clock source needs to be consistent across all participating modules.
5. Reset the FTM counter: Write any value to the CNT register.

To initiate the GTB feature in the configuration described in the preceding figure, write 1 to CONF[GTBEOUT] in the FTM module used as the time base.

43.5.31 Channel trigger output

The channel trigger output provides a trigger signal which has one FTM input clock period width in the channel (n) output.

If the TRIGMODE bit of the CnSC register is set (TRIGMODE = 1), a trigger pulse with one FTM input clock width is generated in the channel (n) output when a match occurs. It is only allowed to use trigger mode when channel (n) is in EPWM or CPWM modes.

The figures below show some cases of channel (n) trigger generation in the channel (n) output.

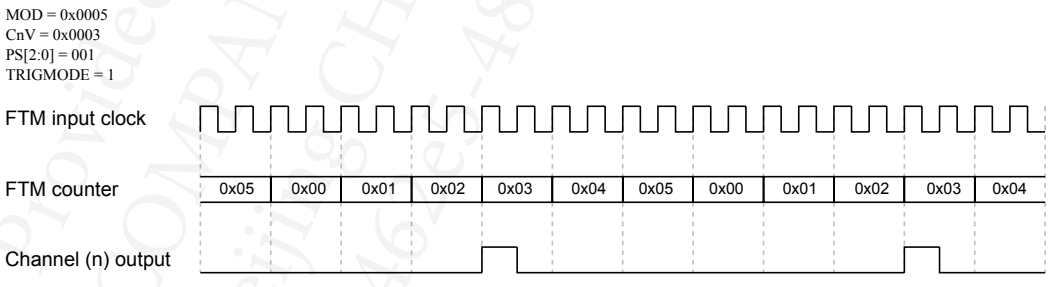


Figure 335. Example of channel (n) trigger at the channel (n) output in EPWM mode

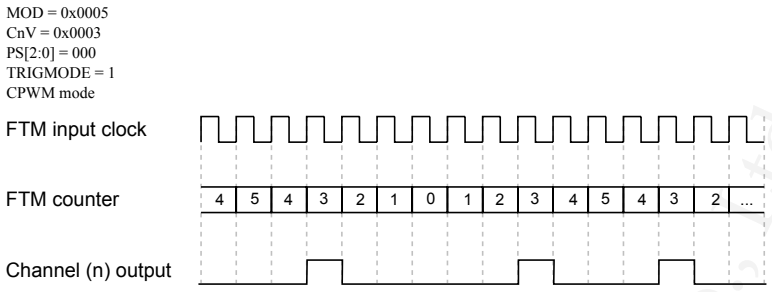


Figure 336. Example of channel (n) trigger at the channel (n) output in CPWM mode

43.5.32 External Control of Channels Output

The channel (n) PWMEN bit can be used in an FTM external logic to control the final value of the channel (n) output. The following figure shows an example of this external logic.

The term "channel (n) output" means the channel (n) output value after the Polarity Control. See Features Priority and Polarity Control for more details.

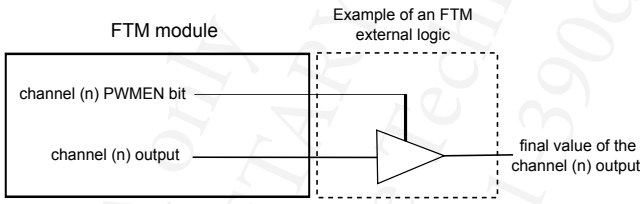


Figure 337. Example of the External Control of the Channel (n) Output

43.5.33 Dithering

FTM implements a fractional delay to achieve fine resolution on the generated PWM signals using dithering. The dithering can be used by applications where more resolution than one unit of the FTM counter is needed.

Two kinds of dithering are available: PWM period dithering and edge dithering.

43.5.33.1 PWM Period Dithering

The PWM period dithering is enabled when a non-zero value is written to FRACMOD.

The internal accumulator used in the PWM period dithering is reset when:

- the field MOD of the register MOD_MIRROR is updated with the value of its write buffer,
- the FRACMOD is updated with the value of its write buffer, or
- the FTM counter is stopped.

NOTE

For the PWM period dithering, the register MOD_MIRROR should be used instead of the register MOD.

To avoid inconsistencies, the field FRACMOD is cleared when the field MOD of the register MOD is updated with the value of its write buffer.

The PWM period dithering is not available:

- when the FTM counter is a free running counter
- when the FTM is in quadrature decoder mode

43.5.33.1.1 Up Counting

When the FTM counter is an up counter and the PWM period dithering is enabled, at the end of each PWM period, the FRACMOD value is added to an internal 5-bit accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), then one unit of FTM counter is added to the end of the current PWM period, and the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

Due to one unit of FTM counter that can be added to the PWM period, the largest valid value for MOD is 0xFFFE for PWM period dithering with unsigned counting and 0x7FFE for PWM period dithering with signed counting.

The following figures show some examples of PWM period dithering when the FTM counter is an up counter.

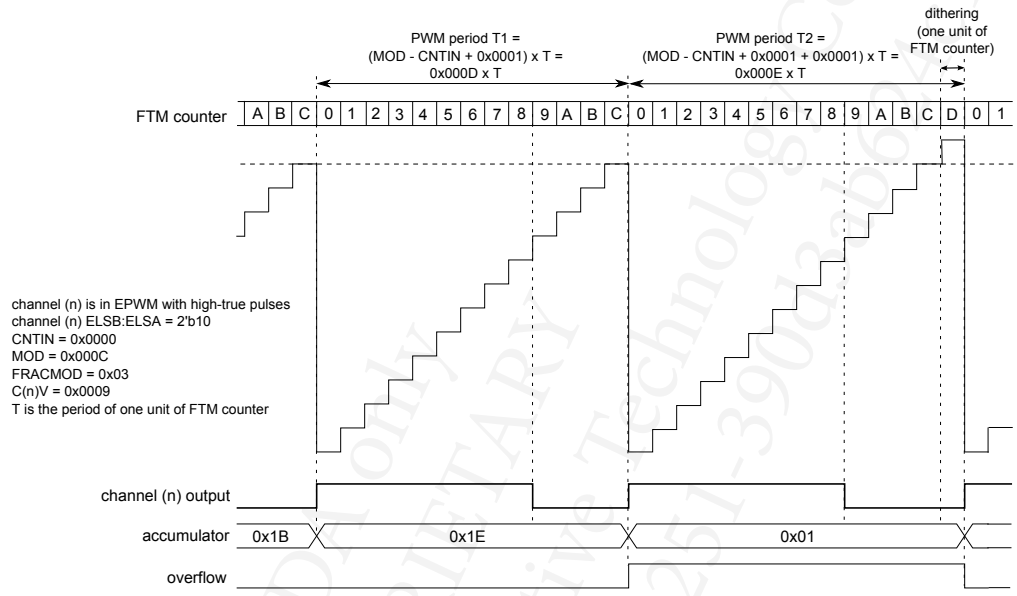


Figure 338. PWM Period Dithering with Up Counting

Assuming:

- the FTM counter is an up counter,
- T is one unit of FTM counter,
- the PWM period without period dithering is [(MOD - CNTIN + 1) x T],
- the number of PWM periods with period dithering is FRACMOD,
- the PWM period with period dithering is [(MOD - CNTIN + 1 + 1) x T],

thus, the average period (in decimal) is [(MOD - CNTIN + 1) + (FRACMOD/32)] x T, where the integer value is (MOD - CNTIN + 1) and the fractional value is (FRACMOD/32). See the example below.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4e24-48a8-b251-390d2ab6b6f1

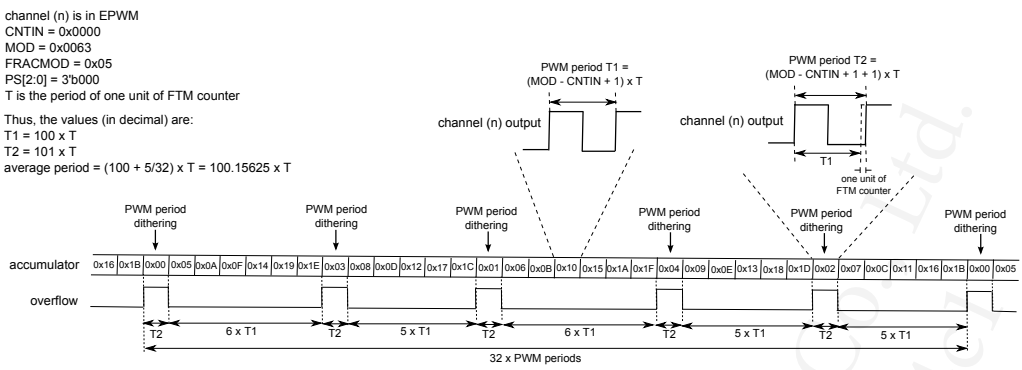


Figure 339. Example of Average Period when the PWM Period Dithering is used with the Up Counting

NOTE

For the generation of 100% PWM signal in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using EPWM mode and PWM Period Dithering, it is recommended to use (C(n) > MOD + 1).

For the generation of PWM signals in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Combine mode and PWM Period Dithering, it is recommended to use:

- For 0% PWM signal: (C(n)V > MOD + 1) and (C(n+1)V > MOD + 1);
- For 100% PWM signal: (C(n)V = CNTIN) and (C(n+1)V > MOD + 1).

For the generation of PWM signals in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Modified Combine PWM mode and PWM Period Dithering, it is recommended to use:

- For 0% PWM signal: (C(n)V > MOD + 1) and (CNTIN ≤ C(n+1)V ≤ MOD);
- For 100% PWM signal: (CNTIN ≤ C(n)V ≤ MOD) and (C(n+1)V > MOD + 1).

43.5.33.1.2 Up-Down Counting

When the FTM counter is an up-down counter and the PWM period dithering is enabled, at the end of each PWM period, the FRACMOD value is added to an internal 5-bit accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), then one unit of FTM counter is added to the end of the current PWM period and other unit is added to the begin of the next PWM period (see the figure below). After the accumulator overflows, the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

Due to one unit of FTM counter that can be added to the PWM period, the largest valid value for MOD is 0x7FFE for PWM period dithering in up-down counting (CPWM mode).

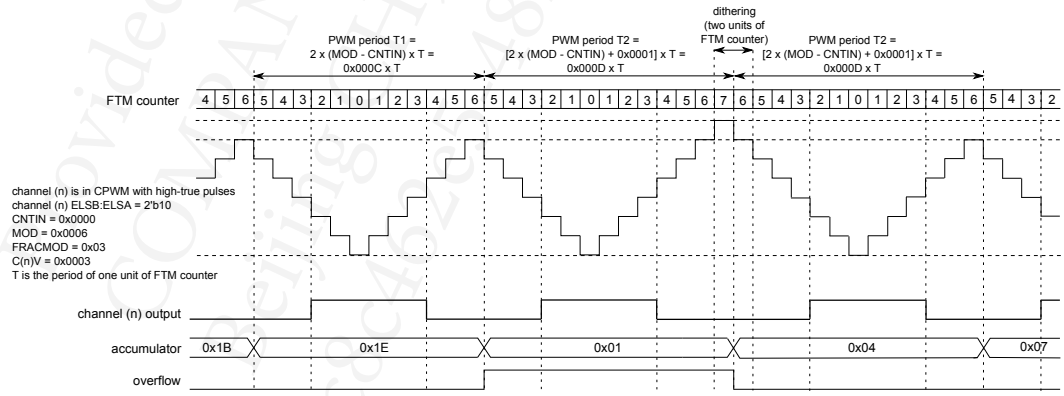


Figure 340. PWM Period Dithering with Up-Down Counting

NOTE

For the generation of 100% PWM signal in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using CPWM mode and PWM Period Dithering, it is recommended to use (C(n)V[15] = 0) and (C(n)V > MOD + 1) and (MOD ≠ 0x0000).

43.5.33.2 PWM Edge Dithering

The channel (n) internal accumulator used in the PWM edge dithering is reset when:

- the field VAL of the register C(n)V_MIRROR is updated with the value of its write buffer,
- the FRACVAL is updated with the value of its write buffer, or
- the FTM counter is stopped.

NOTE

For the PWM edge dithering, the register C(n)V_MIRROR should be used instead of the register C(n)V.

To avoid inconsistencies, the field FRACVAL is cleared when the field VAL of the register C(n)V is updated with the value of its write buffer.

The PWM edge dithering is not available:

- to the channel in input modes, and
- to the channel in output compare mode.

43.5.33.2.1 EPWM Mode

The PWM edge dithering for channel (n) in EPWM mode is enabled when a non-zero value is written to the channel (n) FRACVAL.

If the channel (n) is in EPWM mode and the PWM edge dithering is enabled, at the end of each EPWM period, the channel (n) FRACVAL value is added to the channel (n) internal 5-bit accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

In this configuration, the initial edge of EPWM duty cycle happens when (FTM counter = CNTIN), its position is not modified by the PWM edge dithering. If there was not the overflow of the channel (n) accumulator in the current EPWM period, then the final edge of EPWM duty cycle happens on the channel (n) match (FTM counter = C(n)V), that is, its position is not modified by the edge dithering. However, if there was the overflow of the channel (n) accumulator in the current EPWM period, then the final edge of EPWM duty cycle happens when (FTM counter = C(n)V + 0x0001).

The following figures show some examples of PWM edge dithering when the channel (n) is in EPWM mode.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a9-a1d3-390d3a0c24c1

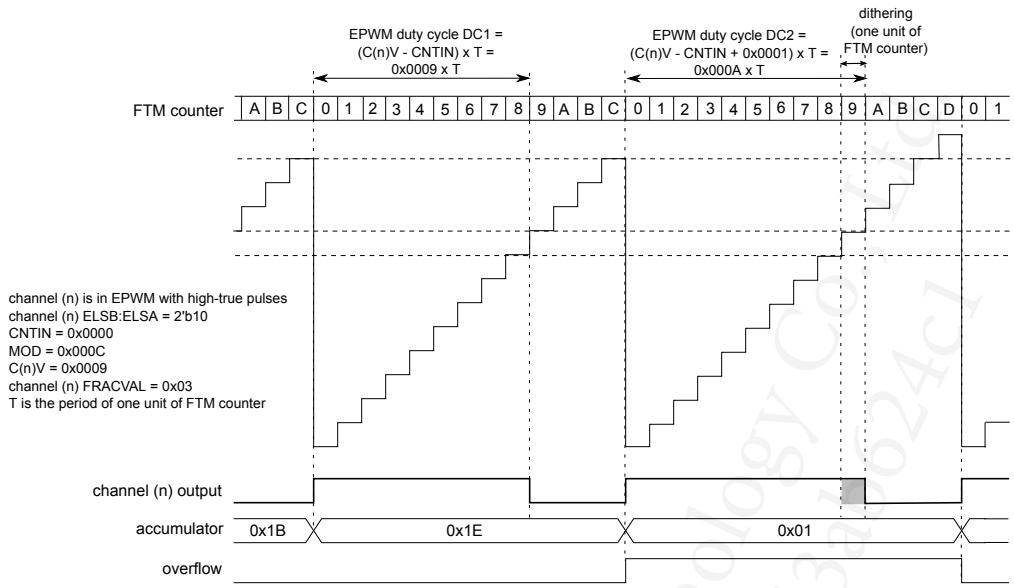


Figure 341. Channel (n) is in EPWM Mode with PWM Edge Dithering

Assuming:

- the channel (n) is in EPWM mode,
- T is one unit of FTM counter,
- the EPWM duty cycle without edge dithering is [(C(n)V - CNTIN) x T],
- the number of PWM periods which duty cycle that has edge dithering is FRACVAL,
- the EWM duty cycle with edge dithering is [(C(n)V - CNTIN + 1) x T],

thus, the average duty cycle (in decimal) is [(C(n)V - CNTIN) + (FRACVAL/32)] x T, where the integer value is (C(n)V - CNTIN) and the fractional value is (FRACVAL/32). See the example below.

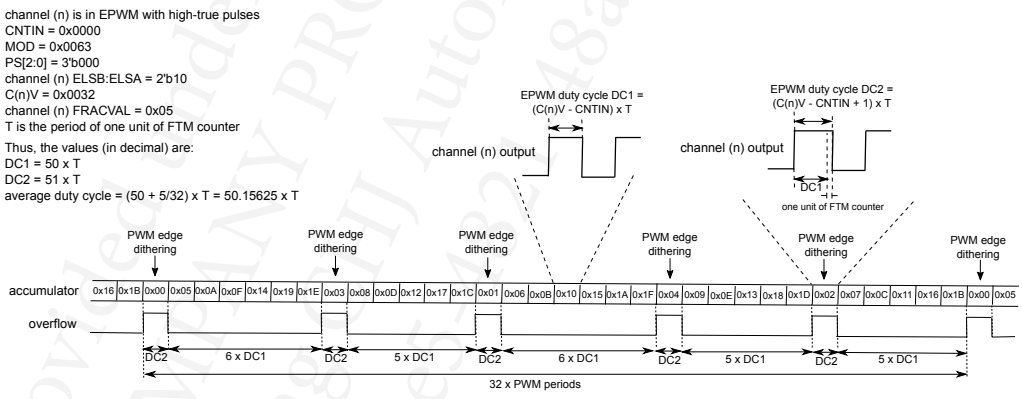


Figure 342. Example of Average Duty Cyle when the Channel (n) is in EPWM Mode with PWM Edge Dithering

43.5.33.2.2 CPWM Mode

The PWM edge dithering for channel (n) in CPWM mode is enabled when a non-zero value is written to the channel (n) FRACVAL.

If the channel (n) is in CPWM mode and the PWM edge dithering is enabled, at the end of each CPWM period, the channel (n) FRACVAL value is added to the channel (n) internal 5-bit accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

In this configuration, if there was not the overflow of the channel (n) accumulator in the current CPWM period, then the duty cycle is not modified by the PWM edge dithering, that is, the initial edge of CPWM duty cycle happens on channel (n) match (FTM counter = C(n)V) when the FTM counter is decrementing, and the final edge of CPWM duty cycle on channel (n) match when the FTM counter is incrementing.

However, if there was the overflow of the channel (n) accumulator in the current CPWM period, then the initial edge of CPWM duty cycle happens when (FTM counter = C(n)V + 0x0001) and the FTM counter is decrementing, and the final edge of CPWM duty cycle when (FTM counter = C(n)V + 0x0001) and the FTM counter is incrementing.

The following figure shows an example of PWM edge dithering when the channel (n) is in CPWM mode.

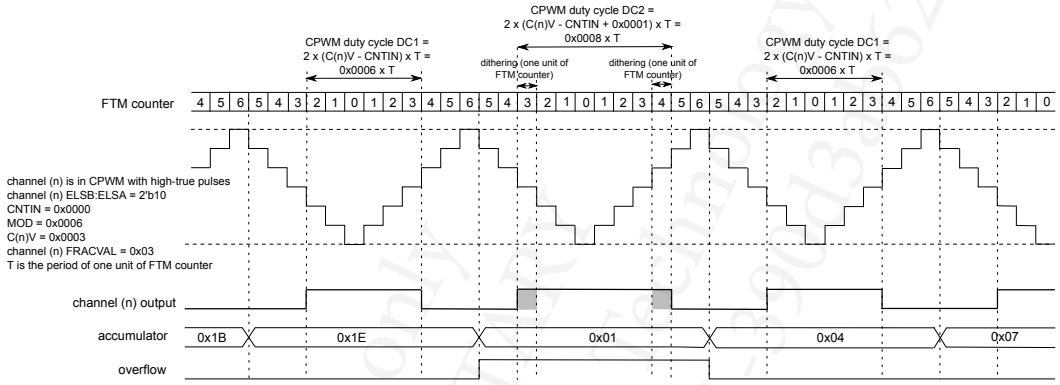


Figure 343. Channel (n) is in CPWM Mode with PWM Edge Dithering

43.5.33.2.3 Combine Mode

In the Combine mode, the PWM edge dithering can be done:

- in the channel (n) match (FTM counter = C(n)V) edge or
- in the channel (n+1) match (FTM counter = C(n+1)V) edge.

The channel (n) match edge dithering is enabled when a non-zero value is written to the channel (n) FRACVAL.

For the channel (n) match edge dithering, the channel (n) has an internal 5-bit accumulator. At the end of each PWM period, the channel (n) FRACVAL value is added to the channel (n) accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

If there was not the overflow of the channel (n) accumulator in the current PWM period, the channel (n) match edge is not modified, that is, it happens on channel (n) match. However, if there was the overflow of the channel (n) accumulator, the channel (n) match edge happens when (FTM counter = C(n)V + 0x0001).

The following figure shows an example of the channel (n) match edge dithering when the channels (n) and (n+1) are in Combine mode.

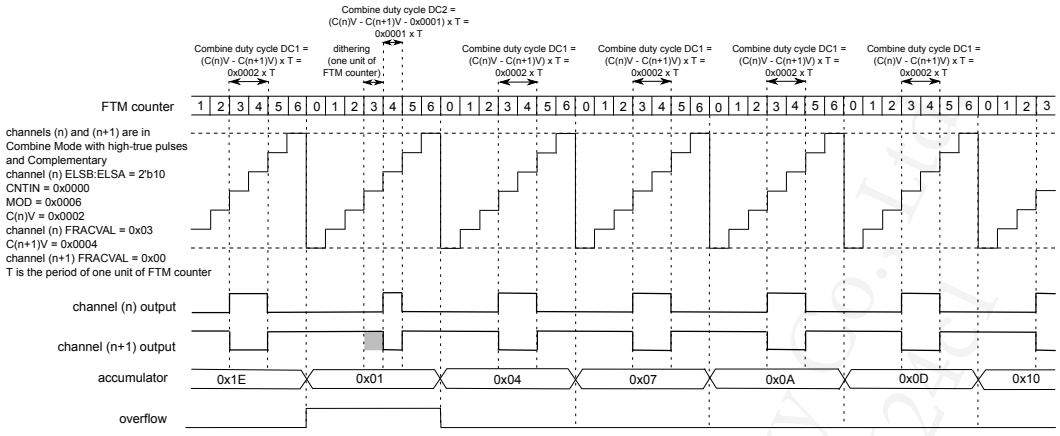


Figure 344. Channel (n) Match Edge Dithering in Combine Mode

The channel (n+1) match edge dithering is enabled when a non-zero value is written to the channel (n+1) FRACVAL.

For the channel (n+1) match edge dithering, the channel (n+1) has an internal 5-bit accumulator. At the end of each PWM period, the channel (n+1) FRACVAL value is added to the channel (n+1) accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

If there was not the overflow of the channel (n+1) accumulator in the current PWM period, the channel (n+1) match edge is not modified, that is, it happens on channel (n+1) match. However, if there was the overflow of the channel (n+1) accumulator, the channel (n+1) match edge happens when (FTM counter = C(n+1)V + 0x0001).

The following figure shows an example of the channel (n+1) match edge dithering when the channels (n) and (n+1) are in Combine mode.

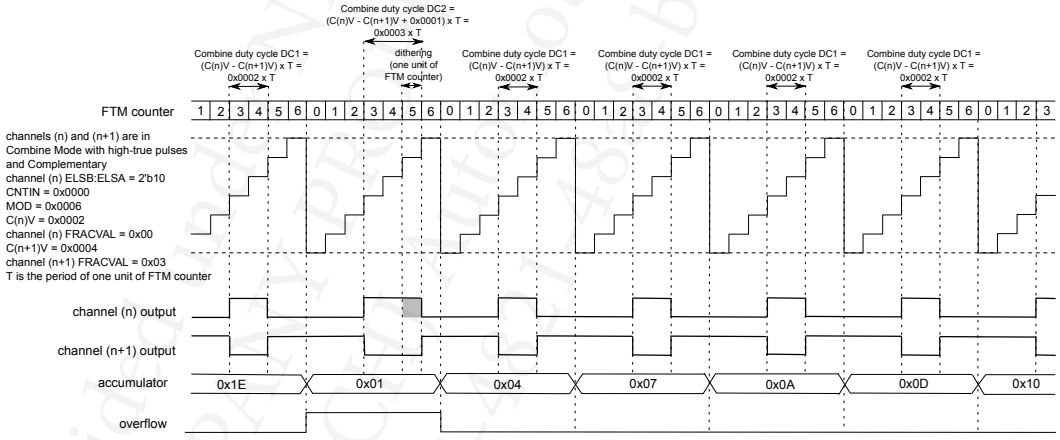


Figure 345. Channel (n+1) Match Edge Dithering in Combine Mode

NOTE

It is recommended to use only one PWM Edge Dithering (channel (n) PWM Edge Dithering or channel (n+1) PWM Edge Dithering) at a time.

For the generation of 0% PWM in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Combine mode and PWM Edge Dithering, it is recommended to use:

- (C(n)V < CNTIN or C(n)V > MOD) and (channel (n) FRACVAL is zero) and
- (channel (n+1) FRACVAL is zero).

For the generation of 100% PWM in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Combine mode and PWM Edge Dithering, it is recommended to use:

- (C(n)V = CNTIN) and (channel (n) FRACVAL is zero) and
- (C(n+1)V < CNTIN or C(n+1)V > MOD) and (channel (n+1) FRACVAL is zero).

43.5.33.2.4 Modified Combine PWM Mode

In the Modified Combine PWM mode, the PWM edge dithering can be done:

- in the channel (n) match (FTM counter = C(n)V) edge or
- in the channel (n+1) match (FTM counter = C(n+1)V) edge.

The channel (n) match edge dithering is enabled when a non-zero value is written to the channel (n) FRACVAL.

For the channel (n) match edge dithering, the channel (n) has an internal 5-bit accumulator. At the end of each PWM period, the channel (n) FRACVAL value is added to the channel (n) accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

If there was not the overflow of the channel (n) accumulator in the current PWM period, the channel (n) match edge is not modified, that is, it happens on channel (n) match. However, if there was the overflow of the channel (n) accumulator, the channel (n) match edge happens when (FTM counter = C(n)V + 0x0001).

The following figure shows an example of the channel (n) match edge dithering when the channels (n) and (n+1) are in Modified Combine PWM mode.

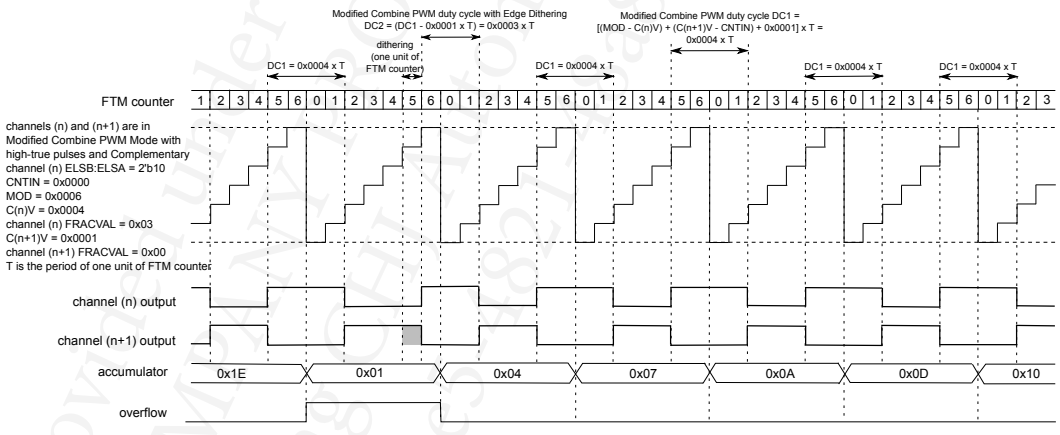


Figure 346. Channel (n) Match Edge Dithering in Modified Combine PWM Mode

The channel (n+1) match edge dithering is enabled when a non-zero value is written to the channel (n+1) FRACVAL.

For the channel (n+1) match edge dithering, the channel (n+1) has an internal 5-bit accumulator. At the end of each PWM period, the channel (n+1) FRACVAL value is added to the channel (n+1) accumulator. When this accumulator overflows (that is, the result of the adding is greater or equal than 0x20), the accumulator remains with the rest of the subtraction: (the result of this adding - 0x20).

If there was not the overflow of the channel (n+1) accumulator in the current PWM period, the channel (n+1) match edge is not modified, that is, it happens on channel (n+1) match. However, if there was the overflow of the channel (n+1) accumulator, the channel (n+1) match edge happens when (FTM counter = C(n+1)V + 0x0001).

The following figure shows an example of the channel (n+1) match edge dithering when the channels (n) and (n+1) are in Modified Combine PWM mode.

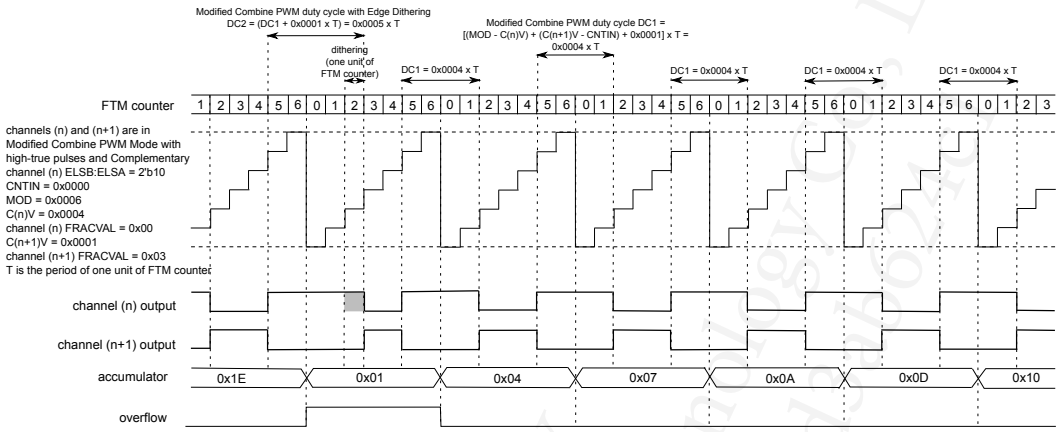


Figure 347. Channel (n+1) Match Edge Dithering in Modified Combine PWM Mode

NOTE

It is recommended to use only one PWM Edge Dithering (channel (n) PWM Edge Dithering or channel (n+1) PWM Edge Dithering) at a time.

For the generation of 0% PWM in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Modified Combine PWM mode and PWM Edge Dithering, it is recommended to use:

- (C(n)V < CNTIN or C(n)V > MOD) and (channel (n) FRACVAL is zero) and
- (CNTIN ≤ C(n+1)V ≤ MOD) and (channel (n+1) FRACVAL is zero).

For the generation of 100% PWM in the channel (n) (with channel (n) ELSB:ELSA = 2'b10) using Modified Combine PWM mode and PWM Edge Dithering, it is recommended to use:

- (CNTIN ≤ C(n)V ≤ MOD) and (channel (n) FRACVAL is zero) and
- (C(n+1)V < CNTIN or C(n+1)V > MOD) and (channel (n+1) FRACVAL is zero).

43.6 Reset Overview

The FTM is reset whenever any chip reset occurs.

When the FTM exits from reset:

- the FTM counter and the prescaler counter are zero and are stopped (CLKS[1:0] = 2'b00);
- the timer overflow interrupt is zero (Timer Overflow Interrupt);
- the channels interrupts are zero (Channel (n) Interrupt);
- the channels are in input capture mode (Input Capture Mode);
- the channels outputs are zero;
- the channels ELSB:ELSA = 0:0 (Channel Modes) and PWMEN = 0 (External Control of Channels Output).

The following figure shows the FTM behavior after the reset. At the reset (item 1), the FTM counter is disabled (CLKS[1:0] = 2'b00), its value is updated to zero and the pins are not controlled by FTM (Channel Modes).

After the reset, the FTM should be configurated (item 2). It is necessary to define the FTM counter mode, the FTM counting limits (MOD and CNTIN registers value), the channels mode and CnV registers value according to the channels mode.

Thus, it is recommended to write any value to CNT register (item 3). This write updates the FTM counter with the CNTIN register value and the channels output with its initial value (except for channels in output compare mode) (Counter reset).

The next step is to select the FTM counter clock by the CLKS[1:0] bits (item 4). It is important to highlight that the pins are only controlled by FTM when CLKS[1:0] bits are different from zero.

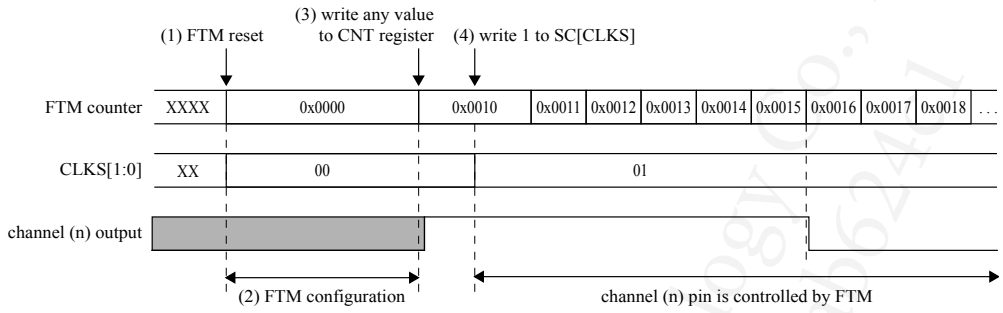


Figure 348. FTM behavior after reset when the channel (n) is in Combine mode

The following figure shows an example when the channel (n) is in Output Compare mode and the channel (n) output is toggled when there is a match. In the Output Compare mode, the channel output is not updated to its initial value when there is a write to CNT register (item 3). In this case, use the software output control (Software Output Control Mode) or the initialization (Initialization) to update the channel output to the selected value (item 4).

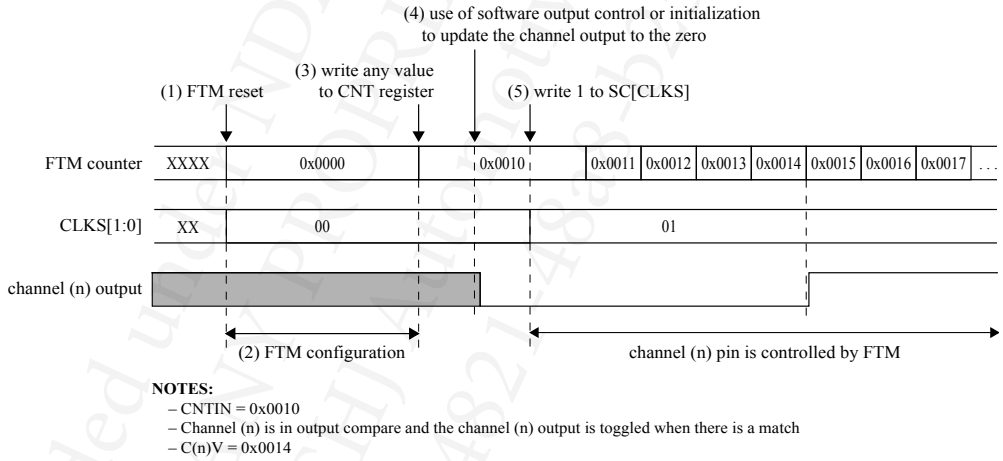


Figure 349. FTM behavior after reset when the channel (n) is in Output Compare mode

43.7 FTM Interrupts

43.7.1 Timer Overflow Interrupt

The timer overflow interrupt is generated when (TOIE = 1) and (TOF = 1).

43.7.2 Reload Point Interrupt

The Reload Point interrupt is generated when (RIE = 1) and (RF = 1).

43.7.3 Channel (n) Interrupt

The channel (n) interrupt is generated when (CHIE = 1) and (CHF = 1).

43.8 Initialization Procedure

The following initialization procedure is recommended to configure the FlexTimer. This procedure can also be used to do a new configuration.

1. Define the POL bits.
2. Mask the channels outputs using SYNCHOM = 0. Two clocks after the write to OUTMASK, the channels outputs are in the safe value.
3. (Re)Configuration FTM counter and channels to generation of periodic signals:
    a. Disable the clock. Disable the Quadrature Decoder mode.
    b. Examples of (re)configuration:
    - Write to MOD
    - Write to CNTIN
    - Configure the channels that will be used
    - Write to CnV for the channels in output modes
    - (Re)Configure deadtime
    - Do not use the SWOC without SW synchronization (see item 6)
    - Do not use the Inverting without SW synchronization (see item 6)
    - Do not use the Initialization
    - Do not change the polarity control
    - Do not configure the HW synchronization
4. Write any value to CNT. The FTM Counter is reset and the channels outputs are updated according to new configuration.
5. Enable the clock. Write to CLKS[1:0] bits a value different from zero. Enable the Quadrature Decoder mode (if it is desired).
6. Configure the SW synchronization for SWOC (if it is necessary), Inverting (if it is necessary) and Output Mask (always)
    a. Select synchronization for Output Mask
        - Write to SYNC (SWSYNC = 0, TRIG2 = 0, TRIG1 = 0, TRIG0 = 0, SYNCHOM = 1, REINIT = 0, CNTMAX = 0, CNTMIN = 0)
    b. Write to SYNCONF
        - HW Synchronization can not be enabled (HWSOC = 0, HWINVC = 0, HWOM = 0, HWWRBUF = 0, HWRSTCNT = 0, HWTRIGMODE = 0)
        - SW Synchronization for SWOC (if it is necessary): SWSOC = [0/1] and SWOC = [0/1]
        - SW Synchronization for Inverting (if it is necessary): SWINVC = [0/1] and INVC = [0/1]
        - SW Synchronization for SWOM (always): SWOM = 1
        - No enable the SW Synchronization for write buffers (because the writes to registers with write buffer are done using CLKS[1:0] = 2’b00): SWWRBUF = 0 and CNTINC = 0
        - SW Synchronization for counter reset (always): SWRSTCNT = 1
        - Enhanced synchronization (always): SYNCMODE = 1

c. If the SWOC is used (SWSOC = 1 and SWOC = 1), then write to SWOCTRL register.

d. If the Inverting is used (SWINVC = 1 and INVC = 1), then write to INVCTRL register.

e. Write to OUTMASK to enable the masked channels.

7. Generate the Software Trigger

   - Write to SYNC (SWSYNC = 1, TRIG2 = 0, TRIG1 = 0, TRIG0 = 0, SYNCHOM = 1, REINIT = 0, CNTMAX = 0, CNTMIN = 0)

8. Configure PWMEN bits (External Control of Channels Output).

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 44
Real Time Clock (RTC)

44.1 Chip-specific RTC information

44.1.1 RTC clocking

SIRC is the only clock available to the RTC module during Standby mode. For SIRC specification, see the SIRC section in the Data Sheet of chip.

44.2 Introduction

The Real-Time Clock (RTC) is a free-running counter used for time keeping applications. The RTC can be configured to generate an interrupt at a pre-defined interval. The RTC also supports an API function used to generate a periodic wakeup request to exit a low-power mode or an interrupt request.

44.2.1 Features

RTC features include:

- 32-bit counter
- Selectable counter clock sources (IRCs and OSCs)
  - Clock source 0
  - Clock source 1
  - Clock source 2
  - Clock source 3
- Optional 512 prescaler and optional 32 prescaler to run the 32 bit counter.
- RTC interrupt with interrupt enable.
- Counter runs in all modes of operation.
- RTC counter is reset when the counter is disabled by software and by reset to RTC block.
- Autonomous periodic interrupt support includes:
  - 32-bit compare value to support range of wakeup intervals/interrupts
  - API logic has a separate enable to support changing compare value while RTC is running
  - API interrupt with interrupt enable
  - Operates in all modes
  - API compare value can be modified while RTC is running
- Optional interrupt for RTC match, API match, and RTC rollover.

44.2.2 Block diagram

The following figure shows Clock gating for RTC clocks.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-4818-1251-390d3ab6b4c1

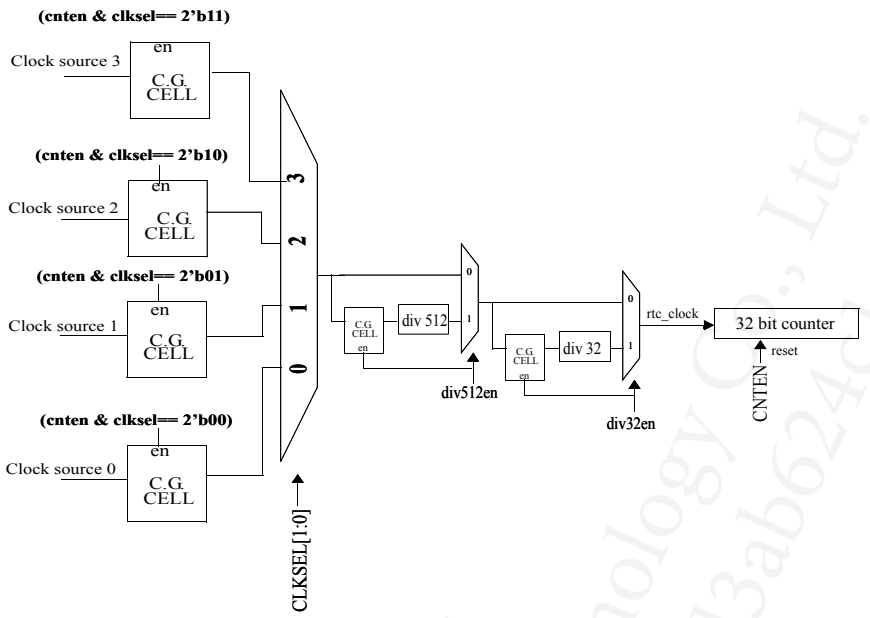


Figure 350. Clock gating for RTC clocks

The following figure shows the block diagram of RTC.

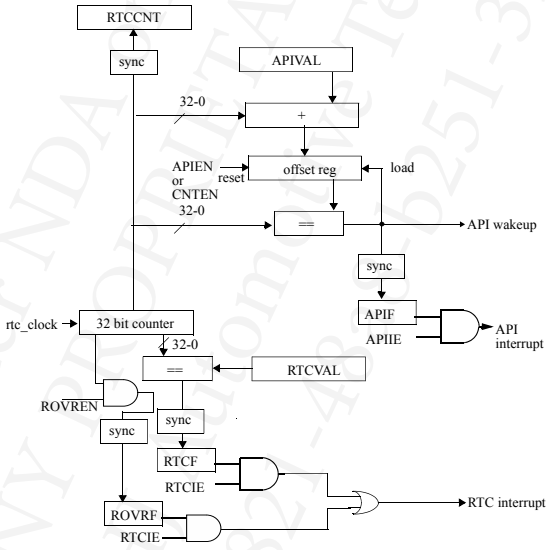


Figure 351. Block diagram

44.2.3 Modes of operation

44.2.3.1 Functional mode

There are two functional modes of operation for RTC: normal operation and low-power mode. In normal operation, all RTC registers can be read or written . The RTC/API and associated interrupts are optionally enabled. In low-power mode, the bus interface is disabled. The RTC/API is enabled (if enabled prior to entry into low-power mode).

44.2.3.2 Debug mode

On entering into the debug mode, the RTC counter freezes on the last valid count if the FRZEN is set. On exit from debug mode, counter continues from the frozen value.

44.3 RTC register descriptions

The RTC registers are listed in this section.

NOTE

Address offset - 0x18h should not be accessed by application as corresponding feature/s are not available. Therefore, transfer error will not be generated at this offset.

NOTE

XFR error will be generated when RTCSUPV is accessed in user mode, any other register is accessed in user mode when SUPV bit is set, write attempt is made for RTCCNT register and any register accessed out of address range.

44.3.1 RTC memory map

RTC base address: 4006_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	RTC Supervisor control register (RTCSUPV)	32	RW	8000_0000h
4h	RTC Control register (RTCC)	32	RW	0000_0000h
8h	RTC Status register (RTCS)	32	W1C	0000_0000h
Ch	RTC Counter register (RTCCNT)	32	RO	0000_0000h
10h	API Compare value register (APIVAL)	32	RW	0000_0000h
14h	RTC Compare value register (RTCVAL)	32	RW	0000_0000h

44.3.2 RTC Supervisor control register (RTCSUPV)

Offset

Register	Offset
RTCSUPV	0h

Function

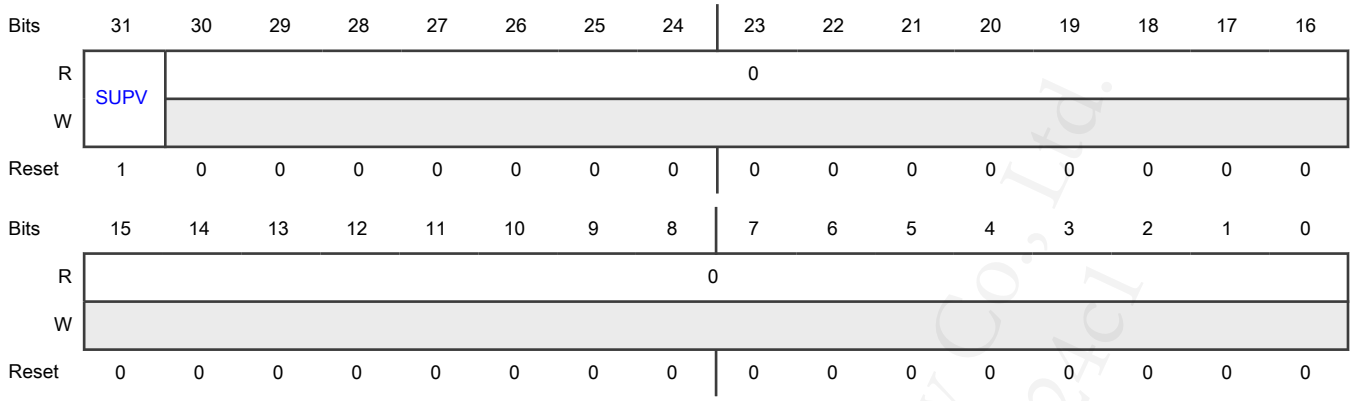
The RTCSUPV register contains the SUPV bit that determines whether other registers are accessible in supervisor mode or user mode.

NOTE

You can access this register only in Supervisor mode, and you must write a value only to the SUPV field of the register.

Provided under NDA only
COMPANY CONFIDENTIAL
Beijing CHJ Automotive Technology Co., Ltd.
c8462e54-821-48a8-b251-390d3ab04c1

Diagram



Fields

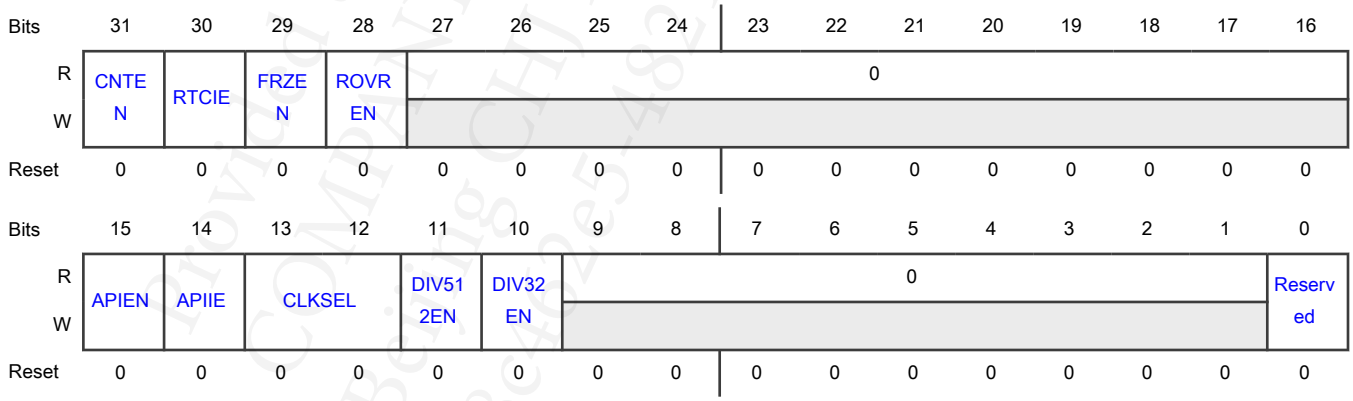
Field	Function
31 SUPV	RTC Supervisor Bit. 0b - All registers are accessible in both user as well as supervisor mode. 1b - All other registers are accessible in the supervisor mode only.
30-0 —	Reserved

44.3.3 RTC Control register (RTCC)

Offset

Register	Offset
RTCC	4h

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8a2e548-21-48a8-b251-30d3ab624c1

Fields

Field	Function
31 CNTEN	Counter Enable The CNTEN bit enables the RTC counter. Setting CNTEN bit to 0b has the effect of asynchronously resetting (synchronous reset negation) all the RTC and API logic. This allows RTC configuration and clock source selection to be updated without causing synchronization issues. **NOTE** CNTEN should be disabled when INV_RTC, INV_API are cleared. 0b - Counter disabled 1b - Counter enabled
30 RTCIE	RTC Interrupt Enable. The RTCIE bit enables interrupts requests to the system if RTCF is asserted. 0b - RTC interrupts disabled. 1b - RTC interrupts enabled.
29 FRZEN	Freeze Enable Bit The counter freezes on entering the debug mode on the last valid count value if the FRZEN bit is set. After passing of the debug mode counter starts from the frozen value. This bit should not be changed when debug mode is enabled. 0b - Counter does not freeze in debug mode. 1b - Counter freezes in debug mode.
28 ROVREN	Counter Roll Over Interrupt Enable. The ROVREN bit enables interrupt requests when the RTC has rolled over from 0xFFFF_FFFF to 0x0000_0000. The RTCIE bit must also be set in order to generate an interrupt from a counter rollover. 0b - RTC rollover interrupt disabled 1b - RTC rollover interrupt enabled.
27-16 —	Reserved
15 APIEN	Autonomous Periodic Interrupt Enable. The APIEN bit enables the autonomous periodic interrupt function. Setting this bit to 0b, asynchronously disables API wakeup output of RTC as well. 0b - API disabled. 1b - API enabled.
14 APIIE	API Interrupt Enable. The APIIE bit enables interrupts requests to the system if APIF is asserted. 0b - API interrupts disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - API interrupts enabled.
13-12 CLKSEL	Clock select The CLKSEL[1:0] bits select the clock source for the RTC. CLKSEL may only be updated when CNTEN is 0. The user should ensure that oscillator is enabled before selecting it as a clock source for RTC. **NOTE** Refer the RTC clocking section in the Clocking chapter of this document. 00b - Clock source 0 01b - Clock source 1 10b - Clock source 2 11b - Clock source 3
11 DIV512EN	Divide by 512 enable The DIV512EN bit enables the 512 clock divider. DIV512EN may only be updated when CNTEN is 0. 0b - Divide by 512 is disabled. 1b - Divide by 512 is enabled.
10 DIV32EN	Divide by 32 enable. The DIV32EN bit enables the 32 clock divider. DIV32EN may only be updated when CNTEN is 0. 0b - Divide by 32 is disabled 1b - Divide by 32 is enabled
9-1 —	Reserved
0 —	Reserved

44.3.4 RTC Status register (RTCS)

Offset

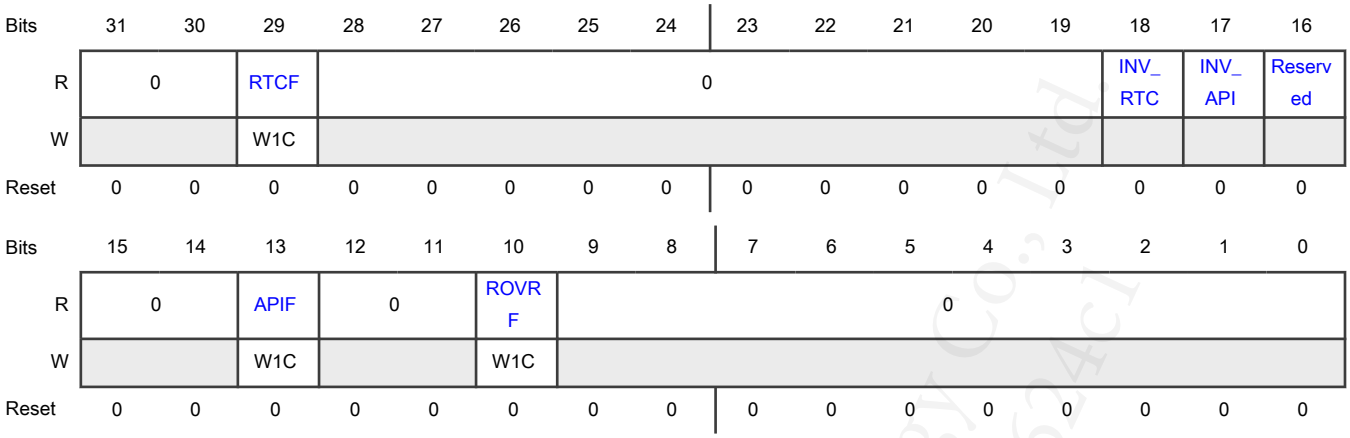
Register	Offset
RTCS	8h

Function

NOTE

W1C has priority over setting of the RTCF, APIF and ROVRF bits, in case both clearing and setting occurs at the same time.

Diagram



Fields

Field	Function
31-30 —	Reserved
29 RTCF	RTC Interrupt Flag The RTCF bit indicates that the RTC counter has reached the counter value matching RTC Compare value register (RTCVAL). RTCF is cleared by writing a 1 to RTCF. Writing a 0 to RTCF has no effect. 0b - RTC counter is not equal to RTCVAL. 1b - RTC counter matches RTCVAL.
28-19 —	Reserved
18 INV_RTC	Invalid RTC write This bit returns value 1 after a value is written to the RTCVAL register and the synchronization process is in progress. During this synchronization period, any attempt to write to the RTCVAL register again is ignored. Synchronization will complete only when CNTEN is set.
17 INV_API	Invalid APIVAL write This bit returns value 1 after a value is written to the APIVAL register and the synchronization process is in progress. During this synchronization period, any attempt to write to the APIVAL register again is ignored. Synchronization will complete only when CNTEN is set.
16 —	Reserved
15-14 —	Reserved
13	API Interrupt Flag.

Table continues on the next page...

Table continued from the previous page...

Field	Function
APIF	The APIF bit indicates that the RTC counter has reached the counter value matching API offset value. APIF is cleared by writing a 1 to APIF. Writing a 0 to APIF has no effect. 0b - Counter is not equal to API offset value. 1b - Counter matches the API offset value.
12-11 —	Reserved
10 ROVRF	Counter Roll Over Interrupt Flag. The ROVRF bit indicates that the RTC has rolled over from 0xFFFF_FFFF to 0x0000_0000. ROVRF is cleared by writing a 1 to ROVRF. 0b - RTC has not rolled over 1b - RTC has rolled over
9-0 —	Reserved

44.3.5 RTC Counter register (RTCCNT)

Offset

Register	Offset
RTCCNT	Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RTCCNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RTCCNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c467e54821-48a8-b251-590d3ab624c1

Fields

Field	Function
31-0 RTCCNT	RTC Counter Value Because of clock synchronization, the RTCCNT value may represent a previous counter value.

44.3.6 API Compare value register (APIVAL)

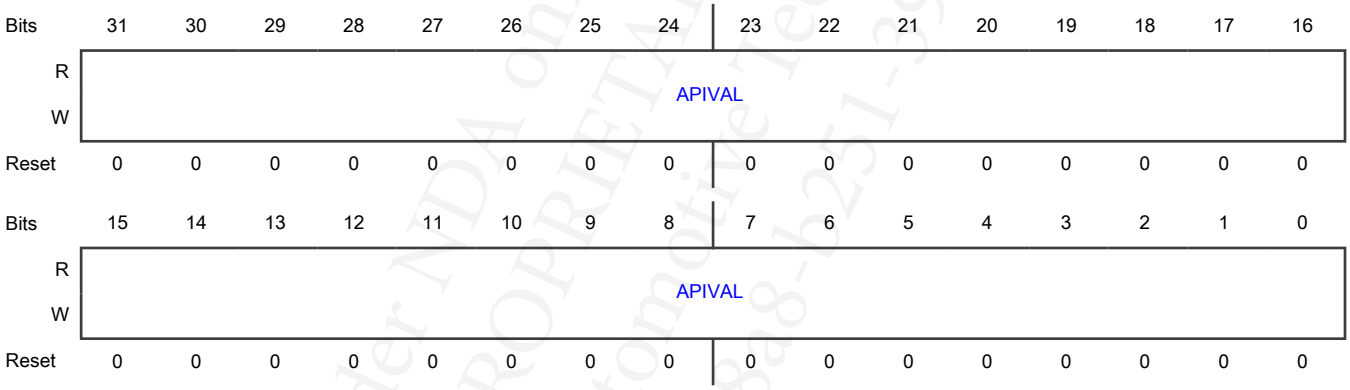
Offset

Register	Offset
APIVAL	10h

Function

The APIVAL offset bits are compared to the RTC counter bits and if a match occurs, an interrupt/wakeup request is asserted.

Diagram



Fields

Field	Function
31-0 APIVAL	API Compare Value. APIVAL bits are added to the current count to calculate an offset. The APIVAL offset bits are compared to the RTC counter bits and if a match occurs, an interrupt/wakeup request is asserted. **NOTE** API functionality is active only when APIVAL is non zero. The first API interrupt takes two more cycles because of synchronization of APIVAL to RTC clock. After that, interrupts are periodic in nature and it takes APIVAL+1 cycles. The Minimum supported value of APIVAL is 4. This is because of synchronization.

Provided under NDA only
COMPANY PROPRIETARY
Beijing eSOL Automotive Technology Co., Ltd.
c8a5e5e1-5482-4848-b251-590c3ab624c1

44.3.7 RTC Compare value register (RTCVAL)

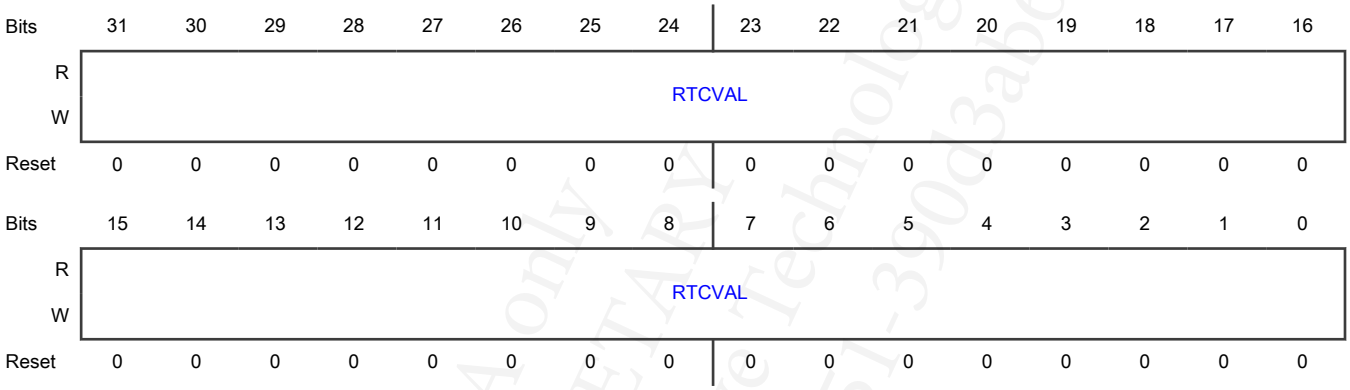
Offset

Register	Offset
RTCVAL	14h

Function

The RTCVAL bits are compared to the RTC counter bits and if a match occurs, RTCF is set. The minimum value of RTCVAL should be 4.

Diagram



Fields

Field	Function
31-0 RTCVAL	RTC Compare Value. The RTCVAL bits are compared to the RTC counter bits and if a match occurs, RTCF is set.

44.4 Functional description

44.4.1 RTC

The RTC consists of a 32-bit free running counter enabled with the CNTEN bit (CNTEN, when negated, asynchronously resets the counter and synchronously enables the counter when enabled). After disabling CNTEN, RTCVAL, APIVAL, needs to be written again for desired functionality. The value of the counter may be read via the RTC Counter register (RTCCNT) register. Note that because of clock synchronization, the RTC Counter register (RTCCNT) value may represent a previous counter value. The difference between the counter and the read value depends on the ratio of counter clock and bus clock. Maximum possible difference between the two is 6 count values.

The clock source to the counter is selected with the CLKSEL field, which gives four options for clocking the RTC/API. The four clock sources are assumed to be on these:Clock source 0, Clock source 1, Clock source 2, and Clock source 3. The output of the clock mux can be optionally divided by a combination of 512 and 32 to give various count periods for different clock sources. Note that the CNTEN bit should be disabled when the RTC/API clock source is switched.

When the RTC Counter register (RTCCNT) counter value for counter bits 31-0 match the 32-bit value in the RTCVAL field, then the RTCF interrupt flag bit is set (after proper clock synchronization). If the RTCIE interrupt enable bit is set, then the RTC interrupt request is generated. RTC Compare value register (RTCVAL) register can be written only when INV_RTC bit is clear. Initially

INV_RTC=0, and hence RTC Compare value register (RTCVAL) can be written once and hence INV_RTC gets set. This bit can now be cleared only by enabling the RTC counter. After the counter is enabled, RTC Compare value register (RTCVAL) can be written anytime, until RTC is disabled again. RTC Compare value register (RTCVAL) is first synchronized to the RTC clock domain, therefore, if RTC Compare value register (RTCVAL) is updated at the point where a counter match is due to happen in the next 2-3 RTC clocks because of previous RTC Compare value register (RTCVAL), the RTCF flag is set. However, if the RTC Compare value register (RTCVAL) is updated at the point where no counter match is due as per the previous RTC Compare value register (RTCVAL), the RTCF flag is set when the counter matches the new RTC Compare value register (RTCVAL).

If RTC Compare value register (RTCVAL) is updated just after counter match, new RTC Compare value register (RTCVAL) value should not be within next 6 RTC counter values.

A rollover interrupt can be generated when the RTC transitions from a count of 0xFFFF_FFFF to 0x0000_0000. The rollover flag is enabled by setting the ROVREN bit. If RTCIE is enabled, an interrupt request is generated for an RTC counter rollover.

All the flags and counter values are synchronized with the Bus clock. It is assumed that the Bus clock and RTC clock selected through CLKSEL follows the below relation:

Bus clock >= (1.5 * RTC clock)/(div_factor)

- if both DIV32EN and DIV512EN bits are disabled, div_factor = 1
- if DIV32EN=1 and DIV512EN=0, div_factor = 32
- if DIV32EN=0 and DIV512EN=1, div_factor = 512
- if both DIV32EN and DIV512EN bits are enabled, div_factor = 512*32 = 16384

In case, RTC wakeup's are used as a wakeup source, Bus clock should be disabled after enabling the required wakeup and ensuring sufficient time gap (few Bus clock or RTC clock cycles, whichever is slower) between Bus clock disabling and wakeup event.

44.4.2 API functional description

Setting APIEN bit enables the autonomous interrupt function. The 32-bit APIVAL field selects the time interval for triggering an interrupt and/or wakeup event. Because the RTC is a free-running counter, the APIVAL is added to the current count to calculate an offset. When the counter reaches the offset count, an interrupt and/or wakeup request is generated. Then the offset value is recalculated and again re-triggers a new request when the new value is reached. API function is enabled only when CNTEN and APIEN bits are asserted and APIVAL is non-zero. Also, APIVAL can be updated anytime. After APIVAL is updated, the first API interrupt is generated according to the previous value. From the second interrupt onwards, the API interrupt is generated with the new APIVAL. When a compare is reached, the APIF interrupt flag bit is set (after proper clock synchronization). If the APIIE interrupt enable bit is set, then the API interrupt request is generated. If there is a match while being in the low-power mode, then the API first generates a wakeup request to force a wakeup into normal operation, and then the APIF flag is set.

When the CNTEN is de-asserted, the API function is reset , though wakeup API is not asynchronously de-asserted with CNTEN. If APIEN is disabled when counter matches the offset, APIF can be missed.

44.5 Recommended programming flow

1. Program RTCVAL register with the value greater than 4 if RTCF related functionality is required.
2. Program APIVAL register with the value greater than 4 if API functionality is required.
3. Program all fields (as required) of RTC Control register (RTCC) register with CNTEN=1. After setting CNTEN field, counter starts running and RTC/API functionality will be active as per the configurations.
4. If RTCVAL or APIVAL needs to be updated during run, check corresponding INV_RTC or INV_API bit is cleared before writing and do meet all the restrictions of the corresponding register.
5. If CLKSEL or DIV32EN or DIV512EN needs to be updated, first update RTC Control register (RTCC) register with CNTEN=0 (when all INV_RTC or INV_API are 0) keeping other field values same. Wait for minimum 3 RTC clock cycles so that CNTEN will be synchronized to RTC clock domain. Then write the RTC Control register (RTCC) register with new configuration required with CNTEN=1.

44.6 Glossary

API	Autonomous Periodic Interrupt
IRCs	Internal RC Oscillator
OSCs	Oscillator

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 45
Inter-Integrated Circuit (I2C)

45.1 Overview

This chapter describes the Inter-Integrated Circuit (I^2C) module implemented on this chip and presents the following topics:

- Introduction to I2C
- External signal descriptions
- I2C register descriptions
- Functional description
- Initialization/application information

45.2 Introduction to I^2C

This section presents the following topics:

- Definition: I2C module
- Module block diagram
- Features
- Modes of operation
- Definition: I2C conditions

45.2.1 Definition: I^2C module

The I^2C module is a functional unit that provides a two-wire—serial data (SDA) and serial clock (SCL)—bidirectional serial bus. This bus provides a simple and efficient method of data exchange between this chip and other devices, such as microcontrollers, EEPROMs, real-time clock devices, analog-to-digital converters, and LCDs.

45.2.2 Module block diagram

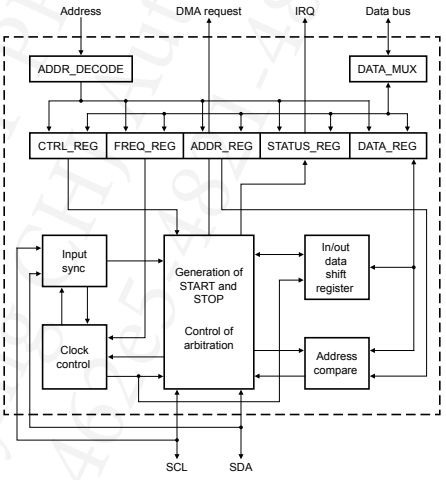


Figure 352. I^2C block diagram

Provided under NDA only
COMPANY PROPRIETARY
Beijing Huatom Automotive Technology Co., Ltd.
c8c4b5e4-92f7-4a8a-8951-390d3ab624c1

45.2.3 Features

The I²C module has the following key features:

- Compatible with I²C bus standard[2]
- Operating speeds
  - Up to 100 kbps in the Standard Mode defined by the I²C protocol specification
  - Up to 400 kbps in the Fast Mode defined by the I²C protocol specification
  - Operation at higher baud rates (up to a maximum of module clock/20) with reduced bus loading
  - Actual baud rate dependent on the SCL rise time (which depends on external pullup resistor values and bus loading)
- Multi-master operation
- Software-programmable for one of 256 different serial clock frequencies
- Software-selectable acknowledge bit
- Interrupt-driven byte-by-byte data transfer
- Arbitration lost interrupt with automatic mode switching from master to slave
- Calling address identification interrupt
- Start and stop signal generation/detection
- Repeated start signal generation
- Acknowledge bit generation/detection
- Bus busy detection
- DMA interface

45.2.4 Modes of operation

The I²C module supports the chip modes described in Table 286.

Table 286. Chip modes supported by the I²C module

Chip mode	Description	Important notes
RUN	Basic mode of operation	—
DEBUG	Allows the chip to freeze all ongoing activities (such as an ongoing transaction, counter values, and register status) for debugging	See DEBUG mode.

In addition to chip modes, the I²C module has several module-specific modes. These are described in Table 287.

Table 287. Module-specific modes supported by the I²C module

Module mode	Description	Important notes
Master mode	The I²C module is the driver of the SDA line.	• Do not use the I²C module's slave address as a calling address.

Table continues on the next page...

[2] Compliant with I²C 2.0 standard with the exception that HS (high speed) mode is not supported

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c84025c4-8214-48a8-b251-390d3ab624c1

Table 287. Module-specific modes supported by the I²C module (continued)

Module mode	Description	Important notes
		• The I²C module cannot be a master and a slave simultaneously.
Slave mode	The I²C module is not the driver of the SDA line.	• Enable the I²C module before a START condition from a non-I²C master is detected. • By default the I²C module performs as a slave receiver.

45.2.5 Definition: I²C conditions

The following table shows the I²C-specific conditions defined for the I²C module.

Table 288. I²C conditions

Condition	Description
START	A condition that denotes the beginning of a new data transfer and awakens all slaves. Each data transfer contains several bytes of data. It is defined as a high-to-low transition of SDA while SCL is high, as shown in the following figure.
STOP	A condition generated by the master to terminate a transfer and free the bus. It is defined as a low-to-high transition of SDA while SCL is high, as shown in the following figure.
Repeated START	A START condition that is generated without a STOP condition to terminate the previous transfer.

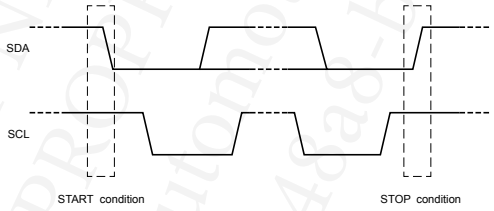


Figure 353. START and STOP conditions

45.3 External signal descriptions

This section presents the following topics:

- Signal overview
- Detailed external signal descriptions

45.3.1 Signal overview

The I²C module uses the Serial Data (SDA) and Serial Clock (SCL) signals as a communication interconnect with other devices. The signal patterns driven on the SDA signal represent address, data, or read/write information at different stages of the protocol.

All devices connected to the SDA and SCL signals must have open-drain or open-collector outputs. The I²C bus connects the SDA lines of all devices together, thus performing a logical AND of the SDA signals. Similarly, the I²C bus connects all the SCL lines together, thus performing a logical AND of the SCL signals. The combined SDA and SCL lines also connect to their respective pullup resistors. For the electrical characteristics of these signals, see the data sheet for this chip.

45.3.2 Detailed external signal descriptions

The SDA and SCL signals are described in the following table.

Table 289. External signal descriptions

Signal	Description
SCL	Bidirectional serial clock line of the module, compatible with the I^2C bus specification
SDA	Bidirectional serial data line of the module, compatible with the I^2C bus specification

45.4 I2C register descriptions

The memory map for the I^2C module is given below. The total address for each register is the sum of the base address for the I^2C module and the address offset for each register.

Address location 0x0007 is a reserved location, but access to this location will not generate any bus error.

All registers are accessible via 8-bit, 16-bit, or 32-bit accesses. However, 16-bit accesses must align to 16-bit boundaries, and 32-bit accesses must align to 32-bit boundaries.

As an example, the IBFD is at address offset 0x01, and you can access it in any of the following ways:

- 8-bit R/W access of address offset 0x0001
- Second byte of 16-bit R/W access of address offset 0x0000
- Second byte of 32-bit R/W access of address offset 0x0000

You cannot access the IBFD register by a 16- or 32-bit RW operation at address offset 0x0001 because those operations require an address aligned to a 16- or 32-bit boundary.

45.4.1 I2C memory map

I2C_0 base address: 401E_4000h

I2C_1 base address: 401E_8000h

I2C_2 base address: 401E_C000h

I2C_3 base address: 402D_8000h

I2C_4 base address: 402D_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	I2C Bus Address Register (IBAD)	8	RW	00h
1h	I2C Bus Frequency Divider Register (IBFD)	8	RW	00h
2h	I2C Bus Control Register (IBCR)	8	RW	80h
3h	I2C Bus Status Register (IBSR)	8	W1C	80h
4h	I2C Bus Data I/O Register (IBDR)	8	RW	00h
5h	I2C Bus Interrupt Config Register (IBIC)	8	RW	00h
6h	I2C Bus Debug Register (IBDBG)	8	RW	00h

45.4.2 I2C Bus Address Register (IBAD)

Offset

Register	Offset
IBAD	0h

Function

This register contains the address that the I^2C bus responds to when addressed as a slave. This is not the address sent on the bus during the address transfer.

Access: Supervisor mode only (the module ignores user mode accesses and does not assert an error response if you access this register in user mode)

Diagram

Bits	7	6	5	4	3	2	1	0
R	ADR							0
W								
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-1 ADR	Slave address Specific slave address to be used by the I^2C module. **NOTE** The default mode of I^2C bus is slave mode for an address match on the bus.
0 —	Reserved

45.4.3 I2C Bus Frequency Divider Register (IBFD)

Offset

Register	Offset
IBFD	1h

Function

Access: Supervisor mode only (the module ignores user mode accesses and does not assert an error response if you access this register in user mode)

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cac462e5-4c21-48c8-b251-390d5ab624d1

Diagram

Bits	7	6	5	4	3	2	1	0
R W	IBC							
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-0 IBC	IBC I-Bus clock rate. This field is used to prescale the bus clock for bit rate selection. See Clock rate and IBFD settings.

45.4.4 I2C Bus Control Register (IBCR)

Offset

Register	Offset
IBCR	2h

Function

Access: Supervisor mode only (the module ignores user mode accesses and does not assert an error response if you access this register in user mode)

Diagram

Bits	7	6	5	4	3	2	1	0
R	MDIS	IBIE	MSSL	TXRX	NOACK	0	DMAEN	Reserved
W						RSTA		
Reset	1	0	0	0	0	0	0	0

Fields

Field	Function
7 MDIS	Module disable

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4ca8-b251-390d3ab624d1

Table continued from the previous page...

Field	Function
	This field controls the software reset of the entire I^2C module. **NOTE** If the I^2C module is enabled in the middle of a byte transfer, the interface behaves as follows: slave mode ignores the current transfer on the bus and starts operating whenever a subsequent START condition is detected. Master mode will not be aware that the bus is busy, hence if a start cycle is initiated, the current bus cycle may become corrupt. This ultimately results in either the current bus master or the I^2C module losing arbitration, after which bus operation returns to normal. 0b - The module is enabled. You must clear this field before any other IBCR fields have any effect. 1b - The module is reset and disabled. This is the power-on reset situation. When high, the interface is held in reset, but registers can still be accessed. Status register fields (IBSR) are not valid when the module is disabled.
6 IBIE	I2C bus interrupt enable 0b - Interrupts from the I2C module are disabled. This does not clear any currently pending interrupt condition. 1b - Interrupts from the I2C module are enabled. An I2C interrupt occurs if IBSR[IBIF]=1.
5 MSSL	Master/slave mode select When you change this field from 0 to 1, the module generates a START signal on the bus and selects the master mode. When you change this field from 1 to 0, the module generates a STOP signal and changes the operation mode from master to slave. You should generate a STOP signal only if IBSR[IBIF]=1. The module clears this field without generating a STOP signal when the master loses arbitration. 0b - Slave mode 1b - Master mode
4 TXRX	Transmit/receive mode select This field selects the direction of master and slave transfers. When the I^2C module is addressed as a slave, your software must program this field based on IBSR[SRW]: • When IBSR[SRW]=0, program TXRX=0. • When IBSR[SRW]=1, program TXRX=1. In master mode, your software must program this field according to the type of transfer required. Therefore, for address cycles, this field will always be 1. 0b - Receive 1b - Transmit
3 NOACK	Data acknowledge disable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field specifies the value driven onto SDA during data acknowledge cycles for both master and slave receivers. When the I²C module is enabled, it always acknowledges address matches, regardless of the value of NOACK. **NOTE** Values written to this field are only used when the I²C module is a receiver, not a transmitter. 0b - The module sends an acknowledge signal to the bus at the 9th clock bit after receiving one byte of data. 1b - The module does not send an acknowledge-signal response (that is, acknowledge bit = 1).
2 RSTA	Repeat START If the I²C module is the current bus master, and you program RSTA=1, the I²C module generates a repeated START condition. This field always reads as a 0. If you attempt a repeated START at the wrong time—if the bus is owned by another master—the result is loss of arbitration. 0b - No effect 1b - Generate repeated START condition
1 DMAEN	DMA enable When this field is 1, and the I²C module requires data to be read or written to I2C Bus Data I/O Register (IBDR), the module asserts the DMA Tx and Rx lines. When DMAEN=1, the I²C module does not generate an interrupt after the transfer of a byte is complete (that is, when the I²C module detects the falling edge of the 9th clock cycle). However, the module generates an interrupt if arbitration is lost or the module is addressed as a slave. The DMA mode is only valid when the I²C module is configured as a master, and the DMA transfer still requires CPU intervention at the start and the end of each frame of data. See DMA application information for more details. 0b - Disable the DMA TX/RX request signals 1b - Enable the DMA TX/RX request signals
0 —	Reserved Although this field supports read/write access, writing to this field is not required, and can produce unexpected results.

45.4.5 I2C Bus Status Register (IBSR)

Offset

Register	Offset
IBSR	3h

Function

Access: Supervisor mode only (the module ignores user mode accesses and does not assert an error response if you access this register in user mode)

Diagram

Bits	7	6	5	4	3	2	1	0
R	TCF	IAAS	IBB	IBAL	0	SRW	IBIF	RXAK
W				W1C			W1C	
Reset	1	0	0	0	0	0	0	0

Fields

Field	Function
7 TCF	Transfer complete The I2C module programs this field to 0 during the time that the module transfers one byte of data. After the I2C module detects the falling edge of the 9th clock of a byte transfer, the module programs this field to 1. **NOTE** This field is relevant to you only during or immediately after a transfer to or from the I2C module. 0b - Transfer in progress 1b - Transfer complete
6 IAAS	Addressed as a slave When its own specific address (I-Bus Address Register) is matched with the calling address, this field is set. The CPU is interrupted if IBCR[IBIE]=1. Then the CPU needs to check SRW and set the TXRX field accordingly. Writing to the I-Bus Control Register clears IAAS. 0b - Not addressed 1b - Addressed as a slave
5 IBB	Bus busy This field indicates the status of the bus. When a START signal is detected, IBB is set. If a STOP signal is detected, IBB is cleared and the bus enters idle state. **NOTE** Software must ensure that the I2C bus is idle by checking the IBSR[IBB] field (bus busy) before switching to master mode and attempting a START cycle. 0b - Bus is idle 1b - Bus is busy
4 IBAL	Arbitration Lost The I2C module programs IBAL=1 when the arbitration procedure is lost. Arbitration is lost in the following circumstances: • SDA is sampled low when the master drives a high during an address or data transmit cycle. • SDA is sampled low when the master drives a high during the acknowledge bit of a data receive cycle.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• A start cycle is attempted when the bus is busy. • A repeated start cycle is requested in slave mode. • A stop condition is detected when the master did not request it. Your software must clear this field by writing a one to it. A write of zero has no effect.
3 —	Reserved
2 SRW	Slave Read/Write When IAAS=1, SRW indicates the value of the R/W command bit of the calling address sent from the master. SRW is valid only when all of these are true: • The I-Bus is in slave mode. • A complete address transfer has occurred with an address match. • No other transfers have been initiated. By reading this field, the CPU can detect slave transmit/receive mode according to the command of the master. 0b - Slave receive, master writing to slave 1b - Slave transmit, master reading from slave
1 IBIF	I2C bus interrupt flag IBIF=1 when one of the following conditions occurs: • Arbitration lost (IBAL=1) • Byte transfer complete (TCF=1 and DMAEN=0) • Addressed as slave (IAAS=1) • NoAck from Slave (MS and Tx bits set) • I^2C Bus going idle (IBB high-low transition and enabled by BIIE) A processor interrupt request will be caused if IBIE=1. Your software must clear IBIF by writing a one to it. A write of zero has no effect. In DMA mode (DMAEN set) a byte transfer complete condition will not trigger the setting of IBIF. All other conditions still apply.
0 RXAK	Received acknowledge This is the value of SDA during the acknowledge bit of a bus cycle. If RXAK=0, it indicates an acknowledge signal has been received after the completion of eight bits of data transmission on the bus. If RXAK=1, it means no acknowledge signal is detected at the ninth clock. RXAK is valid only after transfer is complete. 0b - Acknowledge received 1b - No acknowledge received

Preliminary NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8a02e-4e1-48a8-b251-390d3a1624c1

45.4.6 I2C Bus Data I/O Register (IBDR)

Offset

Register	Offset
IBDR	4h

Function

In master transmit mode, when your software writes to IBDR, the I^2C module initiates a data transfer. The module sends the most significant bit first.

In master receive mode, when your software reads IBDR, the I^2C module initiates the reception of the next data byte.

In slave mode, the same functions are available after an address match has occurred.

NOTE

The IBCR[TXRX] field must correctly reflect the desired direction of transfer in master and slave modes for the transmission to begin. For instance, if the I^2C is configured for master transmit but a master receive is desired, then reading the IBDR will not initiate the receive.

When the the I^2C module is configured for master-receive or slave-receive modes, and your software reads IBDR, the I^2C module returns the most-recent byte received. The IBDR does not reflect every byte transmitted on the I^2C bus. Your software also cannot verify that it wrote a byte to IBDR correctly by reading the byte back.

In master transmit mode, the I^2C module takes the first byte of data written to IBDR after assertion of MSSL and uses this byte for the address transfer. This byte must comprise the calling address (in position DATA[7:1]) concatenated with the required R/W_b bit (in position D0). (In the I^2C bus specification, the R/W_b bit is presented as R/W with an overbar on the W.)

Access: Supervisor mode only (the module ignores user mode accesses and does not assert an error response if you access this register in user mode)

Diagram

Bits	7	6	5	4	3	2	1	0
R W	DATA							
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-0 DATA	DATA Data transmitted or received

Printed under NDA by COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-4821-4948-b2f3-30d3b6240d1

45.4.7 I2C Bus Interrupt Config Register (IBIC)

Offset

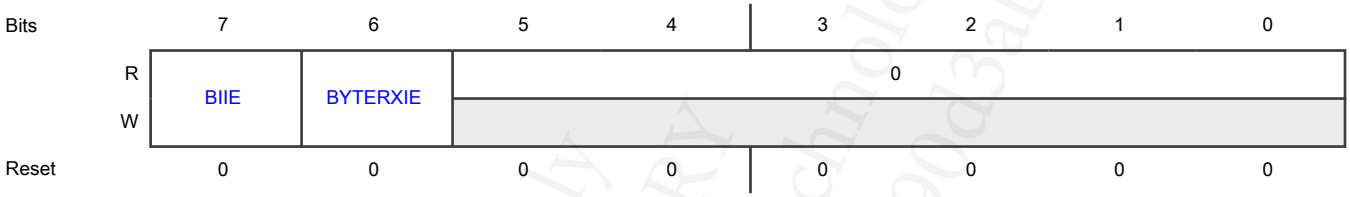
Register	Offset
IBIC	5h

Function

To program BIIE = 1, you must ensure that IBCR[MDIS] = 0.

Access: Supervisor mode only (the module ignores user mode accesses and does not assert an error response if you access this register in user mode)

Diagram



Fields

Field	Function
7 BIIE	Bus idle interrupt enable You can use this configuration field to enable the generation of an interrupt after the I^2C bus becomes idle. After BIIE=1, a high-to-low transition of IBSR[IBB] results in IBSR[IBIF]=1. You can use this feature to inform the CPU about the completion of a STOP on the I^2C bus. 0b - Bus idle interrupts disabled 1b - Bus idle interrupts enabled
6 BYTERXIE	Byte receive interrupt enable You can use this field to generate an interrupt every time the I^2C master/slave receives a new byte. This feature can be useful when an I^2C master receives data from a slave that transmits on an irregular basis. The I^2C module updates BYTERXIE only when the module is enabled (IBCR[MDIS]=0).
5-0 —	Reserved

45.4.8 I2C Bus Debug Register (IBDBG)

Offset

Register	Offset
IBDBG	6h

Function

Access: Supervisor mode only (the module ignores user mode accesses and does not assert an error response if you access this register in user mode)

Diagram

Bits	7	6	5	4	3	2	1	0
R	0				GLFLT_EN	BYTE_RX	IPG_DEB...	IPG_DEB...
W						W1C		
Reset	0	0	0	0	0	0	0	0

Fields

Field	Function
7-4 —	Reserved
3 GLFLT_EN	Glitch filter enable This field allows you to filter out any glitch on the SCL or SDA line that may cause the I^{2}C module to enter an unrecoverable state. **NOTE** When GLFLT_EN = 1, the effect of the SCL and SDA lines is delayed by six IPG clock cycles. So, in cases where a glitch is not expected at the SCL or SDA inputs, you should not program GLFLT_EN = 1. 0b - The I2C module allows all pulses on the SCL or SDA input lines to pass through. 1b - The I2C module filters out any pulse (rising or falling) on the SCL or SDA input line that is shorter than six IPG clock cycles.
2 BYTE_RX	The I2C module programs BYTE_RX=1 when an I2C master/slave receives a new byte. The module generates an interrupt if IBIC[BYTERXIE]=1.
1 IPG_DEBUG_HALTED	Debug halted This is a status field that you can read back after asserting the debug enable signal. The field informs you whether the I^{2}C module entered DEBUG mode or not. 0b - The I2C module is still executing a transaction. 1b - The I2C module entered DEBUG mode.
0 IPG_DEBUG_EN	Debug enable This field allows the I^{2}C module to enter the DEBUG mode when the IPG DEBUG signal is high. All the registers, counter values and status bits are frozen and can be accessed by the CPU.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If your software programs IPG_DEBUG_EN=1, and the IPG DEBUG signal becomes high in the middle of a transaction, the I^2C module enters DEBUG mode only after successfully completing the current transaction, after which no further transactions can occur until the I^2C module exits DEBUG mode. 0b - Normal operation. The bus idle interrupts are disabled. 1b - The I2C module is in DEBUG mode.

45.5 Functional description

This section presents the following topics:

- Notes about module operation
- Transactions
- Arbitration procedure
- Clock behavior
- Interrupts
- DEBUG mode
- DMA interface

45.5.1 Notes about module operation

- The I^2C module always performs as a slave receiver by default, unless explicitly programmed to be a master or slave transmitter.
- When the I^2C module is acting as a master, it must not try to call its own slave address.

45.5.2 Transactions

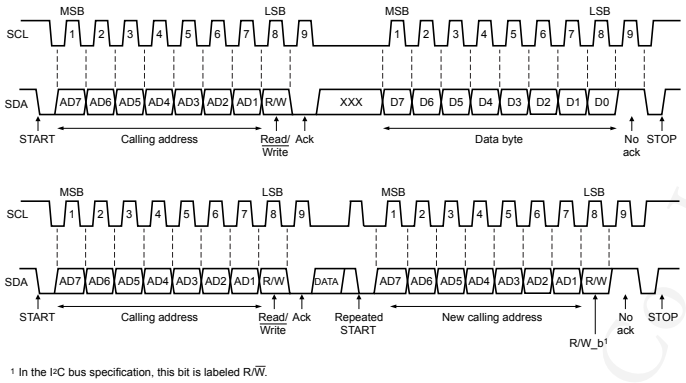
This section presents the following topics:

- Protocol overview
- Transaction protocol definitions
- High-level protocol steps
- START condition
- Slave address transmission
- Data transmission
- STOP condition
- Repeated START condition

45.5.2.1 Protocol overview

The following figure shows the behavior of SCL and SDA during a typical I^2C transaction.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-9251-390d3ab624c1



[1] In the I²C bus specification, this bit is labeled R/$\overline{W}$.

Figure 354. I²C transaction protocol

45.5.2.2 Transaction protocol definitions

This section defines several important terms presented in Figure 354.

Table 290. I²C definitions

Term	Definition
START	A START condition, as defined in Definition: I2C conditions
STOP	A STOP condition, as defined in Definition: I2C conditions
Calling (slave) address	A seven-bit address used to identify a slave on the I²C bus (see I2C calling address requirements for the requirements for specifying this address)
Read/write (R/W)	A bit that specifies the direction of the data transfer to the slave as follows: • 0=The data is being transferred from the master to the slave ("write") • 1=The data is being transferred from the slave to the master ("read")
Ack	A bit that specifies the acknowledgement of a calling address, indicated by pulling SDA low

45.5.2.3 I²C calling address requirements

The calling addresses of the devices used on an I²C network are subject to the following requirements:

- Each slave must have a unique calling address.
- A master must not transmit a calling address that is the same as its own slave address.

45.5.2.4 High-level protocol steps

The I²C protocol conceptually supports two types of transfers, as illustrated in Figure 354. The significant steps in these transfers are presented in the following table. Details of each of these steps are presented in subsequent sections.

Table 291. I²C high-level protocol steps

Standard transfer	Repeated START transfer
1. START condition	1. START condition
2. Slave target or general call address transmission	2. Slave target or general call address transmission
3. Acknowledgment from slave	3. Acknowledgment from slave
4. Data transfer	4. Data transfer
5. STOP condition	5. Repeated START condition
6. (repeat Steps 1-4)	6. (repeat Steps 2-4 as needed)
	7. STOP condition
	8. (repeat Steps 1-7)

45.5.2.5 START condition

When the bus is free (no master device engages the bus; both SCL and SDA lines are at logical high), a master can initiate communication by sending a START condition (see Definition: I2C conditions). This signal denotes the beginning of a new data transfer (which may contain several bytes of data) and brings all slaves out of their idle states.

See Clock rate and IBFD settings and I2C Bus Frequency Divider Register (IBFD) for the associated timing requirements.

45.5.2.6 Slave address transmission

The master transmits the slave address on the next clock cycle after the START condition (see START condition). The following table presents the process of slave address transmission.

Table 292. Slave address transmission process

Step	Action
1	The master transmits the seven-bit slave address.
2	The master transmits the R/W bit.
3	Each slave examines the transmitted address and compares it to its own. If the addresses match, the slave device returns the acknowledge bit on the ninth SCL clock cycle.
4	The master waits for the acknowledge bit and determines the next step as follows: • The acknowledge bit is set: The master must initiate a data transfer followed by either a STOP condition or a repeated START condition. • The acknowledge bit is cleared: The master must wait for SCL to return to logic zero.

45.5.2.7 Data transmission

A data transfer session has the following characteristics:

- Can transmit one or more bytes of data
- Awakens all slaves
- Proceeds on a byte-by-byte basis in the direction specified by the R/W bit sent by the calling master

The transmitted data is subject to the following requirements:

- Each data byte must consist of 8 bits.
- Data bits can be changed only while SCL is low and must be held stable while SCL is high.
- One data bit is transmitted during one SCL clock pulse.
- The most significant bit (msb) must be transmitted first.
- Each data byte must be followed by an acknowledge bit on the ninth SCL clock pulse.

If the slave receiver does not acknowledge the master, the slave must leave the SDA line high. The master can then generate a STOP condition (to abort the data transfer) or a repeated START condition.

If the master receiver does not acknowledge the slave transmitter after a byte of transmission, the slave interprets that as reaching the end of data. The slave then releases the SDA line for the master to generate a STOP or a START condition.

45.5.2.8 STOP condition

The master can terminate the communication by generating a STOP condition (see Definition: I2C conditions). It can do so even if the slave has generated an acknowledge, at which point the slave must release the bus.

A master is not required to send a STOP condition at the end of every transfer. For more information, see Repeated START condition.

See Clock rate and IBFD settings and I2C Bus Frequency Divider Register (IBFD) for the associated timing requirements.

45.5.2.9 Repeated START condition

The I^2C protocol also supports a repeated START condition, which can be generated without a preceding STOP condition. A master device can use this condition to communicate with another slave or with the same slave in a different mode without releasing the bus. This condition is illustrated in the second timing diagram of Figure 354.

45.5.3 Arbitration procedure

The I^2C bus is a true multi-master bus that allows more than one master to be connected to the bus. If two or more masters try to control the bus at the same time, a clock synchronization procedure determines the bus clock. For this clock, the low period is equal to the longest clock low period and the high period is equal to the shortest period among the masters. A data arbitration procedure determines the relative priority of the contending masters. A bus master loses arbitration if it transmits logic "1" while another master transmits logic "0". The losing masters immediately switch over to slave mode and stop driving the SDA output. In this case, the transition from master to slave mode does not generate a STOP condition. Meanwhile, the I^2C module sets IBSR[IBAL] to indicate loss of arbitration.

45.5.4 Clock behavior

This section presents the following topics:

- Clock synchronization
- Clock stretching
- Handshaking
- Clock rate and IBFD settings

45.5.4.1 Clock synchronization

Due to the wired-AND logic on the SCL line, a high-to-low transition on the SCL line affects all devices connected on the bus. The devices begin counting their low period when the master drives the SCL line low. After a device has driven SCL low, it holds the SCL line low until the clock high state is reached.

However, the change of low-to-high in a device clock may not change the state of the SCL line if another device is still within its low period. Therefore, the synchronized clock signal, SCL, is held low by the device with the longest low period. Devices with shorter low periods enter a high wait state during this time (see the following figure). When all devices concerned have counted off their

low period, the synchronized SCL line is released and pulled high. Then there is no difference between the devices' clocks and the state of the SCL line, and all the devices begin counting their high periods. The first device to complete its high period pulls the SCL line low again.

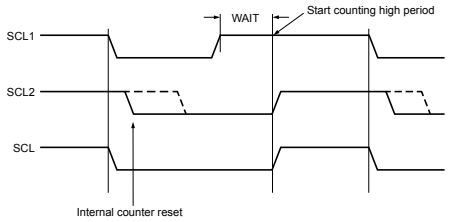


Figure 355. I²C bus clock synchronization

45.5.4.2 Clock stretching

Slaves can use the clock synchronization mechanism to slow down the bit rate of a transfer. After the master drives SCL low, the slave can drive SCL low for the required period and then release it. If the slave's SCL low period is greater than the master's SCL low period, the resulting SCL bus signal low period is stretched.

45.5.4.3 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfer. Slave devices may hold the SCL low after completion of one byte transfer (nine bits). In such cases, it halts the bus clock and forces the master clock into wait state until the slave releases the SCL line.

45.5.4.4 Clock rate and IBFD settings

45.5.4.4.1 Timing definitions

Table 293. Timing definitions relevant to clock rate and IBFD settings

Term	Definition
SCL Divider	The factor used to prescale the CPU clock for bit rate selection. This is relevant to: • Figure 356 • Table 295 • Table 296
SCL period	(CPU clock period)×(SCL Divider)
SCL Hold	The required number of CPU clocks to generate a START or STOP condition. This is relevant to: • Figure 356 • Table 295 • Table 296
SDA Hold	This is relevant to: • Figure 357 • Table 295 • Table 296

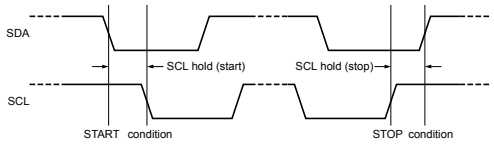


Figure 356. SCL Divider and SDA Hold

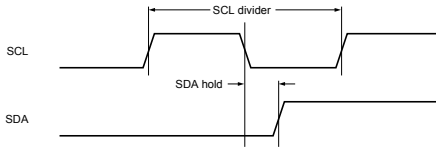


Figure 357. SDA Hold time

45.5.4.4.2 Divider and hold values

Table 295 provides the values for:

- SCL Divider
- SDA Hold
- SCL Hold (start)
- SCL Hold (stop)

when the glitch filter is disabled. (To disable this filter, write IBDBG[GLFLT_EN]=0.)

You have up to three MUL options available for all divider values, as shown in Table 294. Your choice of MUL determines the internal monitor rate of the I^2C bus (SCL and SDA signals):

- A lower MUL value results in a higher sampling rate of the I^2C signals.
- A higher MUL value results in a lower sampling rate of the I^2C signals. This gives the I^2C module greater immunity against glitches in the I^2C signals.

Table 294. MUL values as a function of IBC

When IBC is...	MUL is...
00h–3Fh	1
40h–7Fh	2
80h–BFh	4

NOTE

The frequency divider values listed in Table 295 and Table 296 depend on the board parasitics at the SCL pad. The observed I^2C frequency may differ from the expected frequency by 10–15%.

Table 295. I^2C divider and hold values when glitch filter is disabled

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
00	34	7	6	11
01	36	7	7	12

Table continues on the next page...

Table 295. I²C divider and hold values when glitch filter is disabled (continued)

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
02	38	8	8	13
03	40	8	9	14
04	42	9	10	15
05	44	9	11	16
06	46	10	13	18
07	54	10	16	21
08	44	7	10	15
09	48	7	12	17
0A	52	9	14	19
0B	56	9	16	21
0C	60	11	18	23
0D	64	11	20	25
0E	72	13	24	29
0F	84	13	30	35
10	64	9	18	25
11	72	9	22	29
12	80	13	26	33
13	88	13	30	37
14	96	17	34	41
15	104	17	38	45
16	120	21	46	53
17	144	21	58	65
18	96	9	38	41
19	112	9	46	49
1A	128	17	54	57

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e25-4821-48a8-b251-390d3ab624e1

Table 295. I²C divider and hold values when glitch filter is disabled (continued)

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
1B	144	17	62	65
1C	160	25	70	73
1D	176	25	78	81
1E	208	33	94	97
1F	256	33	118	121
20	160	17	78	81
21	192	17	94	97
22	224	33	110	113
23	256	33	126	129
24	288	49	142	145
25	320	49	158	161
26	384	65	190	193
27	480	65	238	241
28	320	33	158	161
29	384	33	190	193
2A	448	65	222	225
2B	512	65	254	257
2C	576	97	286	289
2D	640	97	318	321
2E	768	129	382	385
2F	960	129	478	481
30	640	65	318	321
31	768	65	382	385
32	896	129	446	449
33	1024	129	510	513

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e25-4821-48a8-b1b1-390d3ab624e1

Table 295. I²C divider and hold values when glitch filter is disabled (continued)

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
34	1152	193	574	577
35	1280	193	638	641
36	1536	257	766	769
37	1920	257	958	961
38	1280	129	638	641
39	1536	129	766	769
3A	1792	257	894	897
3B	2048	257	1022	1025
3C	2304	385	1150	1153
3D	2560	385	1278	1281
3E	3072	513	1534	1537
3F	3840	513	1918	1921
40	68	14	12	22
41	72	14	14	24
42	76	16	16	26
43	80	16	18	28
44	84	18	20	30
45	88	18	22	32
46	96	20	26	36
47	108	20	32	42
48	88	14	20	30
49	96	14	24	34
4A	104	18	28	38
4B	112	18	32	42
4C	120	22	36	46

Table continues on the next page...

Table 295. I²C divider and hold values when glitch filter is disabled (continued)

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
4D	128	22	40	50
4E	144	26	48	58
4F	168	26	60	70
50	128	18	36	50
51	144	18	44	58
52	160	26	52	66
53	176	26	60	74
54	192	34	68	82
55	208	34	76	90
56	240	42	92	106
57	288	42	116	130
58	192	18	76	82
59	224	18	92	98
5A	256	34	108	114
5B	288	34	124	130
5C	320	50	140	146
5D	356	50	156	162
5E	416	66	188	194
5F	512	66	236	242
60	320	34	156	162
61	384	34	188	194
62	448	66	220	226
63	512	66	252	258
64	576	98	284	290
65	640	98	316	322

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e2e5-4821-48a8-b251-390d3ab624e1

Table 295. I²C divider and hold values when glitch filter is disabled (continued)

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
66	768	130	380	386
67	960	130	476	482
68	640	66	316	322
69	768	66	380	386
6A	896	130	444	450
6B	1024	130	508	514
6C	1152	194	572	578
6D	1280	194	636	642
6E	1536	258	764	770
6F	1920	258	956	962
70	1280	130	636	642
71	1536	130	764	770
72	1792	258	892	898
73	2048	258	1020	1026
74	2304	386	1148	1154
75	2560	386	1276	1282
76	3072	514	1532	1538
77	3840	514	1916	1922
78	2560	258	1276	1282
79	3072	258	1532	1538
7A	3584	514	1788	1794
7B	4096	514	2044	2050
7C	4608	770	2300	2306
7D	5120	770	2556	2562
7E	6144	1026	3068	3074

Table continues on the next page...

Table 295. I²C divider and hold values when glitch filter is disabled (continued)

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
7F	7680	1026	3836	3842
80	136	28	24	44
81	144	28	28	48
82	152	32	32	52
83	160	32	36	56
84	168	36	40	60
85	176	36	44	64
86	192	40	52	72
87	216	40	64	84
88	176	28	40	60
89	192	28	48	68
8A	208	36	56	76
8B	224	36	64	84
8C	240	44	72	92
8D	256	44	80	100
8E	288	52	96	116
8F	336	52	120	140
90	256	36	72	100
91	288	36	88	116
92	320	52	104	132
93	352	52	120	148
94	384	68	136	164
95	416	68	152	180
96	480	84	184	212
97	576	84	232	260

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624...

Table 295. I²C divider and hold values when glitch filter is disabled (continued)

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
98	358	36	152	164
99	448	36	184	196
9A	512	68	216	228
9B	576	68	248	260
9C	640	100	280	292
9D	704	100	312	324
9E	832	132	376	388
9F	1024	132	472	484
A0	640	68	312	324
A1	768	68	376	388
A2	896	132	440	452
A3	1024	132	504	516
A4	1152	196	568	580
A5	1280	196	632	644
A6	1536	260	760	772
A7	1920	260	952	964
A8	1280	132	632	644
A9	1536	132	760	772
AA	1792	260	888	900
AB	2048	260	1016	1028
AC	2304	388	1144	1156
AD	2560	388	1272	1284
AE	3072	516	1528	1540
AF	3840	516	1912	1924
B0	2560	260	1272	1284

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e5-4821-48a8-b5b1-390d3ab624

Table 295. I²C divider and hold values when glitch filter is disabled (continued)

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
B1	3072	260	1528	1540
B2	3584	516	1784	1796
B3	4096	516	2040	2052
B4	4608	772	2296	2308
B5	5120	772	2552	2564
B6	6144	1028	3064	3076
B7	7680	1028	3832	3844
B8	5120	516	2552	2564
B9	6144	516	3064	3076
BA	7168	1028	3576	3588
BB	8192	1028	4088	4100
BC	9216	1540	4600	4612
BD	10240	1540	5112	5124
BE	12288	2052	6136	6148
BF	15360	2052	7672	7684

Table 296 provides the values for:

- SCL Divider
- SDA Hold
- SCL Hold (start)
- SCL Hold (stop)

when the glitch filter is enabled. (To enable this filter, write IBDBG[GLFLT_EN]=1.)

Table 296. I²C divider and hold values when glitch filter is enabled

IBC (hex)	SCL Divider (clocks)	SDA Hold (clocks)	SCL Hold (start)	SCL Hold (stop)
00	34	14	6	18
01	36	14	7	19
02	38	15	8	20

Table continues on the next page...

Table 296. I²C divider and hold values when glitch filter is enabled (continued)

03	40	15	9	21
04	42	16	10	22
05	44	16	11	23
06	48	17	13	25
07	54	17	16	28
08	44	15	10	23
09	48	15	12	25
0A	52	17	14	27
0B	56	17	16	29
0C	60	19	18	31
0D	64	19	20	33
0E	72	21	24	37
0F	84	21	30	43
10	64	17	18	33
11	72	17	22	37
12	80	21	26	41
13	88	21	30	45
14	96	25	34	49
15	104	25	38	53
16	120	29	46	61
17	144	29	58	73
18	96	17	38	49
19	112	17	46	57
1A	128	25	54	65
1B	144	25	62	73
1C	160	33	70	81
1D	176	33	78	89
1E	208	41	94	105
1F	256	41	11	129
20	160	17	78	81
21	192	17	94	97
22	224	33	110	113
23	256	33	126	129

Table continues on the next page...

Table 296. I²C divider and hold values when glitch filter is enabled (continued)

24	288	49	142	145
25	320	49	158	161
26	384	65	190	193
27	480	65	238	241
28	320	33	158	161
29	384	33	190	193
2A	448	65	222	225
2B	512	65	254	257
2C	576	97	286	289
2D	640	97	318	321
2E	768	129	382	385
2F	960	129	478	481
30	640	65	318	321
31	768	65	382	385
32	896	129	446	449
33	1024	129	510	513
34	1152	193	574	577
35	1280	193	638	641
36	1536	257	766	769
37	1920	257	958	961
38	1280	129	638	641
39	1536	129	766	769
3A	1792	257	894	897
3B	2048	257	1022	1025
3C	2304	385	1150	1153
3D	2560	385	1278	1281
3E	3072	513	1534	1537
3F	3840	513	1918	1921
40	68	28	12	36
41	72	28	14	38
42	76	30	16	40
43	80	30	18	42
44	84	32	20	44

Table continues on the next page...

Table 296. I²C divider and hold values when glitch filter is enabled (continued)

45	88	32	22	46
46	96	34	26	50
47	108	34	32	56
48	88	30	20	46
49	96	30	24	50
4A	104	34	28	54
4B	112	34	32	58
4C	120	38	36	62
4D	128	38	40	66
4E	144	42	48	74
4F	168	42	60	86
50	128	34	36	66
51	144	34	44	74
52	160	42	52	82
53	176	42	60	90
54	192	50	68	98
55	208	50	76	106
56	240	58	92	122
57	288	58	116	146
58	192	34	76	98
59	224	34	92	114
5A	256	50	108	130
5B	288	50	124	146
5C	320	66	140	162
5D	352	66	156	178
5E	416	82	188	210
5F	512	82	236	258
60	320	34	156	162
61	384	34	188	194
62	448	66	220	226
63	512	66	252	258
64	576	98	284	290
65	640	98	316	322

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHTC Automotive Technology Co., Ltd.
c8c4e2e5-4821-48a8-b251-390d3ab62d61

Table 296. I²C divider and hold values when glitch filter is enabled (continued)

66	768	130	380	386
67	960	130	476	482
68	640	66	316	322
69	768	66	380	386
6A	896	130	444	450
6B	1024	130	508	514
6C	1152	194	572	578
6D	1280	194	636	642
6E	1536	258	764	770
6F	1920	258	956	962
70	1280	130	636	642
71	1536	130	764	770
72	1792	258	892	898
73	2048	258	1020	1026
74	2304	386	1148	1154
75	2560	386	1276	1282
76	3072	514	1532	1538
77	3840	514	1916	1922
78	2560	258	1276	1282
79	3072	258	1532	1538
7A	3584	514	1788	1794
7B	4096	514	2044	2050
7C	4608	770	2300	2306
7D	5120	770	2556	2562
7E	6144	1026	3068	3074
7F	7680	1026	3836	3842
80	136	56	24	72
81	144	56	28	76
82	152	60	32	80
83	160	60	36	84
84	168	64	40	88
85	176	64	44	92
86	192	68	52	100

Table continues on the next page...

Table 296. I²C divider and hold values when glitch filter is enabled (continued)

87	216	68	64	112
88	176	60	40	92
89	192	60	48	100
8A	208	68	56	108
8B	224	68	64	116
8C	240	76	72	124
8D	256	76	80	132
8E	288	84	96	148
8F	336	84	120	172
90	256	68	72	132
91	288	68	88	148
92	320	84	104	164
93	352	84	120	180
94	384	100	136	196
95	416	100	152	212
96	480	116	184	244
97	576	116	232	292
98	384	68	152	196
99	448	68	184	228
9A	512	100	216	260
9B	576	100	248	292
9C	640	132	280	324
9D	704	132	312	356
9E	832	164	376	420
9F	1024	164	472	516
A0	640	68	312	324
A1	768	68	376	388
A2	896	132	440	452
A3	1024	132	504	516
A4	1152	196	568	580
A5	1280	196	632	644
A6	1536	260	760	772
A7	1920	260	952	964

Table continues on the next page...

Table 296. I²C divider and hold values when glitch filter is enabled (continued)

A8	1280	132	632	644
A9	1536	132	760	772
AA	1792	260	888	900
AB	2048	260	1016	1028
AC	2304	388	1144	1156
AD	2560	388	1272	1284
AE	3072	516	1528	1540
AF	3840	516	1912	1924
B0	2560	260	1272	1284
B1	3072	260	1528	1540
B2	3584	516	1784	1796
B3	4096	516	2040	2052
B4	4608	772	2296	2308
B5	5120	772	2552	2564
B6	6144	1028	3064	3076
B7	7680	1028	3832	3844
B8	5120	516	2552	2564
B9	6144	516	3064	3076
BA	7168	1028	3576	3588
BB	8192	1028	4088	4100
BC	9216	1540	4600	4612
BD	10240	1540	5112	5124
BE	12288	2052	6136	6148
BF	15360	2052	7672	7684

45.5.5 Interrupts

This section presents the following topics:

- Interrupt vector
- Interrupt description

45.5.5.1 Interrupt vector

The I²C module uses only one interrupt vector.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab62f6c

Table 297. Interrupt summary

Interrupt	Offset	Vector	Priority	Source	Description
I2C interrupt	—	—	—	IBAL, TCF, IAAS, IBB fields in IBSR	When any of IBAL, TCF, or IAAS fields are 1, the I2C module may cause an interrupt based on Arbitration lost, Transfer Complete, or Address Detect conditions. If enabled by BIIE, the de-assertion of IBB can also cause an interrupt, indicating that the bus is idle.

45.5.5.2 Interrupt description

There are five types of internal interrupts in the I2C. The interrupt service routine can determine the interrupt type by reading the Status register.

I2C Interrupt can be generated on the following events:

- Arbitration Lost condition (IBSR[IBAL]=1)
- Byte Transfer condition (IBSR[TCF]=1 and IBCR[DMAEN]=0)
- Address Detect condition (IBSR[IAAS]=1)
- No Acknowledge from slave received when expected
- Bus idle (IBSR[IBB]=0)

The IBCR[IBIE] field enables the I2C interrupt. To clear this field, program IBSR[IBIF]=1 in the interrupt service routine.

To use the Bus idle interrupt, you must enable it by using the IBIC[BIIE] field.

45.5.6 DEBUG mode

This mode allows the CPU to debug the I2C module by freezing all the counters, registers, and status bits. To enter this mode, the I2C module must receive a DEBUG mode request, defined as this combination of actions:

- Another module on the chip sends an internal debug signal to the I2C module.
- You program IBDBG[IPG_DEBUG_EN]=1.

The I2C module completes all the ongoing transactions and comes to a graceful halt after receiving a DEBUG mode request.

The I2C module programs IBDBG[IPG_DEBUG_HALTED]=1 to indicate that the module has successfully serviced the DEBUG mode request.

The I2C module programs IBDBG[IPG_DEBUG_HALTED]=0 after the module that sent the DEBUG mode request ends that request and you program IBDBG[IPG_DEBUG_EN]=0.

As soon as the I2C module enters DEBUG mode:

- No transaction can take place.
- All the registers, counters and status bits are frozen. They all can be accessed by the CPU to enable the debugging.

The I2C module enters this mode only after successfully completing the current ongoing transaction. For example, if the current transaction consists of eight bytes and you initiated DEBUG mode at the time of the second byte, the IPG Debug Halted signal will be asserted after the eighth byte is transmitted/received. See the simulation result in Figure 358 for more details.

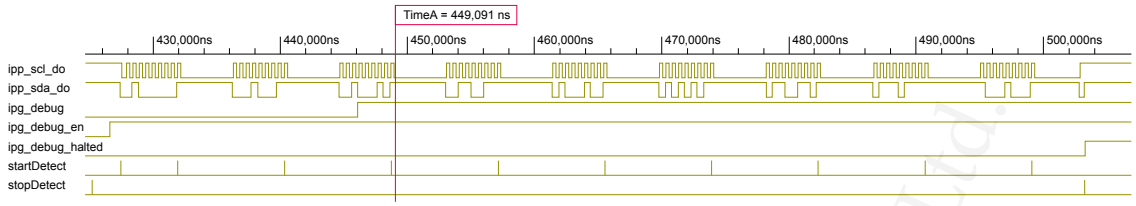


Figure 358. Simulation result of IPG Debug Halted in case of frame transmission

The I^{2}C module enters DEBUG mode after completing the transmission or reception of the current byte when both of these conditions are true:

- Any DMA transaction (transmit or receive) is in progress
- The I^{2}C module receives a DEBUG mode request at the same time that a byte reception or transmission occurs

As soon as the module exits DEBUG mode, transmission or reception of the remaining bytes resumes. See the simulation snapshot shown in Figure 359 for details where the I^{2}C is transmitting eight bytes using DMA and the I^{2}C module receives a DEBUG mode request while the module is transmitting the second byte. The I^{2}C module enters DEBUG mode after the module successfully transmits the second byte (IPG Debug Halted signal asserted). As soon as the module exits DEBUG mode and deasserts the IPG Debug Halted signal, transmission resumes successfully.

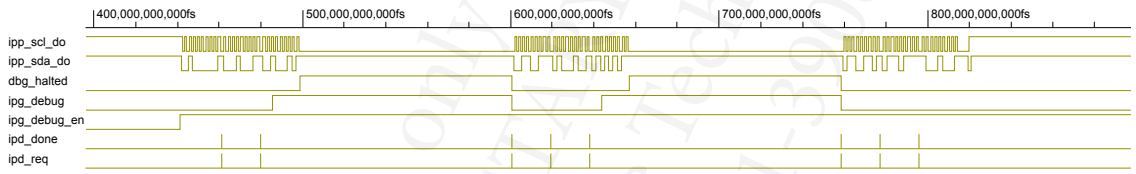


Figure 359. Simulation result of IPG Debug Halted in case of frame transmission using DMA

No more transactions can occur until the module that sent the DEBUG mode request ends that request. At that time, the I^{2}C module starts functioning normally. You can examine the IBDBG[IPG_DEBUG_HALTED] field to determine whether the I^{2}C module is in DEBUG mode. See Figure 360 for more details.

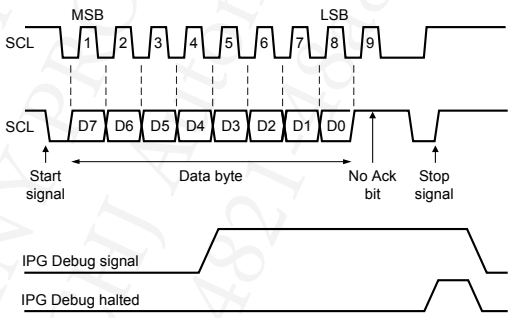
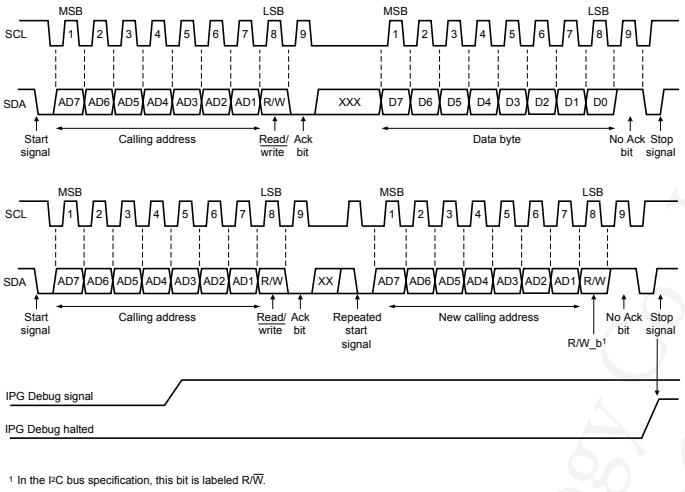


Figure 360. DEBUG mode

Figure 361 shows a case of DEBUG mode with repeated START transaction. In this case, if another module on the chip sends an internal debug signal to the I^{2}C module in the middle of a transaction, the I^{2}C module completes the entire transaction with multiple repeated START before the module enters DEBUG mode.



[1] In the I²C bus specification, this bit is labeled R/$\overline{W}$.

Figure 361. DEBUG mode with repeated START

45.5.7 DMA interface

The I²C module implements a simple DMA interface so that the module can request data transfers with minimal support from the CPU (see DMA application information). You enable DMA mode by programming IBCR[DMAEN].

The DMA interface is operational when the I²C module is configured for Master mode.

You must transfer at least three bytes of data per frame to/from the slave when using DMA mode. However, in practice, DMA mode is only worthwhile when you have a large number of data bytes to transfer per frame.

The I²C module uses two internal signals—TX request and RX request—to signal the DMA controller when the I²C module must write data to IBDR or read data from IBDR.

See Initialization/application information for further details of the DMA interface.

45.6 Initialization/application information

This section presents the following topics:

- Recommended interrupt service flow
- General programming guidelines (for both master and slave mode)
- Programming guidelines specific to Master mode
- Programming guidelines specific to Slave mode
- DMA application information

45.6.1 Recommended interrupt service flow

The following figure shows a flowchart for the recommended I²C interrupt service routine. Deviation from the flowchart may result in unpredictable I²C bus behavior.

Preliminary NDA Only
COMPANY PROPRIETARY
Beijing CH Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-92...9d3ab624c1

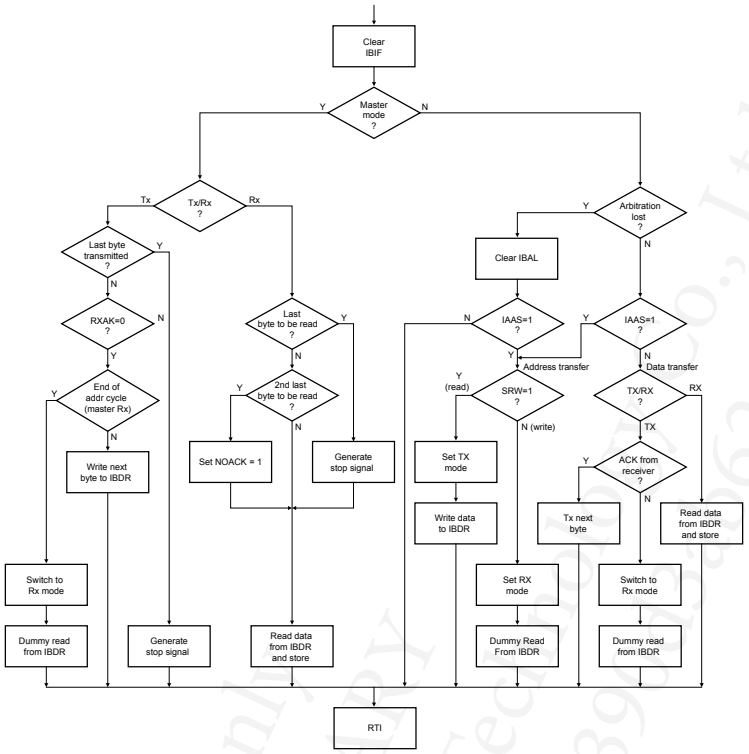


Figure 362. Recommended I²C interrupt service routine flowchart

45.6.2 General programming guidelines (for both master and slave mode)

This section provides programming guidelines recommended for the I²C module in both master and slave mode. It presents the following topics:

- Initializing the I2C module
- Software response after a transfer

45.6.2.1 Initializing the I²C module

The following table describes how to initialize the I²C module.

Table 298. I²C initialization procedure

Step	Action
1	Use I2C Bus Frequency Divider Register (IBFD) to select the required division ratio to obtain SCL frequency from the system clock.
2	Use I2C Bus Address Register (IBAD) to define the slave address.
3	Clear the MDIS field in I2C Bus Control Register (IBCR) to enable the I²C interface system.
4	Use I2C Bus Control Register (IBCR) to select Master/Slave mode, Transmit/Receive mode, and whether interrupts are enabled or disabled.
5	(Optional) Use I2C Bus Interrupt Config Register (IBIC) to further refine the interrupt behavior.

45.6.2.2 Software response after a transfer

Transmission or reception of a byte causes the I^2C module to program IBSR[TCF]=1. This indicates that the I^2C module transmitted one byte. The I^2C module also programs the interrupt flag IBSR[IBIF]=1 and generates an interrupt if your software enabled the interrupt function (by programming IBCR[IBIE]=1) during initialization. Your software can clear IBSR[IBIF] by writing a 1 to that field (in the interrupt service routine, if interrupts are used).

The I^2C module programs IBSR[TCF]=0 to indicate data transfer in progress whenever your software:

- Writes to IBDR in transmit mode
- Reads the IBDR in receive mode

Do not use IBSR[TCF] as a "data transfer complete" flag. This is because the flag timing depends on a number of factors, including the I^2C bus frequency. This field may not conclusively provide an indication of a transfer-complete situation. Your software must detect transfer-complete situations by using IBSR[IBIF].

Your software may service the I^2C I/O in the main program by monitoring IBSR[IBIF] if your software disabled the interrupt function. Your software must poll IBSR[IBIF] instead of IBSR[TCF] because these fields function differently when arbitration is lost.

When a "Transfer Complete" interrupt occurs at the end of the address cycle, the master will always be in transmit mode—that is, the I^2C module transmits the address. If master receive mode is required, indicated by R/W bit sent with slave calling address, then the master's IBCR[TXRX] field must be toggled at this stage. If Master does not receive an ACK from Slave, then transmission must be re-initiated or terminated.

In slave mode, the I^2C module programs IBSR[IAAS]=1 if the Slave address (I2C Bus Address Register (IBAD)) matches the Master calling address. This is an indication that Master-Slave data communication can now start. During address cycles (IBSR[IAAS]=1), your software reads IBSR[SRW] to determine the direction of the subsequent transfer and programs IBCR[TXRX] accordingly. For slave mode data cycles (IBSR[IAAS]=0), IBSR[SRW] is not valid. Your software must read IBCR[TXRX] to determine the direction of the current transfer.

45.6.3 Programming guidelines specific to Master mode

This section presents the following topics:

- Generating START
- Transmit/receive sequence
- Generating STOP
- Generating repeated START
- Loss of arbitration

45.6.3.1 Generating START

After the I^2C module completes its initialization procedure, the module can transmit serial data when your software selects the master transmitter mode. If the device is connected to a multi-master bus system, your software must test the state of IBSR[IBB] to check whether the serial bus is free.

Your software can configure the I^2C module to send the START condition and the first byte (the slave address) if the bus is free (IBSR[IBB]=0). The data that your software writes to the data register (I2C Bus Data I/O Register (IBDR)) comprises the slave calling address and the LSB. Your software configures the LSB to indicate the direction of transfer required from the slave.

The bus free time (the time between a STOP condition and the following START condition) is built into the I^2C module that generates the START cycle. Depending on the relative frequencies of the system clock and the SCL period, your software must wait until the I^2C bus is busy after writing the calling address to the IBDR before proceeding with the next instructions. This is illustrated in the following example.

Here is an example of the sequence of events which generates the START signal and transmits the first byte of data (slave address):

```
while (IBSR[IBB]==1)               // wait in loop for IBB flag to clear
IBCR[MS/MSL] and IBCR[]Tx/Rx] = 1  // master and transmit mode, that is,
                                   // generate start condition
IBDR = calling_address             // send the calling address to the data register
while (bit 5, IBSR ==0)            // wait in loop for IBB flag to be set
```

45.6.3.2 Transmit/receive sequence

The following tables present the sequences for:

- Master transmit
- Master receive
- Slave transmit
- Slave receive

Table 299. Master transmit sequence

Step	Action
a	Use I2C Bus Frequency Divider Register (IBFD) to select the required division ratio to obtain SCL frequency from platform clock/2.
b	Write a 0 to IBCR[MDIS] to enable the I^2C interface system.
c	Use I2C Bus Control Register (IBCR) to select Master mode, Transmit mode, and interrupt enable.
d	Write a 0 to IBSR[IBIF].
e	Write data to I2C Bus Data I/O Register (IBDR).
f	Observe changes in IBSR[TCF]: • When IBSR[TCF] becomes 0, the transfer is in progress. • When IBSR[TCF] becomes 1, the transfer is complete.
g	Wait until IBSR[IBIF] becomes 1.
h	Read the fields in I2C Bus Status Register (IBSR) to determine what happened: • If TCF = 1, the transfer completed. • If RXAK = 1, a No Acknowledge condition occurred. • If IBB = 0, the bus transitioned from Busy to Idle state. • If IBB = 1, you can ignore check of Arbitration Loss (IBAL = 1). **NOTE** You can ignore Address Detect (IAAS = 1) for master mode (it is valid only for slave mode).
i	Examine IBSR[RXAK] for an acknowledgment from the slave.
j	Repeat steps d through i to transfer the next consecutive bytes of data.

Provided under NXP Proprietary Binding Agreement c8a3b2e5-2121-4da8-b251-3a0d3ab624c1

Table 300. Master receive sequence

Step	Action
a	Execute steps a through i in Table 299 for address dispatch.
b	Write a 0 to IBSR[IBIF].
c	Write a 0 IBCR[TXRX] to select Receive mode.
d	Perform a dummy read of I2C Bus Data I/O Register (IBDR) to initiate the receive operation.
e	Wait until IBSR[TCF] becomes 1. This proves that the transfer is complete.
f	Wait until IBSR[IBIF] becomes 1.
g	Read the fields in I2C Bus Status Register (IBSR) to determine what happened: • If TCF = 1, the reception completed. • If IBB = 0, the bus transitioned from Busy to Idle state. • If IBB = 1, you can ignore check of Arbitration Loss (IBAL = 1). **NOTE** You can ignore the No Acknowledge condition (RXAK = 1) for receive mode.
h	Read I2C Bus Data I/O Register (IBDR) to determine the data received from the slave.
i	Change IBSR[IBIF] and IBSR[TCF] to 0 by writing 1 to those fields.
j	Repeat steps e through i to read subsequent bytes.

Table 301. Slave transmit sequence

Step	Action
a	Use I2C Bus Address Register (IBAD) to define the slave address.
b	Write a 0 to IBCR[MDIS] to enable the I^2C interface system.
c	Examine fields in I2C Bus Status Register (IBSR) as follows: • If IAAS = 1, examine IBSR[SRW]. • If IAAS = 1 and SRW = 1, write a 1 to IBCR[TXRX] to select Transmit mode.
d	Write data to I2C Bus Data I/O Register (IBDR).
e	Wait until IBSR[IBIF] becomes 1.
f	Wait until IBSR[RXAK] becomes 0.
g	Write a 0 to IBSR[IBIF].
h	Repeat steps d through g for the next consecutive data transfers.

Table 302. Slave receive sequence

Step	Action
a	Use I2C Bus Address Register (IBAD) to define the slave address.
b	Write a 0 to IBCR[MDIS] to enable the I^2C interface system.

Table continues on the next page...

Table 302. Slave receive sequence (continued)

Step	Action
c	Examine fields in I2C Bus Status Register (IBSR) as follows: • If IAAS = 1, examine IBSR[SRW]. • If IAAS = 1 and SRW = 0, write a 0 to IBCR[TXRX] to select Receive mode.
d	Write a 0 to IBSR[IBIF].
e	Perform a dummy read of I2C Bus Data I/O Register (IBDR) to initiate the receive operation.
f	Wait until IBSR[TCF] becomes 1. This proves that the transfer is complete.
g	Wait until IBSR[IBIF] becomes 1.
h	Read the fields in I2C Bus Status Register (IBSR) to determine what happened: • If TCF = 1, the reception completed. • If IBB = 0, the bus transitioned from Busy to Idle state. • If IBB = 1, you can ignore check of Arbitration Loss (IBAL = 1). **NOTE** You can ignore the No Acknowledge condition (RXAK = 1) for receive mode.
i	Read I2C Bus Data I/O Register (IBDR) to determine the data received from the master.
j	Change IBSR[IBIF] and IBSR[TCF] to 0 by writing 1 to those fields.
k	Repeat steps f through j to read subsequent bytes.

45.6.3.3 Generating STOP

A data transfer ends with a STOP condition generated by the master device. A master transmitter can simply generate a STOP condition after it transmits all the data. Here is an example showing how a master transmitter generates a STOP condition.

```
if (tx_count == 0) or     // check to see if all data bytes have been transmitted
  (bit 0, IBSR == 1) {    // or if no ACK generated
    clear bit 5, IBCR     // generate STOP condition
  }
else {
 IBDR = data_to_transmit  // write byte of data to DATA register
   tx_count --            // decrement counter
 }                        // return from interrupt
```

If a master receiver wants to terminate a data transfer, it must inform the slave transmitter by not acknowledging the last byte of data. The master receiver can do this by programming IBCR[NOACK]=1 before reading the second-to-last byte of data. Before reading the last byte of data, the master receiver must first generate a STOP condition. Here is an example showing how a master receiver generates a STOP condition.

```
rx_count --           // decrease the rx counter
if (rx_count ==1)     // 2nd last byte to be read ?
 bit 3, IBCR = 1      // disable ACK
 if (rx_count == 0)   // last byte to be read ?
 bit 5, IBCR = 0      // generate STOP condition
```

```
else
data_received = IBDR   // read RX data and store
```

45.6.3.4 Generating repeated START

At the end of data transfer, if the master still wants to communicate on the bus, it can generate another START signal followed by another slave address without first generating a STOP signal. Here is a program example.

```
bit 2, IBCR = 1             // generate another start (restart)
IBDR == calling_address     // transmit the calling address
```

45.6.3.5 Loss of arbitration

If several masters try to engage the bus simultaneously, only one master wins and the others lose arbitration. The devices that lost arbitration are immediately switched to slave mode by the I^{2}C module. Their data output to the SDA line is stopped, but SCL is still generated until the end of the byte during which arbitration was lost. An interrupt occurs at the falling edge of the ninth clock of this transfer with IBSR[IBAL]=1 and IBCR[MSSL]=0. If one master attempts to start transmission while the bus is being engaged by another master, the hardware:

- Inhibits the transmission
- Switches IBCR[MSSL] from 1 to 0 without generating a STOP condition
- Generates an interrupt to the CPU
- Programs IBSR[IBAL]=1 to indicate that the attempt to engage the bus is failed
- Does not program IBSR[TCF]=1 due to the loss of data during arbitration

When considering these cases, your slave service routine should test IBSR[IBAL] first and your software should program IBSR[IBAL]=0 if IBSR[IBAL]=1.

45.6.4 Programming guidelines specific to Slave mode

In the slave interrupt service routine, IBSR[IAAS] should be tested to check if a calling of its own address has just been received. If IBSR[IAAS]=1, software should program IBCR[TXRX]=1 according to IBSR[SRW]. Writing to the IBCR clears IBSR[IAAS] automatically. The only time you will read a value of 1 from IBSR[IAAS] is from the interrupt at the end of the address cycle where an address match occurred. Interrupts resulting from subsequent data transfers will have IAAS cleared.

A data transfer may now be initiated by writing information to IBDR for slave transmits or dummy reading from IBDR in slave receive mode. The slave will drive SCL low between byte transfers. SCL is released when the IBDR is accessed in the required mode.

In slave transmitter routine, IBSR[RXAK] must be tested before transmitting the next byte of data. Setting RXAK means an end-of-data signal from the master receiver, after which it must be switched from transmitter mode to receiver mode by software. A dummy read then releases the SCL line so that the master can generate a STOP signal.

45.6.5 DMA application information

The DMA interface on the I^{2}C module is not completely autonomous and requires intervention from the CPU to start and to terminate the frame transfer. DMA mode is only valid for Master transmit and Master receive modes. Your software must ensure that the DMAEN field in I2C Bus Control Register (IBCR) is not set when the I^{2}C module is configured in slave mode.

The DMA controller must only transfer one byte of data per Tx/Rx request. This is because the I^{2}C module does not have a FIFO.

The CPU should also keep the I^{2}C interrupt enabled during a DMA transfer to detect the arbitration lost condition and take action to recover from this situation. The DMAEN field in I2C Bus Control Register (IBCR) can disable the transfer complete interrupt. This means that during normal transfers (no errors) there will always be either an interrupt or a request to the DMA controller, depending on the setting of IBCR[DMAEN]. All error conditions will trigger an interrupt and require CPU intervention. The address match condition will not occur in DMA mode because the I^{2}C should never be configured for slave operation.

The following sections provide details on setting up a DMA transfer and what intervention is required from the CPU. The sections assume that the system DMA controller can generate an interrupt after a certain number of DMA transfers have occurred.

The sections present the following topics:

- DMA mode, master transmit
- DMA mode, master reception
- Exiting DMA mode and system requirement considerations

45.6.5.1 DMA mode, master transmit

The following flowchart describes the exact operation for using a DMA controller to transmit *n* data bytes to a slave. The CPU always transmits the first byte (the slave calling address) and last byte. The DMA controller can transfer all other data bytes.

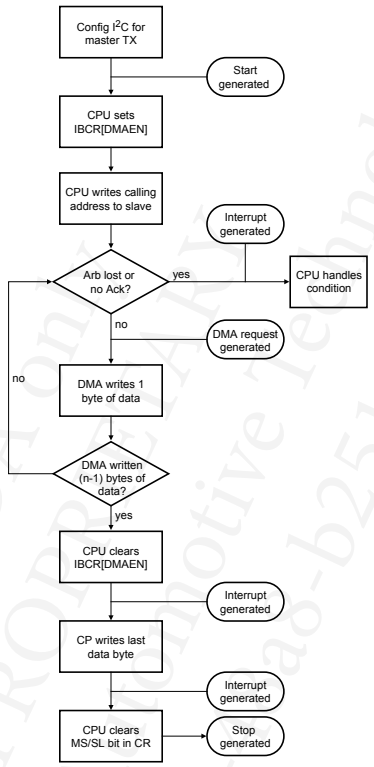


Figure 363. Flowchart of DMA mode master transmit

45.6.5.2 DMA mode, master reception

The following flowchart describes the exact operation for using a DMA controller to receive *n* data bytes from a slave. The CPU always transmits the first byte (the slave calling address) and the last two bytes. The DMA controller can read all other data bytes.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4ab-b251-390d3ab62f

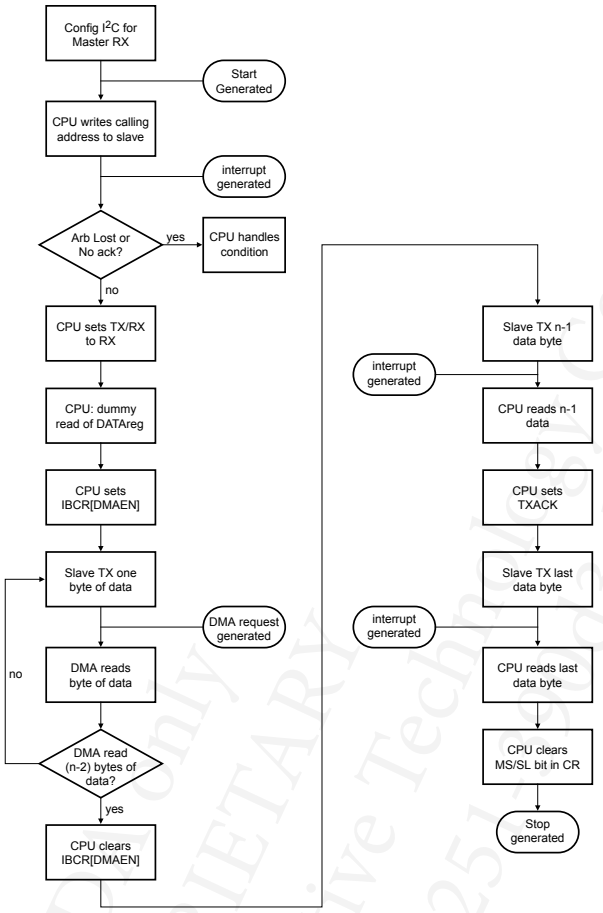


Figure 364. Flowchart of DMA mode master receive

45.6.5.3 Exiting DMA mode and system requirement considerations

As described above, the CPU must request an interrupt to process the last few bytes of a TX or RX transfer. To change from DMA to interrupt-driven transfers in the I^{2}C module, you must disable IBCR[DMAEN]. The trigger to exit the DMA mode is that the programmed DMA Transfer Control Descriptor (TCD) has completed all its transfers to/from the I^{2}C module.

After the last DMA write (TX mode) to the I^{2}C the module immediately starts the next I^{2}C bus transfer. The same is true for receive mode. After the DMA read from the IBDR register the module initiates the next I^{2}C bus transfer. This results in two possible scenarios in the DMA mode exiting scheme.

1. Fast reaction

   IBCR[DMAEN] is cleared before the next I^{2}C bus transfer completes. In this case, the I^{2}C module raises an interrupt request to the CPU which can be serviced normally.

2. Slow reaction

   IBCR[DMAEN] is cleared after the next I^{2}C bus transfer has already completed. In this case, the I^{2}C module does not raise an interrupt request to the CPU. Instead, IBSR[TCF] can be read to determine that the transfer completed and the I^{2}C module is ready for further transfer.

What is fast/slow reaction?

The reaction time T_R for the system to disable IBCR[DMAEN] after the last DMA controller access to the I^{2}C is the time required for one byte transfer over the I^{2}C. For a fast reaction, the disabling must occur before the 9th (ACK) bit of the data transfer. Therefore, the time available is eight times the SCL period.

$$T_R = 8 \times T_{SCL}$$

In fast mode, with 400 kbit/s, T_{SCL} is 2.5 µs, so T_R is 20 µs.

Depending on the system and DMA controller there are different possibilities for the de-assertion of IBCR[DMAEN]. Three options are:

1. CPU intervention via interrupt

   The DMA controller is programmed to signal an interrupt to the CPU. The CPU is then responsible for programming IBCR[DMAEN]=0. This scheme should be supported by most systems but it can result in a slow reaction time if other higher-priority interrupts interfere. Therefore, the interrupt handling routine can become complicated. This is because the routine must check which of the two cases happened (by checking IBSR[TCF]) and act accordingly. In case of a slow reaction, you can force an interrupt for the I^2C in the interrupt controller to have the further transfer handled by the normal I^2C interrupt routine.

   NOTE

   The use of nested interrupts can still cause potential issues in this scenario, if someone tries to stall the DMA interrupt between the de-assertion and IBCR[DMAEN] and checks IBSR[TCF].

2. DMA channel linking

   If the TCD in the DMA controller that performs the data transfer is linked to another channel that does a write to IBCR to disable the DMAEN field, this might probably be the fastest system solution, but it uses two DMA channels.

   NOTE

   In this case, you must ensure—at the system level—that no higher-priority DMA requests occur between the two linked TCDs. Such requests could again create a scenario of slow reaction.

3. DMA scatter/gather process

   If the TCD in the DMA controller that performs the data transfer has the scatter/gather feature activated, this feature initiates a reload of another TCD from system RAM after the completion of the first TCD. The new TCD will have its start bit already set and immediately start the required write to the IBCR to disable the DMAEN field. This TCD also has scatter/gather activated and is programmed to reload the initial TCD upon completion, bringing the system back into a "ready-for-I^2C-transfer" state. The advantage over the two other solutions is that this requires neither CPU intervention nor a second DMA channel. This comes at the cost of 64 bytes RAM (two TCDs), some system bus transfer overhead, and a little increase in application code complexity.

   NOTE

   Here you must ensure—at the system level—that no higher-priority DMA requests occur during the scatter/gather process. Such requests could again create a scenario of slow reaction.

Example latencies for a 32 MHz system with a full speed 32-bit AHB bus and an I^2C connected via half-speed IPI bus:

- Accessing the I^2C from the DMA controller via IPI bus typically requires four cycles (consecutive accesses to the I^2C could be faster):

  $4 \times T_{IPI}$ = 4/16 MHz = 250 ns

- Reloading a new TCD (8 x 32-bit) via AHB to the DMA controller (scatter/gather process):

  $8 \times T_{AHD}$ = 8/32 MHz = 250 ns

Example latencies for a 150 MHz system with a full speed 32-bit AHB bus and an I^2C connected via half speed IPI bus:

- Accessing the I^2C from the DMA controller via IPI bus typically requires four cycles (consecutive accesses to the I^2C could be faster):

  $4 \times T_{IPI}$ = 4/150 MHz = 26.6 ns

- Reloading a new TCD (4 x 64-bit) via AHB to the DMA controller (scatter/gather process):

  $4 \times T_{AHD}$ = 4/150 MHz = 26.6 ns

With the DMA scatter/gather process, the required IBCR access can finish in 0.5 µs, leaving a large margin of 19.5 µs for additional system delays. In this way, the slow reaction case can be prevented. You must decide which usage model best suits your overall requirement.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 46
Gigabit Ethernet Media Access Controller (GMAC)

46.1 Introduction

NOTE

The remaining sections in the chapter are Synopsys Proprietary. Used with permission.

GMAC enables a host application to transmit and receive data over an Ethernet network in compliance with IEEE 802.3-2015. GMAC supports 10/100/1000/2500 Mbit/s [3] applications.

Table 303. Terms and definitions

Term	Definition
APB	Advanced Peripheral Bus—An implementation of the Arm® AMBA® APB interface.
AVB	Audio Video Bridging—Transmission of audio and video signals over an Ethernet network.
AXI	Advanced eXtensible Interface—An implementation of the Arm AMBA AXI interface. GMAC has an AXI master interface that connects to all GMAC DMA channels.
Backpressure	A mechanism for stalling reception of packets transmitted over an Ethernet network.
CRC	Cyclic Redundancy Check—A calculation performed on received data packets used to detect corrupted data.
DMA	Direct Memory Access—Hardware within GMAC that enables movement of data to/from system memory with minimal intervention from the core.
FIFO	First In First Out—A type of data buffer from which data is retrieved in the same order in which it was stored in the buffer.
GMII	Gigabit Media Independent Interface—A data interface and management interface that connects a MAC to a PHY, defining speeds of up to 1000 Mbit/s.
IEEE 802.1AS-Rev	The enhanced IEEE standard for Timing and Synchronization for Time-Sensitive Applications (AVB, for example).
IEEE 802.1Q	The IEEE standard for VLANs on an IEEE 802.3 Ethernet network.
IEEE 802.1Qbu	An amendment to the IEEE 802.1Q standard that supports frame preemption.
IEEE 802.1Qbv	An amendment to the IEEE 802.1Q standard that supports scheduled traffic.
IEEE 802.3-2015	The IEEE standard that specifies Ethernet network operation.
IPv4, IPv6	Internet Protocol version *n*—Refers to versions of the Internet Protocol specification.
Layer 3, Layer 4	Refers to layers defined within the Open System Interconnection (OSI) model of networking. The layers are:

Layer	Name
1	Physical
2	Data Link
3	Network

[3] Supported PHY interface speeds vary per device

Term	Definition
	<table><tr><th>Layer</th><th>Name</th></tr><tr><td>4</td><td>Transport</td></tr><tr><td>5</td><td>Session</td></tr><tr><td>6</td><td>Presentation</td></tr><tr><td>7</td><td>Application</td></tr></table>
MAC	Media Access Controller—Refers to the data-link layer defined by the IEEE 802.3 standard.
MDIO	Management Data Input/Output—A serial interface used for PHY configuration.
MII	Media Independent Interface—A data interface and management interface that connects a MAC to a PHY. The IEEE 802.3 standard defines it..
MTL	MAC Translation Layer—The GMAC component that contains the receive and transmit FIFOs.
Pause packet	Refers to a mechanism defined by the IEEE 802.3x standard, used to request a temporary halt to data transmission on an Ethernet network.
PHY	The physical interface transceiver that implements the OSI physical layer for an Ethernet network. The IEEE 802.3 standard defines it.
RGMII	Reduced Gigabit Media Independent Interface—A GMII specification that has a reduced pin count.
RMII	Reduced Media Independent Interface—An MII specification with a reduced number of signals.
RMON	Remote Network Monitoring—An extension of the Simple Network Management Protocol (SNMP) that enables monitoring of Ethernet network statistics.
Rx	Receive
SFD	Start Frame Delimiter—A field within an Ethernet frame.
SGMII	Serial Gigabit Media Independent Interface—A variant of the GMII that has a reduced pin count and lower power consumption.
TCP	Transmission Control Protocol—A network protocol that provides a mechanism for transmitting and receiving an ordered and error-checked stream of information packets.
Tx	Transmit
UDP	User Datagram Protocol—A network protocol that provides higher speeds than TCP but does not include error checking.
VLAN	Virtual Local Area Network.

46.2 Features

In addition to the default features defined in the IEEE 802.3 specifications, GMAC supports the following:

- MII 10/100 Mbit/s, RMII (10/100 Mbit/s), RGMII (100/1000 Mbit/s)
- GMII to support SGMII (1000/2500 Mbit/s) (supported PHY interface speeds vary per device—check the specifications for each chip)

- Time Aware Shaper (IEEE 802.1Qbv), Time Synchronization (IEEE 802.1AS-Rev), and Frame Preemption (IEEE 802.1Qbu) for time-sensitive networking
- Media clock recovery and generation for AVB
- AXI4 master and APB3 slave
- RMII specification version 1.2 from RMII consortium
- RGMII specification version 2.6 from HP/Marvell
- Separate interfaces for transmit, receive, and control paths
- Full-duplex flow control operations (IEEE 802.3x pause packet and priority flow control)
- Network statistics with RMON counters
- Flexibility to control Pulse per Second (PPS) output signals and support for auxiliary snapshot input signals
- Ethernet packet timestamping as described in IEEE 1588-2002 and IEEE 1588-2008 (64-bit timestamps given in the transmit or receive status of PTP packet) (both one-step and two-step timestamping are supported in transmit direction)
- MDIO (clause 22 and clause 45) interface for configuration and management of the PHY
- Preamble and start of packet data (SFD) insertion/deletion
- Automatic CRC and pad generation/stripping option, with an option to disable CRC checking
- Programmable interpacket gap
- Source address insertion/replacement, VLAN insertion, replacement, and deletion in transmitted packets with per-packet or static-global control IEEE 802.1Q VLAN tag detection, and option to delete in received packets
- Programmable safety watchdog timeout limits
- Flexible address filtering modes:
  - Up to 1 additional 48-bit perfect destination address filters with masks for each byte
  - Up to one 48-bit source address comparison check with masks for each byte
  - Optional 256-bit hash filter for multicast and unicast destination addresses
  - Option to pass all multicast addressed packets
  - Promiscuous mode to pass all packets without any filtering for network monitoring
  - Pass all incoming packets (as per filter) with a status report
- Additional packet filtering:
  - VLAN tag-based: Perfect match and optional hash-based filtering (filtering can be based on either outer or inner VLAN tag)
  - Layer 3 and Layer 4-based TCP or UDP over IPv4 or IPv6
  - Extended VLAN tag based filtering—32 filter selections
  - Programmable lookup-table-based flexible parser for filtering and steering received packets

46.3 Architecture

The following figure is a system-level block diagram of GMAC.

Beijing CHU Automotive Technology Co., Ltd.
c8c40e5122-4818-4251-b0d3ab0b24c1

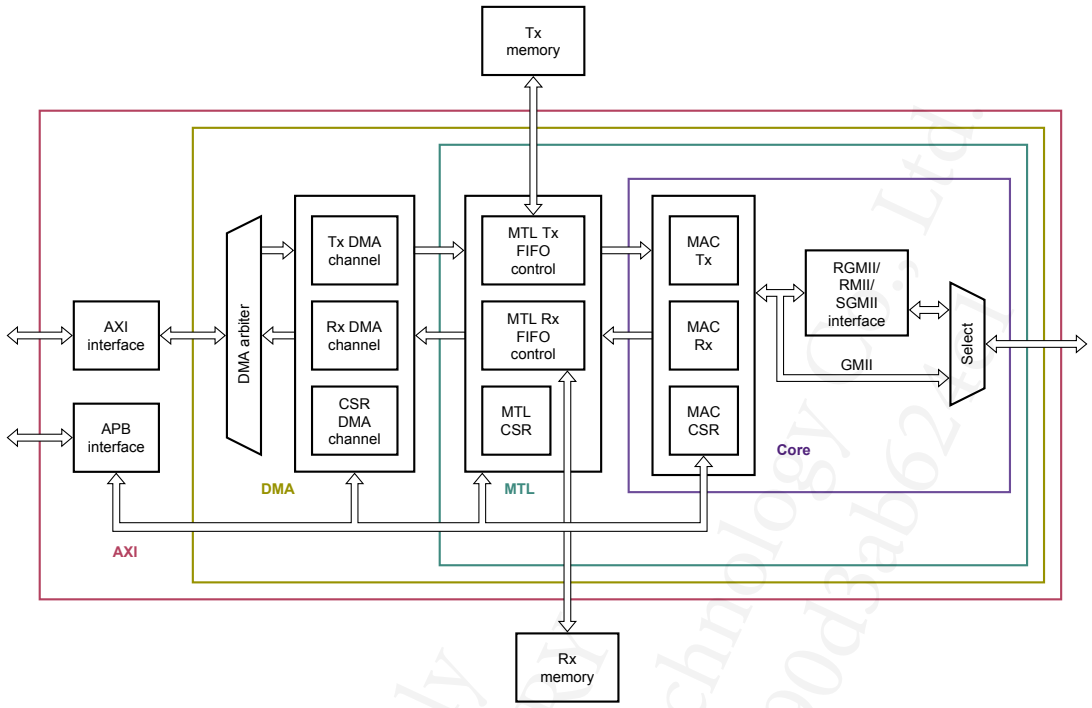


Figure 365. GMAC block diagram

Table 304. Block diagram components

Component	Function	See
AXI interface	High-performance point-to-point interconnect	Interfaces
APB interface	Low-bandwidth interface	—
CSR	GMAC Control and Status Registers (CSRs) provide an interface for software to manage: • Transmit and receive functions • DMA descriptors • Interrupt generation	—
DMA arbiter	Provides arbitration for both transmit and receive for all GMAC DMA channels	—
DMA controller	Manages data transfer operations	DMA controller
Interrupt block (not shown in block diagram)	Manages interrupt functions.	Interrupts
MAC (Core + PHY)	Transmits data to the PHY and provides status.	Media Access Controller (MAC)
MAC Transaction Layer (MTL)	Manages transmit and receive FIFOs.	MAC Transaction Layer (MTL)
PHY interfaces	Supported interfaces include:	Interfaces

Table continues on the next page...

Table 304. Block diagram components (continued)

Component	Function	See
	• MII 10/100 Mbit/s interface • RMII 10/100 Mbit/s interface • RGMII 100/1000 Mbit/s interface • SGMII 1000/2500 Mbit/s through GMII interface (2500 Mbit/s support varies per device).	

46.3.1 Interfaces

GMAC supports the following interfaces:

- AXI
- PHY interfaces.

The AXI interface connects to the chip's AXI master interface. It provides the path for transferring data to and from system memory.

The PHY supports the following interfaces, which you can use alone or in combination:

- Media Independent Interface (MII)
- Reduced GMII (RGMII)
- Serial GMII (SGMII) supported through GMII.

46.3.2 DMA controller

The GMAC DMA controller includes:

- Independent transmit and receive engines:
  - The transmit engine transfers data from system memory to the device port (MTL).
  - The receive engine transfers data from the device port to system memory.
- Control and status registers.

Based on data transfer profiles contained in DMA descriptors, the GMAC DMA controller efficiently moves data from source to destination with minimal intervention from the application core.

46.3.3 MAC Transaction Layer (MTL)

The MAC Transaction Layer (MTL) provides a FIFO memory interface to buffer and regulate packets being transferred between system memory and the MAC. It also enables data to be transferred between the application clock domain and the MAC clock domain. The MTL has two data paths—a transmit path and a receive path. The data path for each can be 64-bit wide.

46.3.4 Media Access Controller (MAC)

The MAC transmits data to the PHY and, when a transmission completes, provides the transmission status to the MTL. When the MAC detects a Start Frame Delimiter (SFD) on the interface, it begins receiving data from the PHY. The MAC also verifies the integrity of the transmitted packet via CRC and drops any packet that fails the CRC check.

46.3.5 Interrupts

Application software can configure GMAC to trigger interrupts based on events captured in status registers. You can independently enable or disable interrupt generation for each interrupt source. GMAC can generate interrupts from events associated with:

- MAC
- MTL
- GMAC DMA.

46.4 Transmit and receive FIFOs

The transmit FIFO buffers data transferred from the application to GMAC. Similarly, the receive FIFO stores packets received from the network interface until they are transferred to the application. Both FIFOs are asynchronous because they also transfer data between the application clock and the MAC line clock domains.

46.5 Flow control

GMAC supports Full-Duplex and Half-Duplex mode packet flow control for transmit operation, and full-duplex packet flow control for receive operation.

Transmit packet flow control involves transmitting *pause packets* in Full-Duplex mode and backpressure in Half-Duplex mode to control the flow of packets from the remote end.

Receive packet flow control is functional only in Full-Duplex mode. GMAC treats any pause packet received as a normal control packet.

46.6 Double VLAN tagging

This feature enables the MAC to process two VLAN tags, and supports:

- Insertion, replacement, or deletion of up to two VLAN tags in the transmit path
- Packet filtering and stripping based on either one of the two VLAN tags in the receive path
- Stripping and providing up to two VLAN tags in the receive path as a part of the receive status.

46.7 Descriptors

The GMAC DMA controller transfers data based on a ring of descriptors. Application software creates the descriptors in system memory. GMAC supports the following types of descriptors:

- A *normal descriptor* is used for packet data and provides control information for packets to be transmitted.
- A *context descriptor* provides control information for a packet to be transmitted.

Each normal descriptor contains two buffers and two address pointers. The buffers enable the adapter port to be compatible with various types of memory management schemes.

46.7.1 Descriptor structure

GMAC supports a ring structure for DMA descriptors as shown in the following figure. The Descriptor Tail Pointer register contains the pointer to the descriptor address (N). The base address of the descriptor ring and the Current Descriptor Pointer determine the address of the current descriptor that the GMAC DMA controller can process. The GMAC DMA controller has ownership of the descriptors up to the location before the one specified by the Descriptor Tail Pointer (N – 1). The GMAC DMA controller continues to process the descriptors until Current Descriptor Pointer == Descriptor Tail Pointer.

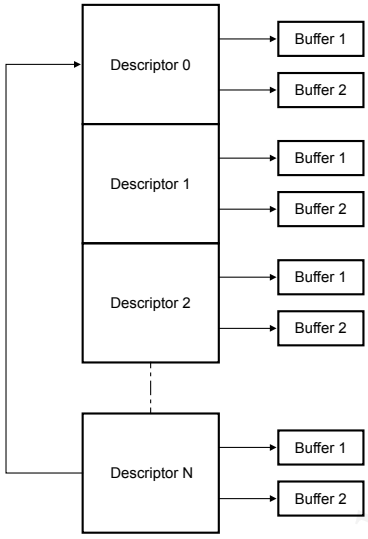


Figure 366. Ring structure

46.7.2 Transmit descriptor

The GMAC DMA controller requires at least one descriptor for a transmit packet. In addition to two buffers, two byte-count buffers, and two address pointers, the transmit descriptor contains fields that control MAC operation on a per-transmit packet basis.

46.7.3 Receive descriptor

The GMAC DMA controller attempts to read a descriptor only if the Descriptor Tail Pointer is not equal to the descriptor ring base address or the Current Descriptor Pointer. We recommend that you have a descriptor ring with a length that can accommodate at least two complete packets received by the MAC. Otherwise, the unavailability of the descriptors significantly degrades DMA performance. In this scenario, the receive FIFO in the MTL becomes full and starts dropping packets.

GMAC supports the following types of receive descriptors:

- Normal descriptors
- Context descriptors

Application software creates all receive descriptors. The GMAC DMA controller treats them as Normal descriptors. The GMAC DMA controller:

1. Reads a descriptor.
2. Transfers all or part of a received packet to the buffers pointed to by the descriptor.
3. Closes the descriptor with the appropriate packet status.

The configuration data within a Normal descriptor is insufficient for writing a complete status for some packets. In this case, the GMAC DMA controller writes the extended status to the next descriptor, without processing the buffers or pointers contained in the descriptor. The format and content of the descriptor is that of a Context descriptor.

46.8 Functional description

Prior to using GMAC functions, application software must perform configuration tasks to ensure proper operation:

- Configure the appropriate GMAC control and status registers for the desired behavior.
- Initialize the receive and transmit DMA descriptors. Among other options, software must provide the descriptors with pointers to the buffers where the GMAC DMA controller will transfer the data to/from.

- When application software transfers data via GMAC, it must manage the DMA descriptors:
  - When GMAC receives one or more packets, the GMAC DMA controller transfers the data to buffers. Software must update the descriptors, including a new buffer address.
  - When application software transmits data, it must:
    - Update the transmit DMA descriptor(s) with the desired buffer addresses.
    - Advance the Descriptor Tail Pointer.
    - Set the appropriate ownership field (also called the "OWN bit") to 1.

    The GMAC DMA controller then begins transferring the data until the Current Descriptor Pointer and Descriptor Tail Pointer values are equal or the descriptor is marked as owned by the application.

During chip operation, application software uses GMAC functions to transfer data between the application and network as detailed below:

- Transmitting data:
  1. The GMAC DMA controller transfers data from the application buffer to the MTL transmit FIFO.
  2. The MAC transfers packets to the PHY via the selected interface—RGMII, for example.
- Receiving data:
  1. When the MAC receives a packet from the PHY, it saves it to the MTL receive FIFO.
  2. The GMAC DMA controller transfers the data from the receive FIFO to the application buffer.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39d13a6214c2

Chapter 47
Packet Forwarding Engine (PFE)

47.1 Introduction

PFE offloads the processing of Ethernet packets from the host cores, yielding higher performance and lower power than pure software processing can achieve. PFE is capable of 2 Gbit/s packet forwarding even at small packet sizes, with virtually no host core load.

47.2 Features

- Three Ethernet ports (PFE_MAC0, PFE_MAC1 and PFE_MAC2)
- Performs the IEEE 802.3 protocol for 10/100/1000
  - MII: 10/100 Mbit/s
  - RMII: 10/100 Mbit/s
  - RGMII: 100/1000 Mbit/s
  - SGMII: 100/1000 Mbit/s
- PFE_MAC0 also supports 2.5 Gbit/s SGMII. When this mode is enabled, PFE_MAC1 must be used in RGMII/RMII/MII mode if enabled.
- Supports packet sizes from 64 bytes up to 1522 bytes
- Capable of autonomously handling all packets belonging to a given stream, without host core intervention, following stream creation
- Capable of performing slow-path or fast-path processing of a packet
- Capable of addressing DDR, security co-processor (HSE_H), and internal on-chip memory
- Capable of routing/bridging an aggregate of 2 Gbit/s of traffic with a 64 byte packet size. Higher rates are achievable for larger packet sizes.
- Supports packet parsing based on standard L2/3/4 header fields or using a flexible parser to take forwarding decisions based on proprietary fields up to 208 bytes from start of frame
- Supports 802.1AS-Rev and IEEE 1588 precision clock synchronization protocol
  - Supports one-step and two-step timestamping
- Full-duplex flow control with recognition of incoming pause frames and hardware generation of transmitted pause frames
- VLAN packet identification and tagging
- Quality of Service (QoS)
  - On transmit, implements flexible configuration of shapers and schedulers
    - Support for credit based shaper algorithm
  - On receive, supports ingress QoS
    - Classification of packets to be managed / unmanaged / reserved traffic
    - Weighted Random Early Drop (WRED)
    - Ingress port based rate shaper
- Four channel DMA host interface for slow path traffic
  - Embedded DMA engine

Freescale NXP Confidential Proprietary
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c20e5 482-48 8125139 0d3ab624c1

  - Interrupt per channel
  - Supports interrupt coalescing
- Independent MDIO interface per MAC

47.3 Initialization

PFE is a firmware-based module. Load the firmware before using PFE. The PFE cores boot during initialization by the host core. After initialization completes, PFE moves to Normal mode and is ready for full operation. The NXP-provided host interface driver and the PFE firmware abstract the initialization.

47.4 Functional description

PFE is based on programmable packet processing engines (PE).

PFE implements a fast-path/slow-path architecture. Fast-path packets can be completely offloaded from host core processing. Slow-path packets (such as control packets) can handle packet processing on the host cores.

The classification module classifies and performs header modification tasks, such as network address translation (NAT). The classifier performs traffic management and also maintains packet counters for statistics.

47.5 Packet parsing

The classifier has many different options for parsing of the packets such as:

- VLAN
- Multicast
- Broadcast
- ARP
- IPv4
- IPv6
- UDP
- TCP
- ICMP

The parsing block parses each packet and uses a handshake mechanism to provide inputs to the PFE processing engines. Firmware then processes the packet routing.

47.6 Layer 2 parsing

The first step in the packet classifier module is the Layer 2 classification. The link layer headers (VLAN, and so on) of the packet are parsed and information is saved into the packet buffer's header structure. If the link layer header is unknown, the packet is marked with a special flag and sent to the slow path for proper processing. After parsing, the packet is sent to bridge processing if bridging is enabled. Otherwise, the packet is sent to route processing.

47.7 Bridge classifier

This module performs the following actions:

- Processes the packet after Layer-2 parsing
- Uses bridge groups and port related information
- Uses a bridge MAC address table for MAC destination address lookup to forward the packets onto transmit bridge ports
- Performs lookups of the source MAC address for learning and aging support

- Sends the packet to the slow path for processing if no matched entries are found as part of the lookup.

47.8 Layer 3 parsing

In route mode, the module looks into Layer 3 and Layer 4 protocol headers of the packet and updates the packet buffer header structure with the parsed information. The route classifier module uses the flow information extracted from the packet to decide which action to take. This module also performs:

- IP version check
- IP packet length check
- IP checksum verification
- Fragment packet identification
- Unknown layer 4 protocols
- TCP flags (SYN, FIN, RST)
- Identification of IPSec tunnel and transport headers

47.9 Route classifier

This module performs the following actions on the packet:

- Hash calculation using specific header fields
- Lookup in the route hash table to match with flow entry
- If no entry is matched, then send the packet to host for slow-path processing
- TTL decrement and increment checksum update
- Perform NAT processing if required
- Construct TX port related link layer headers
- Send the packet to the traffic manager unit

47.10 Queuing block

The queuing block:

- Takes packets and schedules them for the processing blocks
- Moves packet headers from internal/external memory to the PE data memory
- Tags the packets with a sequence number that is used to maintain packet ordering within the module

47.11 Traffic manager unit (TMU)

TMU implements the QoS functionality in the egress packet processing chain. The classifier sends the packets to TMU after the packet is classified and modified. The classifier determines the queue mapping for the packet. The module works closely with the peripheral blocks and schedules the packets for transmission. The traffic manager has programmable queuing disciplines (WRR, WFQ or PQ) for implementing QoS.

TMU has weights (and credits) for guaranteed service and for distribution of the available bandwidths with the following features:

- Eight classes queues per port
- Programmable weights, scheduler tick, and max credits per class

In the case of multicast traffic, multiple references to the same buffer are written to TMU, and TMU processes them independently. TMU contains four shaper units that implement a credit-based shaper algorithm. Association of a shaper to a queue/scheduler is not fixed in hardware; this is maintained by software.

47.12 Buffer management

The buffer management module allocates and de-allocates buffers for the packets. Its structure is very generic and amenable to any operating system. The buffer allocation and freeing is managed within the buffer pool without calling time-consuming memory-allocation and memory-freeing functions for every packet.

47.13 Host Interface (HIF)

The chip host cores transmit and receive packets via the PFE HIF. HIF has a built-in DMA controller that transfers the slow-path packets to/from the host cores. HIF supports chained buffer descriptors (BD). The host software prepares the buffer descriptors, allocates associated buffers, and chains the descriptors. Host software ensures that invalid buffer descriptors have their descriptor enable bit set to zero so that they are not processed erroneously. The end of the BD chain always has a BD with a descriptor-enable bit equal to 0. The host interface has four virtual channels to allow distribution of packets across multiple host cores. Each virtual channel has its own RX/TX buffer descriptor rings and also a dedicated interrupt signal to generate events on the host cores. See the chip interrupt mapping for details.

The interrupt line of channel 3 (the last channel) connects directly to LLCE. You can therefore configure that channel to trigger events to the host core(s) and/or LLCE.

HIF implements interrupt coalescing. You can enable coalescing based on frame count and on a timer. If you enable only timer-based coalescing, PFE generates a coalesce interrupt after the timer reaches its value. If you enable only frame-based coalescing, PFE generates a coalesce interrupt after the number of interrupts equals the frame counter. If you enable both types of coalescing, PFE generates an interrupt when either of the previously-mentioned conditions is true.

47.14 PHY interface options

You have various pin-muxing options available to support a wide range of PHY interface modes and port numbers. For details on pin multiplexing options, see the IOMUX spreadsheet attached to this document.

47.15 PFE configuration

The software solutions described in the following sections abstract the PFE programming.

These software solutions allow configuration of the following features:

- IPv4/IPv6 forwarder/router
  - NAPT
  - IPSec offload
  - Configuration API
- L2 bridging
  - VLAN support
  - VLAN table control API
  - MAC table control API
  - Port control API
- QoS control API
- Statistics

47.16 Software components

The PFE driver loads into the OS networking stack and provides access to PFE Ethernet interfaces. The driver supports basic networking functions such as:

- Frame transmission

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39023ab62

- Frame reception
- Link status monitoring

Offload functions, including forwarding of Ethernet packets without host core intervention and QoS, are also implemented as part of the API.

For further details on the PFE software support (CMM and LibFCI), see the PFE software product brief and Fast Control Interface (FCI) API reference documents.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 48
LINFlexD

48.1 Chip-specific LINFlexD information

48.1.1 Number of LINFlexD filters and channels

There are three instances of LINFlexD on the chip.

The following table shows the numbers of filters and channels for each instance of LINFlexD.

Description	LINFlexD_0	LINFlexD_1	LINFlexD_2
Number of filters (no_of_filters)	0	0	0
Number of Tx DMA channels ($2^{TX\_CH\_NUM}$)	1	1	1
Number of Rx DMA channels ($2^{RX\_CH\_NUM}$)	1	1	1
Slave mode enabled	1	1	1
UART mode enabled	1	1	1
Auto-synchronization enabled	0	0	0
DMA support	Yes	Yes	Yes

48.1.2 LIN_CLK and bus clock relationship

PBRIDGEx_CLK = LIN_CLK/2

48.2 Introduction

The LINFlexD controller is designed to manage a high number of LIN messages efficiently with a minimum of CPU load. To reduce the CPU loading in Master mode, LINFlexD autonomously handles the LIN messages after the software has triggered the header transmission, until the next header transmission request in transmitter mode or until checksum reception in the receiver mode.

LINFlexD:

- Supports LIN protocol version 1.3, 2.0, 2.1
- Provides an 8-byte buffer for transmission/reception data
- Supports some of the basic UART transfers of 8-bit, 9-bit frames
- Supports multi channels and parametric DMA Tx/Rx interface in LIN/UART operating mode

48.2.1 Block diagram

The following figure shows the LINFlexD block diagram.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-481-48f-b251-390d3ab624c1

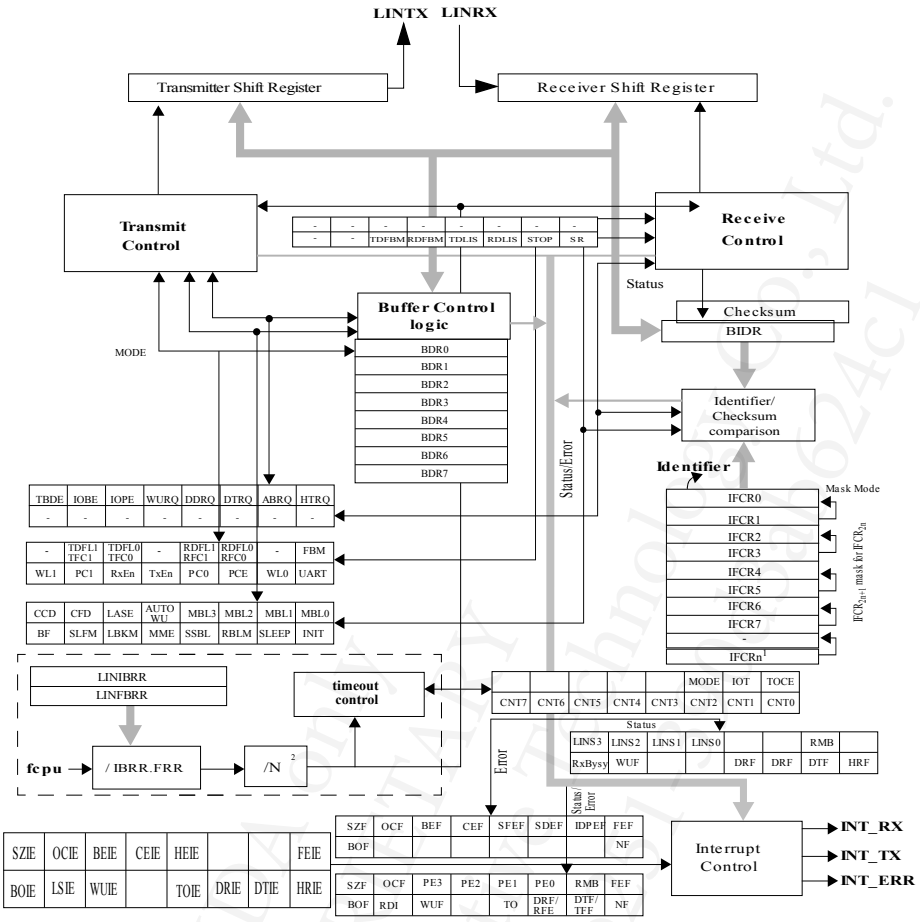


Figure 367. Block diagram

Notes:

1. The value of n depends on the no_of_filters. See the chip configuration details to see the number of filters used in this device.
2. For LIN mode N = 16, N is programmable by user in UART mode.
3. FBRR is not usable when reduced over sampling is enabled in UART mode.

48.3 Main features

The LINFlexD controller can operate in several modes, each of which has a distinct set of features that are described in the following sections. In addition, the LINFlexD controller has several features common to all modes:

- Fractional baud rate generator
- Three operating modes for power saving and configuration registers lock
  - — Initialization
  - — Normal
  - — Sleep
- Test mode: Loop Back
- Maskable interrupts

48.3.1 LIN mode features

- Supports LIN protocol versions 1.3, 2.0, 2.1
- Bit rates up to 20 Kbit/s (LIN protocol)
- Master/Slave modes
- Classic and Enhanced Checksum calculation and check
- Single 8-byte buffer or FIFO for transmission/reception
- Timeout management
- DMA interface
- Master mode with autonomous message handling
- Wakeup event on dominant bit detection
- True LIN field state machine
- Advanced LIN error detection
- Header, response, and frame timeout
- Slave mode
  - Autonomous header handling
  - Autonomous transmit/receive data handling

48.3.2 UART mode features

- Full-duplex communication
- Baud rate is a function of baud clock, LINIBRR and LINFBRR registers. See Baud rate generation for details.
- 7/8 bits data, parity
- 1/2/3 stop bits
- 4-byte buffer for reception; 4-byte buffer for transmission
- 12-bit counter for timeout management
- The maximum baud rate achievable is LIN_CLK/4 Mbit/s.
- For bit rate ≤ LIN_CLK/16 Mbit/s
  - 16 times oversampling
  - 3:1 majority voting
- For LIN_CLK/16 Mbit/s < bit rate ≤ LIN_CLK/8 Mbit/s
  - User-programmable reduced oversampling
  - 3:1 majority voting for reduced over sampling of 8 samples per bit
- For LIN_CLK/8 Mbit/s < bit rate ≤ LIN_CLK/4 Mbit/s
  - User-programmable reduced oversampling
  - 1:1 voting for all reduced over sampling of 4, 5, and 6 samples per bit

Processed under NDA only
COMPANY PROPRIETARY
Beijing eCARX Automotive Technology Co., Ltd.
c8c4e25-4821-48a8-b251-390d3ab624c1

48.4 Functional description

48.4.1 LIN protocol

LIN is a serial communication protocol. A LIN cluster consists of one master task and several slave tasks. A master node contains the master task as well as a slave task. All other nodes contain a slave task only. The master node decides when and which frame is transferred on the bus. The slave task provides the data to be transported by the frame.

48.4.1.1 Frames

A frame consists of a header provided by the master task and a response provided by the slave task. The header consists of a break, a sync pattern, and an identifier. The break is followed by the sync pattern and the sync pattern is followed by the identifier. The slave task associated with the identifier provides the response. The response consists of a data field and checksum. The slave task designated to receive the data associated with the identifier receives the response and verifies the checksum.

Transmittion or reception of LIN frames occurs as shown in Figure 368 and Figure 369.

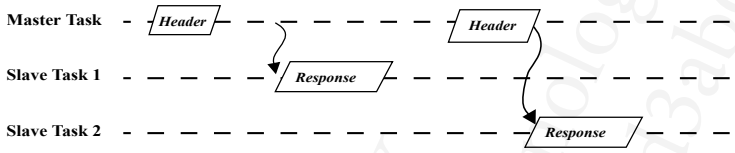


Figure 368. Frames

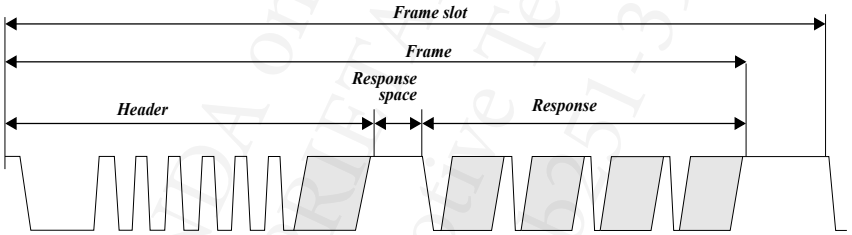


Figure 369. Structure of LIN frame

48.4.1.2 Data field

Nodes and tasks transmit each byte as shown in Figure 370. The LSB of the data is sent first and the MSB is sent last. The start bit is encoded as a bit with value zero (dominant) and the stop bit is encoded as bit value one (recessive).

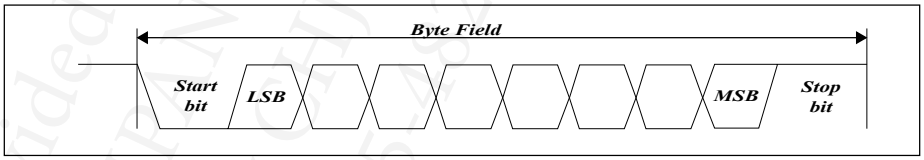


Figure 370. Structure of byte field

48.4.1.3 Break

The break symbol is used to signal the beginning of a new frame. It is the only field that does not comply with the above figure. Only the master generates the break. The break must be at least 13 bits of dominant value including the start bit, followed by a break delimiter as shown in Figure 371. The break delimiter must be of two-bit duration (to be compliant with LIN protocol 2.1).



Figure 371. Break field

48.4.1.4 Sync byte

Sync is a byte field with the data value of 55h.

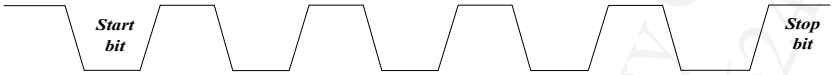


Figure 372. Sync byte field

48.4.1.5 Identifier

The identifier field consists of two sub fields, the identifier and the identifier parity. Bits 0 to 5 are the identifier and bits 6 and 7 indicate the parity.



Figure 373. Identifier field

P0 = ID0 XOR ID1 XOR ID2 XOR ID4

P1 = NOT (ID1 XOR ID3 XOR ID4 XOR ID5)

48.4.1.6 Checksum

The last field of a frame is the checksum. The checksum contains the inverted 8-bit sum with carryover of all data bytes and the identifier. The following table shows the types of checksums.

Table 305. Checksum types

Type	Bytes used	Use
Classic	Only the data bytes	Communication with LIN 1.3 slaves
Enhanced	Data bytes and protected identifier byte	Communication with LIN 2.0 slaves

48.4.2 LINFlexD features

48.4.2.1 Operating modes

The LINFlexD includes three operating modes: Initialization, Normal, and Sleep. After hardware reset, LINFlexD enters the Sleep mode to reduce power consumption. Figure 374 shows the state diagram of operating modes.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca462e2-4821-48a8-b251-3904d3ab624d1

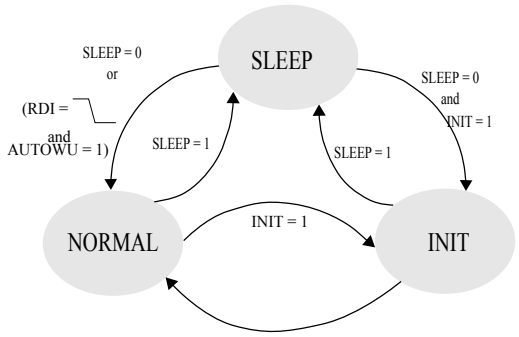


Figure 374. Operating modes

Initialization mode (INITM)

To enter this mode, software writes a 1 to LINCR1[INIT]. To exit INIT, the software must write a 0 to LINCR1[INIT].

When in INIT, LINFlexD stops all message transfers to and from the LIN bus and the status of LIN bus output LINTX is recessive (high). Use this mode only to configure the IP. Your software must not initiate a data transmission in INITM.

If software invokes the INITM when a bus transfer is in progress, the LINFlexD aborts the transfer. Therefore, your software must check the LIN state before writing a 1 to LINCR1[INIT].

To initialize the LINFlexD controller, your software must:

1. Set up the baud rate registers
2. Write a 0 to UARTCR[UART].
3. Select the mode (Master or Slave)
4. Configure checksum control bits

Normal mode (NM)

After the software initializes LINFlexD, the software can enter NM by writing a 0 to LINCR1[INIT].

Sleep mode (SM)

This mode reduces power consumption. To enter this mode, write a 1 to LINCR1[SLEEP]. In this mode, the LINFlexD clock stops. You can awaken LINFlexD from SM by writing a 0 to LINCR1[SLEEP].

If software detects a wakeup pulse of 150 µs on the LIN bus, it can request LINFlexD to wake up from SM. See Wakeup management.

48.4.2.2 Test mode

Loop Back mode

You can use this mode for self-test functions.

Enter this mode by writing a 1 to LINCR[LBKM].

In this mode, LINFlexD:

- Receives the identifier and data that LINFlexD itself transmits
- Writes the identifier and data to the BIDR and data buffers, respectively
- Checks bit errors

- Ignores the LINRX signal
- Feeds back its TX output to its RX input

You can disconnect the TX pin from the LINTX pin using a setting in SIUL2.

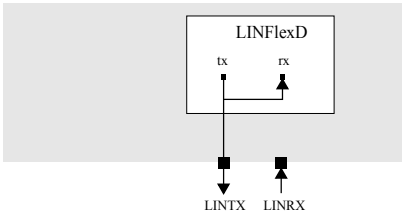


Figure 375. LINFlexD in Loop Back mode

48.4.2.3 Master mode

Enter Master mode by writing a 1 to LINCR1[MME].

48.4.2.3.1 Header transmission

According to the LIN protocol, the master triggers any communication on the LIN bus by sending a header. The master task of a node transmits the header. The slave task of a node transmits the response.

To transmit the header:

1. Use BIDR to program:
   - The identifier
   - Data field length
   - Message direction
   - Checksum enable
2. Write a 1 to LINCR2[HTRQ]

After header transmission starts, do not modify BIDR fields until the current frame completes.

The master node receives the transmitted ID and copies it to BIDR.

48.4.2.3.2 Data transmission

Consider the scenario in which the master sends an identifier and the master node publishes the data corresponding to that identifier. In this scenario, the node's slave task must send the data in the response part of the frame. Therefore, your software must provide the data to LINFlexD before LINFlexD requests the header transmission.

LINFlexD stores the to-be-transmitted data in the BDRL and BDRM message buffers. The number of to-be-transmitted bytes depends on BIDR[DFL]. Your software must use BIDR[CCS] to configure the checksum type (classic or enhanced) for each message.

If LINFlexD successfully sends the response, LINFlexD writes a 1 to LINSR[DTF]. If an error occurs, LINFlexD:

- Does not write a 1 to LINSR[DTF]
- Writes a 1 to the corresponding error flag in LINESR (see Errors for details)

BIDR[DIR] determines the direction of the message buffer. The same node also receives the transmitted data and copies the data to the buffer.

Preliminary Information under NDA only
COMPANY PROPRIETARY
Beijing CUI Automotive Technology Co., Ltd.
ce46e8e7-214a8-b251-390d3ab624c1

48.4.2.3.3 Data reception

To receive data from a slave node, the master sends a header with the corresponding Identifier. LINFlexD then:

- Receives data from the slave
- Stores the data in the message buffer
- Stores the status of the message in LINESR

If LINFlexD successfully receives the response, LINFlexD writes a 1 to LINSR[DRF]. If an error occurs, LINFlexD:

- Does not write a 1 to LINSR[DRF]
- Writes a 1 to the corresponding error flag in LINESR (see Errors for details)

48.4.2.3.4 Data discard

If you want to discard the data after header transmission, write a 1 to LINCR2[DDRQ].

48.4.2.4 Slave mode

LINFlexD enters this mode when you write a 0 to LINCR1[MME].

48.4.2.4.1 Data transmission

After receiving a header, LINFlexD:

- Writes a 1 to LINSR[HRF]
- Generates an RX interrupt

The software must then:

1. Read the received ID in BIDR.
2. Fill the BIDR.
3. Program the CCS and DIR fields in BIDR.
4. Write the data field length to BIDR[DFL].
5. Write a 1 to LINCR2[DTRQ] to trigger data transmission.

Note that the HRF bit should be reset only after the DTRQ bit is set. For the DTRQ to be effective, the HRF bit must be set. This is to ensure that DTRQ is not set randomly, but only after a header reception. It must be noted that you cannot set the DIR and DTRQ bits after RXBUSY is asserted in LINSR.

The same node also receives the transmitted data and copies it to the buffer.

48.4.2.4.2 Data reception

When LINFlexD is the subscriber of the data of the received identifier, and LINFlexD receives a header, LINFlexD:

- Writes a 1 to LINSR[HRF]
- Generates an RX interrupt

The software must read the received ID from BIDR and specify the data field length before the reception of the stop bit of the first data byte. When the checksum is received, an RX interrupt is generated for software to read the received data from BIDR and the RMB bit is set. Software must then release the data buffer by writing a 1 to LINSR[RMB].

If your software filters the identifier, you can discard the data by writing a 1 to LINCR2[DDRQ] while the value of LINSR[HRF] =1.

For software filtering, software must choose the checksum type (BIDR[CCS]) before receiving data. Otherwise, the checksum calculation uses the previous value of BIDR[CCS].

Document under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c04e5482-4ca8-b251-390d3ab624c1

48.4.2.5 Errors

48.4.2.5.1 Header error

A header error is an error during the header reception. The error types are:

- Sync delimiter error (SDEF)
- Sync field error (SFEF)
- Identifier parity error (IDPEF)

48.4.2.5.2 SDEF

The delimiter must be 1 for at least one bit time; otherwise it is considered short and consequently the receiver discards synchronization on the current header. Therefore LINFlexD discards the frame. LINFlexD also generates an interrupt if LINEIER[HEIE]=1.

48.4.2.5.3 IDPEF

The identifier field has two parity bits. LINFlexD checks these bits against the hardware-calculated parity over the other six identifier bits according to these equations:

P0 = ID0 XOR ID1 XOR ID2 XOR ID4

P1 = NOT (ID1 XOR ID3 XOR ID4 XOR ID5)

LINFlexD checks this parity after transferring the ID to BIDR. If LINCR2[IOPE]=1, and parity mismatch occurs, the receiver state machine exits the identifier state immediately.

48.4.2.6 Bit error

LinFlexD flags this error in transmission mode when the value read back from the bus is different from the value transmitted. Bit error checking on each bit is guaranteed if the transceiver delay is less than one bit time minus 6 LIN_CPU cycles. LINFlexD does not check for bit errors.

1-bit time at 20 Kbit/s = 50 µs

6 LIN_CLK cycles at 80 MHz = 75 ns

Thus, (1-bit time) - (6 LIN_CLK cycles) = 49.925 µs

If LINCR2[IOBE]=1, LINFlexD stops transmitting the frame after the corrupted bit. If LINCR2[IOBE]=0, LINFlexD continues to transmit despite the bit error. LINFlexD generates an interrupt if LINIER[BEIER]=1.

If the master does not detect the break delimiter within one bit time after delimiter transmission (for example, because of transceiver delay or bus error), LINFlexD generates a train of bit-error interrupts. In this case, LINFlexD may not replace the identifier in BIDR. Similarly, if the transmitter node does not detect the start of a falling edge of data within one bit time after transmission of the start bit, LINFlexD generates a train of bit-error interrupts. In this case:

- Data replacements in the BIDR are not guaranteed.
- Checksum replacements in the LINCFR are not guaranteed.

48.4.2.7 Framing error

LINFlexD flags this error when it samples a dominant state on the stop bit of the current received character (Sync field, identifier field, data field, or checksum field). LINFlexD discards the current frame and returns to the Idle state. LINFlexD generates an interrupt if LINEIER[FEIE]=1.

LINFlexD also takes the byte that caused the framing error and shifts it to the buffer, but does not write a 1 to LINSR[DRF].

48.4.2.8 Checksum error

LINFlexD flags this error when the checksum computed by hardware does not match the received checksum.

LINFlexD discards the received frame and returns to the Idle state. LINFlexD generates an interrupt if LINEIER[CEIE]=1.

48.4.2.9 Overrun error

After the message buffer is full (LINSR[RMB]=1), the next valid message reception leads to an overrun and the message is lost. LINFlexD signals the overrun condition by writing a 1 to LINESR[BOF]. The buffer lock function control (LINCR1[RBLM]) determines which message is lost.

If RBLM=0, the old message in the buffer is overwritten by the most recent message. If RBLM=1, the most recent message is discarded and the oldest message is available to the software. In the case of master, during overrun condition, internal state machine moves to idle state and any transition from 1 to 0 is considered as a wakeup state till the end of current transfer. Under this scenario, the software must entertain only one WUF interrupt and ignore the rest. In the case of slave, if buffer is not released (RMB is not reset) before reception of next Identifier and if RBLM=1, then the ID along with the data is discarded.

48.4.2.10 Timeout error

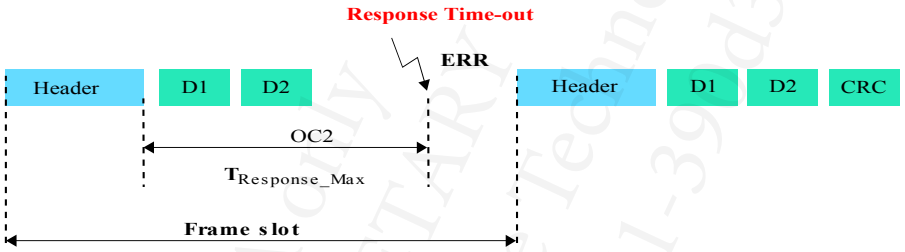


Figure 376. Incomplete response (for example, missing checksum)

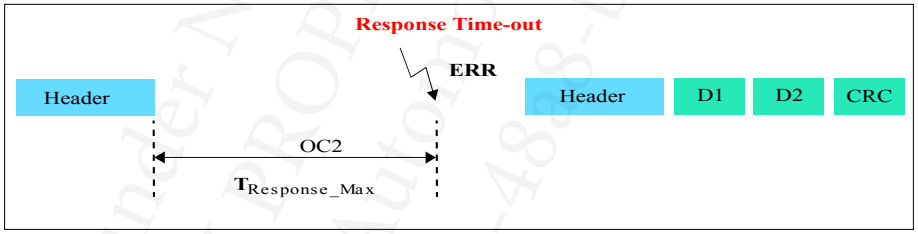


Figure 377. No response

48.4.2.10.1 Response timeout mechanism

Master mode

OC2 is loaded with 1.4 x ((DFL+2)x(10 bit times)) + LINTCSR[CNT] at the end of the stop bit of the Identifier field. This equation is defined according to the reset values of LINTOCR[RTO] and LINCR1[CFD], and it changes with any change in the values of these fields. Here, 1.4 corresponds to (LINTOCR[RTO])/10 and 2 corresponds to (2- LINCR1[CFD]).

This loading takes place at the end of the Identifier field (and not at the beginning of Data field). Moreover, the correct value, which depends on DFL, is loaded immediately. No further OC2 update is required.

In case of no response at all within this time (in other words, start of data is not received), timeout takes place when the counter reaches the OC2 value (no change is compared to earlier implementation).

In case of an incomplete response, timeout also occurs when the counter reaches the OC2 value (no change compared to earlier implementation).

Slave mode

- As the DFL value needs to be updated by software after the identifier field has been received, the implementation is the following:
  - At the end of the ID, OC2 is loaded with 36 (maximum possible response space) + LINFlexD_LINTCSR[CNT].
  - At the end of the first_data_byte it is reloaded again according to DFL
  - Before reloading, LINFlexD checks the count_val. If count value is higher than the value to be reloaded, timeout takes place immediately and no reloading occurs.

48.4.2.10.2 Header timeout mechanism

Master mode

Because LINFlexD generates the header, there are only two cases:

- No error on the bus and timing is correct (nominal header length).
- An error (typically a bit error) occurs on the bus and LINFlexD flags it in LINESR.

Therefore there is no meaning of header timeout in master mode, so it is disabled.

Slave mode

- header_nominal = 13 + 2 + 10 + 10 = 35 Tbit.
- header_max = 1.4 × Header_nominal = 49 Tbit.
- Taking into account a possible 14% clock deviation, header_max seen by LINFlexD is 49 × 1.14 = 56 Tbit..
- If the counter starts after 11 Tbit (break duration), the HTO value is 56 –11 = 45 Tbit.

The reset value of HTO is 45, and this register can only be programmed in Slave mode. Counter restarts after break duration and OC1 gets loaded with the value of HTO.

As response space is not included in the response, frame timeout is no longer needed and is removed completely. Indeed, header timeout and response timeout cover all cases.

The timeout counter can be used to detect other timeouts. In this case, the MODE bit must be reset and the output compare value can be updated in the LINTOCR Register by software.

Stuck at zero timeout error

If the dominant pulse lasts for a time of at least 100 bits, LINFlexD writes a 1 to LINESR[SZF]. If the same dominant pulse prolongs, the subsequent SZF setting is 87 (instead of 100) bit times apart.

48.4.2.11 Noise

During reception each bit is sampled 16 times and the value of the bit is obtained by taking the majority value of the 8^{th}, 9^{th} and 10^{th} samples. If any one of these three samples has a value different from the other two, this error is flagged.

When UARTCR[OSR] = 8, majority of 2nd, 3rd, and 4th samples are taken into account to determine noise.

When UARTCR[OSR] < 8, noise checking is disabled.

This error is flagged when there is noise detected in the start bit (see LIN Error Status Register (LINESR)).

48.4.2.12 Start detection and break delimiter detection in receiver

There is a 10-bit-shift register sample register in the receiver that shifts signal data to the next least significant bit on each incoming sample.

Therefore, a start is detected as soon as it is qualified with 1110 in the sample register and verified with at least two out of three predefined verification samples being zero. Similarly for delimiter detection, qualification samples are 0001 and at least two of the three verification samples are one. The Noise flag is set if the start is verified with only two valid samples.

Table 306. Start detection and delimiter detection in receiver

9	8	7	6	5	4	3	2	1	0	Sample_reg
—	—	—	—	—	—	—	—	—	RT1	counter = 0
—	—	—	—	—	—	—	—	RT1	RT2	counter = 1
—	—	—	—	—	—	—	RT1	RT2	RT3	counter = 2
—	—	—	—	—	—	RT1	RT2	RT3	RT4	counter = 3
—	—	—	—	—	RT1	RT2	RT3	RT4	RT5	counter = 4
—	—	—	—	RT1	RT2	RT3	RT4	RT5	RT6	counter = 5
—	—	—	RT1	RT2	RT3	RT4	RT5	RT6	RT7	counter = 6
—	—	RT1	RT2	RT3	RT4	RT5	RT6	RT7	RT8	counter = 7
—	RT1	RT2	RT3	RT4	RT5	RT6	RT7	RT8	RT9	counter = 8
RT1	RT2	RT3	RT4	RT5	RT6	RT7	RT8	RT9	RT10	counter = 9
RT2	RT3	RT4	RT5	RT6	RT7	RT8	RT9	RT10	RT11	counter = 10
RT3	RT4	RT5	RT6	RT7	RT8	RT9	RT10	RT11	RT12	counter = 11
RT4	RT5	RT6	RT7	RT8	RT9	RT10	RT11	RT12	RT13	counter = 12
RT5	RT6	RT7	RT8	RT9	RT10	RT11	RT12	RT13	RT14	counter = 13
RT6	RT7	RT8	RT9	RT10	RT11	RT12	RT13	RT14	RT15	counter = 14
RT7	RT8	RT9	RT10	RT11	RT12	RT13	RT14	RT15	RT16	counter = 15
1	1	1	0	x	0	x	0	x	0	start detect
1	1	1	0	x	0	x	0	x	1	
1	1	1	0	x	0	x	1	x	0	
1	1	1	0	x	1	x	0	x	0	

Table continues on the next page...

Table 306. Start detection and delimiter detection in receiver (continued)

9	8	7	6	5	4	3	2	1	0	Sample_reg
0	0	0	1	x	1	x	1	x	1	delimiter detect
0	0	0	1	x	1	x	1	x	0	
0	0	0	1	x	1	x	0	x	1	
0	0	0	1	x	0	x	1	x	1	

48.4.2.12.1 Start detection mechanism

Here are the steps for detecting a start bit:

1. See Table 306 for the status of the sample register (shift register) when count = 6
2. At this point, if RT1 (the first incoming sample) = 0 and the previous three samples already received are all ones, then it might be a start bit.
3. To make sure it is indeed a start bit, the incoming data samples in the sample register (4), sample register (2) and sample register (0) which correspond to RT3, RT5, and RT7 is verified using the steps mentioned below.
4. If the majority (that is, 2) out of these 3 samples are equal to 0, then start bit is said to be detected.

   These three samples, RT3, RT5, and RT7 are called the “verification samples” that are checked at count = 6. The sample register bits 9, 8, 7, and 6 are called “qualification samples” that are checked at count = 6.
5. At count = 9, if majority value of RT8, RT9, and RT10 is not equal to 0 then Noise flag is set.

48.4.2.12.2 Break delimiter detection mechanism

Here are the steps for detecting a delimiter bit:

1. See Table 306 for the status of the sample register (shift register) when count = 6
2. At this point, if RT1 (the first incoming sample) = 1 and the previous three samples already received are all zeros, then it might be a delimiter bit.
3. To make sure it is indeed a delimiter bit, the incoming data samples in the sample register (4), sample register (2) and sample register (0) which correspond to RT3, RT5, and RT7 is verified using the steps mentioned below.
4. If the majority (that is, 2) out of these 3 samples are equal to 1, then delimiter bit is said to be detected.
5. These three samples RT3, RT5, and RT7 are called the “verification samples” that are checked at count = 6.
6. The sample register bits 9, 8, 7, and 6 are called “qualification samples” that are checked at count = 6.

Therefore, a start is detected as soon as it is qualified with 1110 in the sample register and verified with at least two out of three predefined verification samples being zero. Similarly for delimiter detection, qualification samples are 0001 and at least two out of the three verification samples are one. The Noise flag is set if the start is verified with only two valid samples.

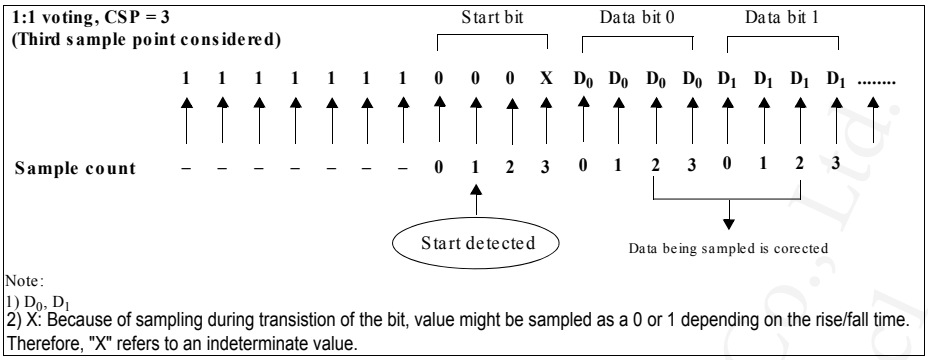


Figure 378. Start detection and sampling for oversampling rate = 4 (Case 1)

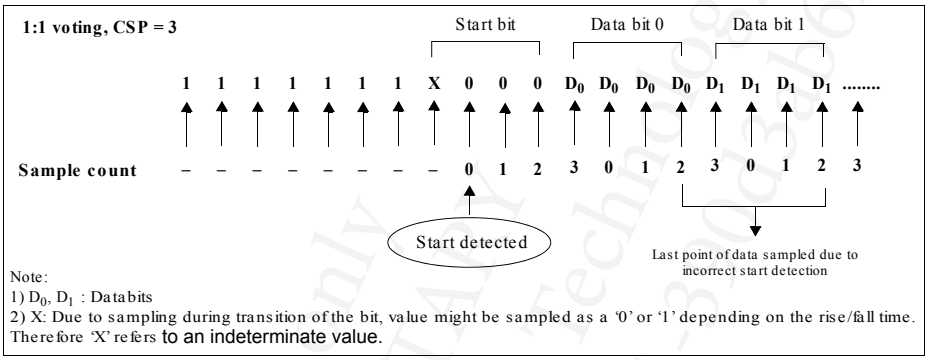


Figure 379. Start detection and sampling for oversampling rate = 4 (Case 2a)

The choice of the correct sample points depends on the external Rx signal quality. During the application development process, the error information indicated by the parity error can help to find the best setting for an application-specific hardware signal.

Therefore, to sample at the correct point, you must write a 2 to CSP, resulting in the following situation:

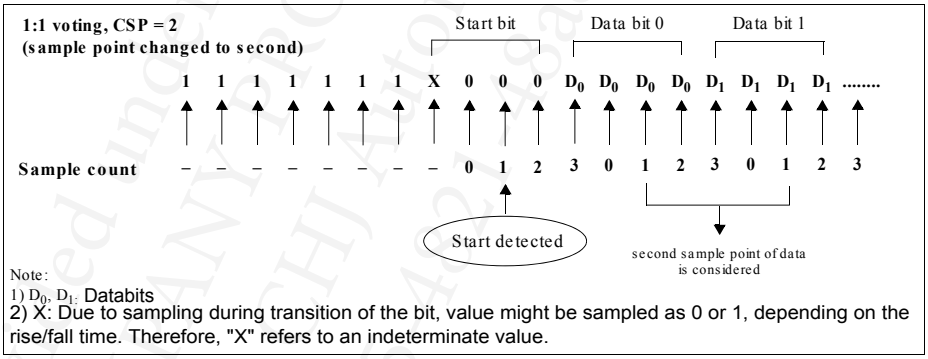


Figure 380. Start detection and sampling for oversampling rate = 4 (Case 2b)

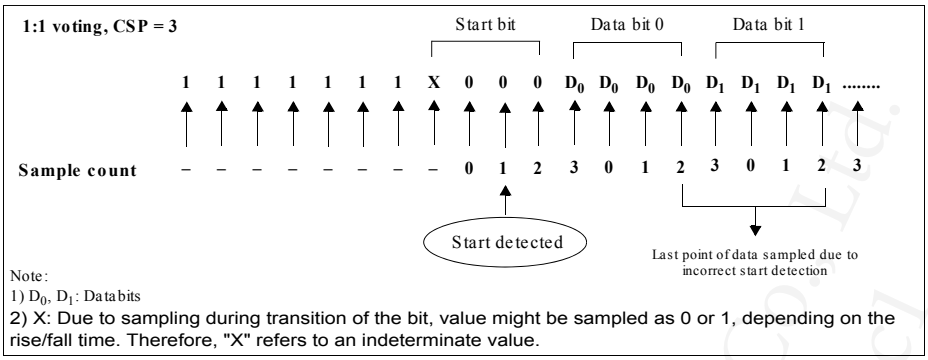


Figure 381. Start detection and sampling for oversampling rate = 4 (Case 3)

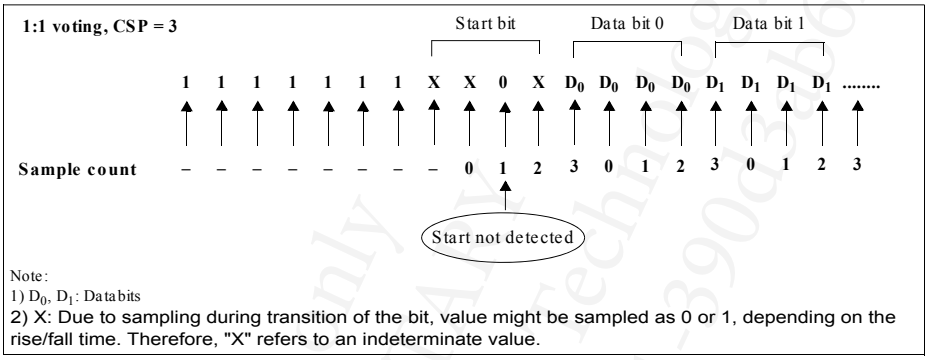


Figure 382. Start detection and sampling for oversampling rate = 4 (Case 4)

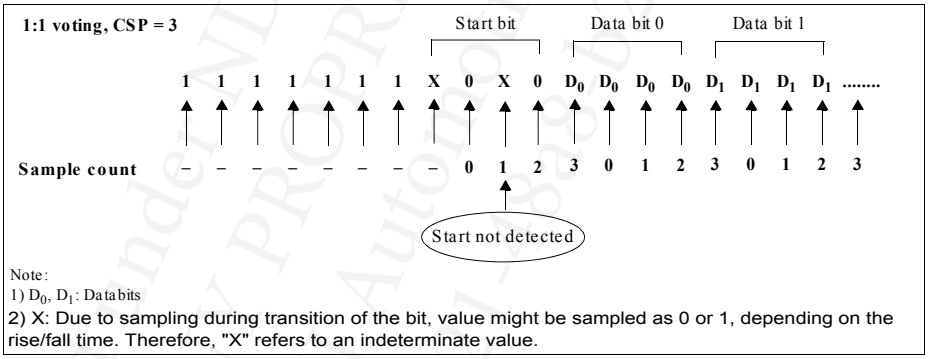


Figure 383. Start detection and sampling for oversampling rate = 4 (Case 5)

48.4.2.13 Baud rate generation

LINIBRR and LINFBRR control the LIN baud rate. You can program these registers only during INITM.

Baud rate is calculated with the following formula for both receiver and transmitter.

When UARTCR[ROSE] = 1, Tx = Rx = LIN_CLK / (OSR × LDIV). When UARTCR[ROSE = 0, Tx = Rx = LIN_CLK / (16 × LDIV), where LIN_CLK is the frequency of the baud clock.

LDIV is an unsigned fixed point number. The mantissa is coded into 20 bits of LINIBRR[IBR] and the fraction is coded on 4 bits of LINFBRR[FBR].

When reduced oversampling is enabled, LINFBRR must not be used and programmed to zero, and LDIV contains only the integer part of LINIBRR.

Table 307. Examples of baud rate calculations

UARTCR1[ROSE]	Mode(s)	LDIV	LIN_CLK	LINIBRR[IBR]	LINFBRR[FBR]	Baud rate
ROSE = 0	LIN and UART	468.75 d	36 MHz	468 d	12	36 MHz/ (16 × 468.75) = 4.8 Kbit/s
ROSE = 1	UART	5 d	80 MHz	5 d	4	LIN_CLK / (OSR × LDIV) = 80 MHz / (4 × 5) = 4 Mbit/s

48.4.2.14 Wakeup management

Any node in a sleeping LIN cluster can request a wakeup. The node issues the wakeup request by forcing the bus to the dominant state for 250 µs to 5 ms. Every slave node must detect the wakeup request (a dominant pulse longer than 150 µs) and be ready to listen to bus commands within 100 ms, measured from the ending edge of the dominant pulse. The master also wakes after detecting a wakeup request, and when the slaves are ready, starts sending frame headers to find out the cause of the wakeup. If the master does not issue a frame header within 150 ms from the wakeup request, then the requesting node can try issuing a new wakeup request.

In LINFlexD, you can generate a wakeup request by:

- Writing the wakeup character (F0h for LIN 2.0) to BDR0
- Writing a 1 to LINCR2[WURQ]

After you write a 1 to LINCR2[WURQ], LINFlexD transmits the character in BDR0.

In LINFlexD, wakeup can be detected in two ways.

Table 308. Wakeup direction

LINCR1[AUTOWU]	Details
0	After detecting a falling edge, LINFlexD: • Writes a 1 to LINSR[WUF] • Generates an interrupt if LINIER[WUIE]=1 Your software must then write a 0 to LINCR1[SLEEP].
1	After detecting a falling edge in Sleep mode, LINFlexD: • Writes a 0 to LINCR1[SLEEP] • Writes a 1 to LINSR[WUF] • Generates an interrupt if LINIER[WUIE]=1 LINFlexD is now in Normal mode and ready to receive frames.

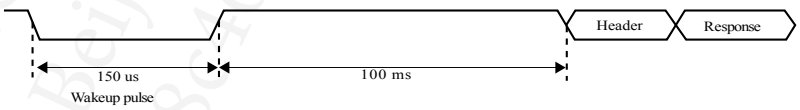


Figure 384. Wakeup sequence

48.4.3 Timer

LinflexD has an 8-bit counter which has different behavior in different modes as described below:

Table 309. Timer behavior

Mode	Details
Output Compare (LINTCSR[MODE]=1)	• Runs even in Sleep and Initialization modes • The counter value, which when matches the two software configurable output compare registers (LINOCR[OC1] or LINOCR[OC2]), generates output compare interrupt (LINESR[OCF]) if LINTCSR[TOCE]=1. • After an interrupt occurs, LINFlexD generates subsequent interrupts only when the application has reset the stuck FSM state.
LIN	• The software has no control over LINTCSR[TOCE] and LINFlexD uses output-compare registers to generate LIN timeout interrupts (header, frame, and response time out). • If LIN moves to SM or INIT state, then this counter remains in Reset state. LIN mode has no meaning if UART is enabled, hence the counter remains in Reset state.

48.4.4 UART mode

Main features in UART mode are:

- Full-duplex communication
- 8-bit, 9-bit, and
- Even/Odd/0/1 parity
- User-programmable oversampling rate to obtain a baud rate of up to LIN_CLK/4 Mbit/s

48.4.4.1 8-bit data frames

The eighth bit can be a data or a parity bit. Even/Odd/0/1 parity can be selected by the PC[1:0] bit in the same register. An even parity is set if the modulo-2 sum of the seven data bits is 1. An odd parity is cleared in this case.

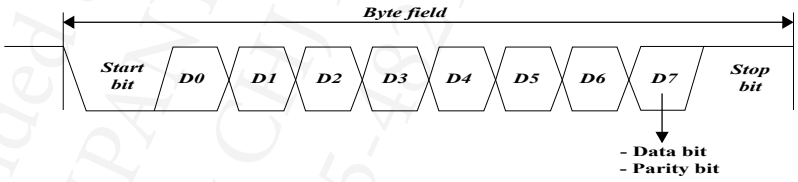


Figure 385. UART mode 8-bit data frame

48.4.4.2 9-bit frames

The ninth bit must be a parity bit. Even/Odd/0/1 parity can be selected by the PC[1:0] field in the same register. An even parity is set if the modulo-2 sum of the seven data bits is 1. An odd parity is cleared in this case. Parity 0 forces a zero logical value. Parity 1 forces a high logical value.

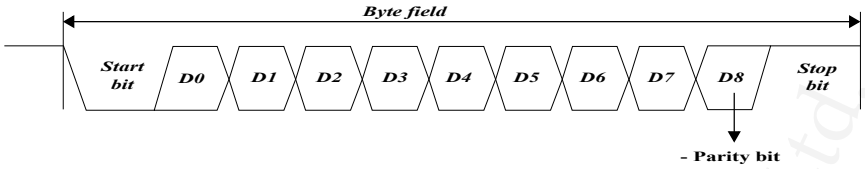


Figure 386. UART mode 9-bit data frame

48.4.4.3 Buffer in UART mode

The 8-byte buffer includes two parts—receiver and transmitter—as shown in the following figure:

Tx0	BDR0
Tx1	BDR1
Tx2	BDR2
Tx3	BDR3
Rx0	BDR4
Rx1	BDR5
Rx2	BDR6
Rx3	BDR7

Figure 387. Structure of 8-byte buffer

48.4.4.4 UART transmitter

To start transmission in UART mode, UART bit should be set and the transmitter enable bit should be set. Transmission starts when the BDR0 (least significant data byte) is programmed and continues until the number of bytes/halfwords transmitted is equal to the UARTCR[TDFL].

The transmit buffer is four bytes (when UARTCR[WL1] = 0) or two halfwords (when UARTCR[WL1] = 1). Therefore, the transmission that you trigger can have a maximum of four bytes (two halfwords). After the programmed number of bytes (halfwords) has been transmitted, the DTF flag is set in UARTSR. If UARTCR[TxEn] is reset in the middle of a transmission, then the current transmission completes and no further transmissions can start.

The buffer can be configured in FIFO mode (mandatory when the DMA Tx is enabled) by writing a 1 to UARTCR[TFBM].

NOTE

If TFF bit is set and a write is performed to the FIFO, the data transmitted may be erroneous.

Table 310. BDRL access in UART mode

IPS operation	Register	Mode (UARTCR[TFBM])	Word length (UARTCR[WL])	IPS operation result
Write Byte0	BDRL	FIFO	Byte	OK
Write Byte1-2-3	BDRL	FIFO	Byte	IPS transfer error
Write Half-word0-1	BDRL	FIFO	Byte	IPS transfer error
Write Word	BDRL	FIFO	Byte	IPS transfer error
Write Byte0-1-2-3	BDRL	FIFO	Half-word	IPS transfer error
Write Half-word0	BDRL	FIFO	Half-word	OK

Table continues on the next page...

Table 310. BDRL access in UART mode (continued)

IPS operation	Register	Mode (UARTCR[TFBM])	Word length (UARTCR[WL])	IPS operation result
Write Half-word1	BDRL	FIFO	Half-word	IPS transfer error
Write Word	BDRL	FIFO	Half-word	IPS transfer error
Read Byte0-1-2-3	BDRL	FIFO	Byte/Half-word	IPS transfer error
Read Half-word0-1	BDRL	FIFO	Byte/Half-word	IPS transfer error
Read Word	BDRL	FIFO	Byte/Half-word	IPS transfer error
Write Byte0-1-2-3	BDRL	BUFFER	Byte/Half-word	OK
Write Half-word0-1	BDRL	BUFFER	Byte/Half-word	OK
Write Word	BDRL	BUFFER	Byte/Half-word	OK
Read Byte0-1-2-3	BDRL	BUFFER	Byte/Half-word	OK
Read Half-word0-1	BDRL	BUFFER	Byte/Half-word	OK
Read Word	BDRL	BUFFER	Byte/Half-word	OK

In UART FIFO mode (UARTCR[TFBM] = 1), any read operation causes an IPS transfer error.

48.4.4.5 UART receiver

The reception of a data byte starts as soon as you:

- Exit INIT mode
- Write a 1 to UARTCR[RXEN]
- Detect the start bit

LINFlexD has a dedicated data buffer for received data. This buffer comprises 4 bytes if UARTCR[WL1]=0 or two half-words if UARTCR[WL1]=1. After the programmed number (RDFL bits) of bytes have been received, the DRF flag is set in UARTSR and the current reception finishes. You must write a 1 to UARTCR[RxEn] only to start the reception. The reception is automatically completed as soon as the programmed number (RDFL bits) of bytes have been received.

The buffer can be configured in FIFO mode (mandatory when the DMA Rx is enabled) by writing a 1 to UARTCR[RFBM].

Table 311. BDRM access in UART mode

IPS operation	Mode (UARTCR[RFBM])	Word length (UARTCR:WL)	IPS operation result
Read Byte4	FIFO	Byte	OK
Read Byte5-6-7	FIFO	Byte	IPS transfer error
Read Half-word2-3	FIFO	Byte	IPS transfer error

Table continues on the next page...

Table 311. BDRM access in UART mode (continued)

IPS operation	Mode (UARTCR[RFBM])	Word length (UARTCR:WL)	IPS operation result
Read Word	FIFO	Byte	IPS transfer error
Read Byte4-5-6-7	FIFO	Half-word	IPS transfer error
Read Half-word2	FIFO	Half-word	OK
Read Half-word3	FIFO	Half-word	IPS transfer error
Read Word	FIFO	Half-word	IPS transfer error
Write Byte4-5-6-7	FIFO	Byte/Half-word	IPS transfer error
Write Half-word2-3	FIFO	Byte/Half-word	IPS transfer error
Write Word	FIFO	Byte/Half-word	IPS transfer error
Read Byte4-5-6-7	BUFFER	Byte/Half-word	OK
Read Half-word2-3	BUFFER	Byte/Half-word	OK
Read Word	BUFFER	Byte/Half-word	OK
Write Byte4-5-6-7	BUFFER	Byte/Half-word	IPS transfer error
Write Half-word2-3	BUFFER	Byte/Half-word	IPS transfer error
Write Word	BUFFER	Byte/Half-word	IPS transfer error

NOTE

See the layout of the BDRL and BDRM registers to identify the mapping between byte x and data bits of these registers.

Consider the following for UART receiver:

- If you do not know in advance how many bytes are to be received, RDFL must not be programmed in advance. The reset value of RDFL is 0. This ensures that the reception happens byte by byte. The state machine moves to the Idle state after each byte reception.
- If RDFL is programmed for a certain value but that number of bytes are not received, then reception hangs. In that case, software needs to take care of timeout by seeing the flag. Software has to set the sleep bit to move to Idle state.
- If a STOP request arrives in the middle of a reception, it is only acknowledged after all the programmed number of data bytes have received; it is not served immediately. If the programmed number of data bytes are not received, then software has to take care of timeout. When the state machine moves to Idle state, then only a stop request is served.
- If during the reception of any byte a parity error occurs, then the corresponding PEx bit in UARTSR is set. LINFlexD does not generate an interrupt in this case. If a framing error occurs in any byte (FE bit in UARTSR is set) then an interrupt is generated if FEIE bit in LINIER is set. As there is only one register bit for framing error, this interrupt is helpful in identifying which byte has framing error.
- If the last received frame has not been read from the buffer (in other words, you did not write a 1 to LINSR[RMB]), then upon reception of the next byte, an overrun error occurs (BOF bit in UARTSR is set) and one message is depending on RBLM bit of LINCR1. An interrupt is generated if the BOIE bit in LINIER is set.

48.4.5 DMA interface

48.4.5.1 Main features

The LINFlexD DMA offers a parametric and programmable solution with the following distinctive features:

Table 312. LINFlexD DMA features

Node	Mode	Details
LIN Master	TX	Single DMA channel
LIN Master	RX	Single DMA channel
LIN Slave	TX	1 to DMA_TX_CH_NUM DMA channel See the chip-specific LINFlexD information for the value of DMA_TX_CH_NUM used in this chip.
LIN Slave	RX	1 to DMA_RX_CH_NUM DMA channel See the chip-specific LINFlexD information for the value of DMA_RX_CH_NUM used in this chip.
UART	TX	Single DMA channel
UART	RX	Single DMA channel plus timeout

48.4.5.2 Definitions

The LINFlexD control register and their fields are described in Table 313.

Table 313. LINFlexD DMA control fields description

Field	Mode	Value	Level	Description
DMA_TEN	LIN master Tx or UART Tx	DMATXE[0]	High	Logical AND between the 2 DMA enable bits (LINFlexD and eDMA).
DMA_REN	LIN master Rx or UART Rx	DMARXE[0]	High	Logical AND between the 2 DMA enable bits (LINFlexD and eDMA).

Control/status fields of the transfer control descriptors referred to in this document are described in Table 314.

Table 314. TCD control fields description

TCD Field	Level	Description
CITER[14:0]	—	Current "major" iteration count

Table continues on the next page...

Table 314. TCD control fields description (continued)

TCD Field	Level	Description
BITER[14:0]	—	Beginning "major" iteration count
NBYTES[31:0]	—	Inner "minor" byte transfer count. Number of bytes to be transferred in each service request of the channel.
SADDR[31:0]	—	Source address
SOFF[15:0]	—	Source address signed offset applied to the current source address as each source read is completed.
SSIZE[2:0]	—	Source data transfer size 000 => 8-bit 001 => 16-bit 010 => 32-bit 011 => 64-bit
SLAST[31:0]	—	Last source address adjustment
DADDR[31:0]	—	Destination address
DOFF[15:0]	—	Destination address signed offset applied to the current destination address as each destination write is completed.
DSIZE[2:0]	—	Destination data transfer size 000 => 8-bit 001 => 16-bit 010 => 32-bit 011 => 64-bit
DLAST_SGA[31:0]	—	Last destination address adjustment or the memory address for the next TCD to be loaded into this channel (scatter/gather)
INT_MAJ	High	Enable an interrupt when major iteration count completes
START	High	The channel is explicitly started via a software initiated service request.
DONE	High	Channel done (the DMA has completed the outer major loop).
D_REQ	High	Disable request. If this flag is set the DMA hardware automatically clears the corresponding DMAERQ bit when the current major iteration count reaches zero.

48.4.5.3 Master node – TX mode

On a master node in TX mode, the DMA interface requires a single TX channel. Each TCD controls a single frame. The following figure shows the memory map associated with the TCD chain (RAM area and LINFlexD registers).

Provided under NDA
COMPANY CONFIDENTIAL
Freescale Semiconductor, Inc.
c4625a21-8a8-b25-390d3ab6241

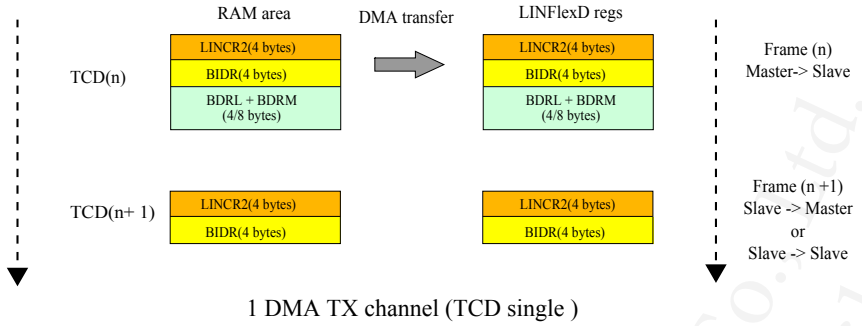


Figure 388. Master node – TX memory map

The TCD chain of the DMA Tx channel on a master node supports:

- Master to Slave: transmission of the entire frame (header + data)
- Slave to Master: transmission of the header; the Rx channel on the master node controls the data reception
- Slave to Slave: transmission of the header

The following table provides the settings for LINCR2 and BIDR for each class of LIN frame.

Table 315. Master node – Tx mode – Register setting

LIN frame	LINCR2	BIDR
Master to Slave	DDRQ = 1 DTRQ = 0 HTRQ = 0	DFL = payload size ID = address CCS = checksum DIR = 1 (TX)
Slave to Master	DDRQ = 0 DTRQ = 0 HTRQ = 0	DFL = payload size ID = address CCS = checksum DIR = 0 (RX)
Slave to Slave	DDRQ = 1 DTRQ = 0 HTRQ = 0	DFL = payload size ID = address CCS = checksum DIR = 0 (RX)

The DMA Tx interface is shown in Figure 389. DMA TX moves to Idle state immediately at next clock edge if DMATXE[0] = 0. The TCD setting (word transfer) is given in Table 316. All other TCD fields = 0. TCD settings based on halfword or byte transfer are allowed.

Table 316. TCD setting – Master node – Tx mode

TCD field	Value	Description
CITER[14:0]	1	Single iteration for the "major" loop

Table continues on the next page...

Table 316. TCD setting – Master node – Tx mode (continued)

TCD field	Value	Description
BITER[14:0]	1	Single iteration for the "major" loop
NBYTES[31:0]	[4 + 4] + 0/4/8 = N	Data buffer is stuffed with dummy bytes if the length is not word aligned. LINCR2 + BIDR + BDRL + BDRM
SADDR[31:0]	-	RAM address
SOFF[15:0]	4	Word increment
SSIZE[2:0]	2	Word transfer
SLAST[31:0]	–N	
DADDR[31:0]	-	LINCR2 address
DOFF[15:0]	4	Word increment
DSIZE[2:0]	2	Word transfer
DLAST_SGA[31:0]	–N	No scatter/gather processing
INT_MAJ	0/1	Interrupt disabled/enabled
D_REQ	1	Only on the last TCD of the chain
START	0	No SW request

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
c8c462e5-4821-48a8-825-390d3ab624c1

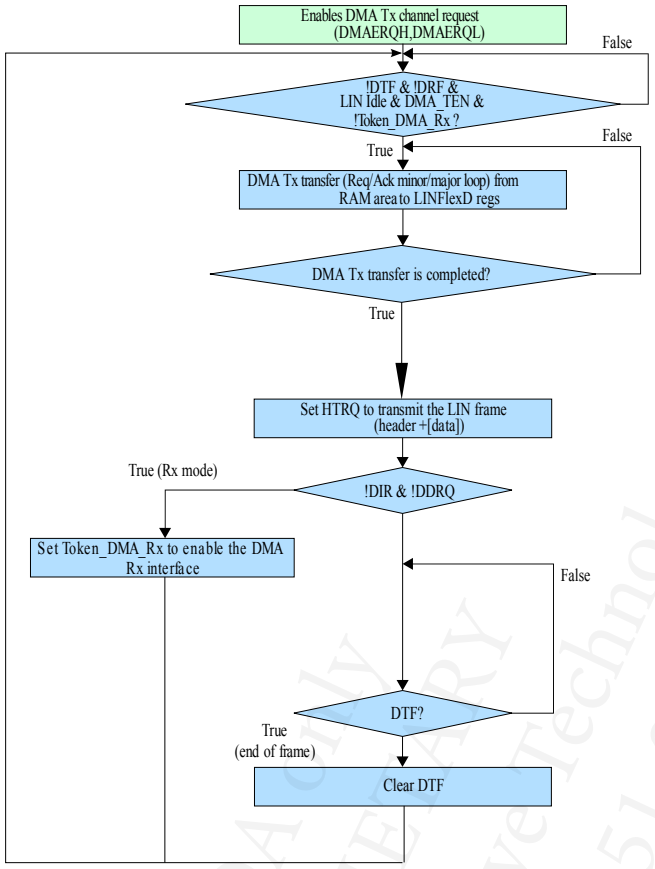


Figure 389. Master node – DMA Tx (concept scheme)

48.4.5.4 Master node - RX mode

On a master node in RX mode, the DMA interface requires a single RX channel. Each TCD controls a single frame. The following figure shows the memory map associated with the TCD chain (RAM area and LINFlexD registers).

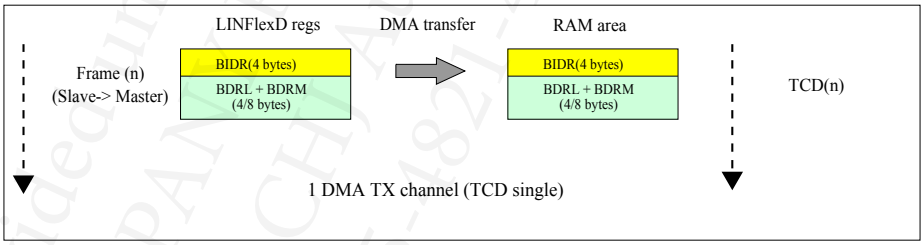


Figure 390. Master node – RX memory map

The TCD chain of the DMA Rx channel on a master node supports Slave to Master: reception of the data field of the header.

The BIDR is optionally copied into the RAM area, and this register (part of FIFO data) contains the ID of each message, which only allows the CPU to figure out which ID the LINFlexD DMA received if the single DMA channel setup is used.

The DMA Rx interface is shown in Figure 391. DMA RX moves to Idle state immediately at the next clock edge if DMARXE[0] are 0. The TCD setting (word transfer) is given in the next table. All other TCD fields are 0. TCD settings based on halfword or byte transfer are allowed.

Table 317. TCD setting – Master node – Rx mode

TCD field	Value	Description
CITER[14:0]	1	Single iteration for the major loop
BITER[14:0]	1	Single iteration for the major loop
NBYTES[31:0]	4 +4/8 = N	Data buffer is stuffed with dummy bytes if the length is not word aligned. BIDR + BDRL + BDRM
SADDR[31:0]	—	BIDR address
SOFF[15:0]	4	Word increment
SSIZE[2:0]	2	Word transfer
SLAST[31:0]	–N	—
DADDR[31:0]	—	RAM address
DOFF[15:0]	4	Word increment
DSIZE[2:0]	2	Word transfer
DLAST_SGA[31:0]	–N	No scatter/gather processing
INT_MAJ	0/1	Interrupt disabled/enabled
D_REQ	1	Only on the last TCD of the chain
START	0	No SW request

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-8e25-390d3ab624c1

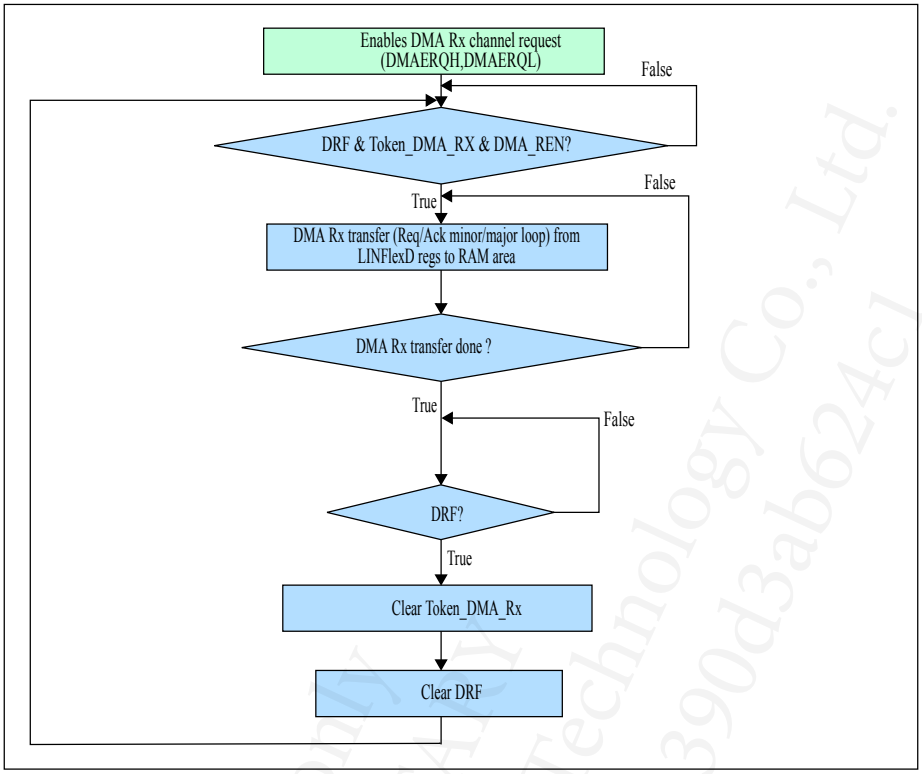


Figure 391. Master node – DMA Rx (concept scheme)

48.4.5.5 UART – TX mode

In UART TX mode, the DMA interface requires a DMA TX channel. A single TCD can control the transmission of an entire Tx buffer. The following figure shows the memory map associated with the TCD chain (RAM area and LINFlexD registers).

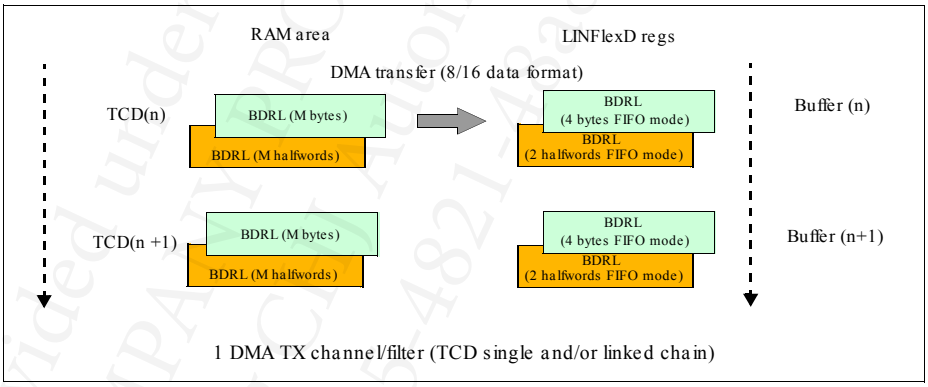


Figure 392. UART – TX memory map

The UART TX buffer must be configured in FIFO mode to:

- Allow the transfer of large data buffer by a single TCD
- Absorb the latency, following a DMA request (because of the DMA arbitration), to move data from the RAM to the FIFO
- Use low-priority DMA channels

The Tx FIFO size is:

- 4 bytes in 8-bit data format

FIFO-not-full (TX) status signals trigger a DMA request.

The concept FSM to control the DMA Tx interface is given in Figure 393. DMA Tx FSM will move to Idle state if DMATXE[0] = 0.The TCD setting (typical case) is given in Table 318. All other TCD fields = 0. The minor loop transfers a single byte/halfword as soon as a free entry is available in the Tx FIFO.

Table 318. TCD setting – UART – Tx mode

TCD field	Value		Description
	8 bits data	16 bits data	
CITER[14:0]	M		Multiple iterations for the "major" loop
BITER[14:0]	M		Multiple iterations for the "major" loop
NBYTES[31:0]	1	2	Minor loop transfer = 1 or 2 bytes
SADDR[31:0]	-		RAM address
SOFF[15:0]	1	2	Byte/Half-word increment
SSIZE[2:0]	0	1	Byte/Half-word transfer
SLAST[31:0]	–M	–M × 2	-
DADDR[31:0]	-		BDRL address
DOFF[15:0]	0		No increment (FIFO)
DSIZE[2:0]	0	1	Byte/Half-word transfer
DLAST_SGA[31:0]	0		No scatter/gather processing
INT_MAJ	0/1		Interrupt disabled/enabled
D_REQ	1		Only on the last TCD of the chain.
START	0		No SW request

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4ab6-8255-3903a8e02ac1

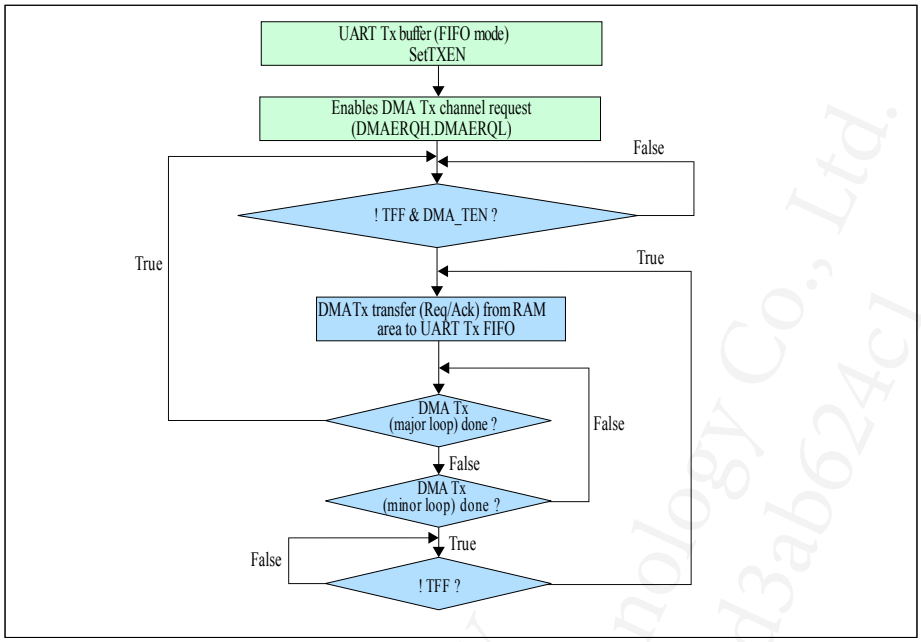


Figure 393. UART – DMA Tx FSM (concept scheme)

48.4.5.6 UART – RX mode

In UART RX mode, the DMA interface requires a DMA RX channel. A single TCD can control the reception of an entire Rx buffer. The following figure shows the memory map associated with the TCD chain (RAM area and LINFlexD registers).

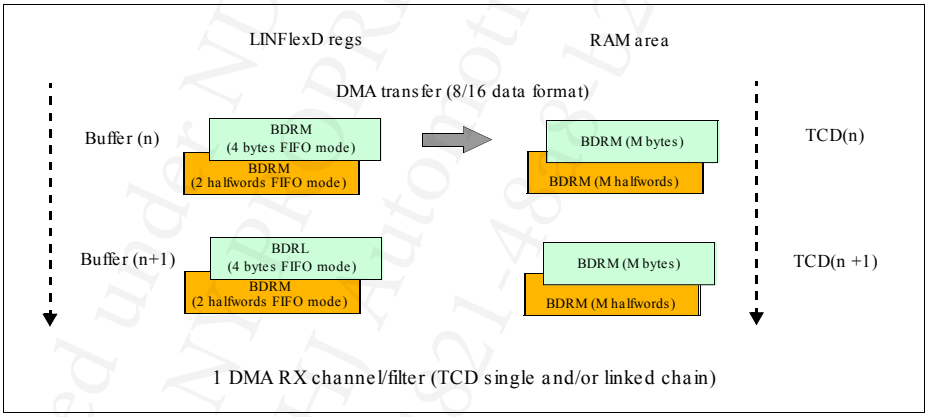


Figure 394. UART – RX memory map

The UART RX buffer must be configured in FIFO mode to:

- Allow the transfer of large data buffer by a single TCD
- Absorb the latency, following a DMA request (due to the DMA arbitration), to move data from the FIFO to the RAM
- Use low priority DMA channels

The Rx FIFO comprises 4 bytes in an 8-bit data format.

This is sufficient because just one byte allows a reaction time of about 3.8 µs (at 2 Mbit/s) (~450 clock cycles at 120 MHz) before the transmission is affected. FIFO triggers a DMA request. Empty (Rx) status signals do not trigger a DMA request.

The DMA Rx interface is shown in Figure 395. DMA Rx moves to Idle state if DMARXE[0] = 0.The TCD setting (typical case) is shown in the next table. All other TCD fields equal 0. The minor loop transfers a single byte/halfword as soon an entry is available in the Rx FIFO. A new software reset bit is required that allows the LINFlexD FSMs to be reset in case this timeout state is reached or in any other case. You can rewrite the timeout counter at any time to extend the timeout period.

Table 319. TCD setting – UART – Rx mode

TCD field	Value		Description
	8 bits data	16 bits data	
CITER[14:0]	M		Multiple iterations for the "major" loop
BITER[14:0]	M		Multiple iterations for the "major" loop
NBYTES[31:0]	1	2	Minor loop transfer = 1 or 2 bytes
SADDR[31:0]	-		BDRM address
SOFF[15:0]	0		No increment (FIFO)
SSIZE[2:0]	0	1	Byte/Half-word transfer
SLAST[31:0]	0		
DADDR[31:0]	-		RAM address
DOFF[15:0]	1	2	Byte/Half-word increment
DSIZE[2:0]	0	1	Byte/Half-word transfer
DLAST_SGA[31:0]	–M	–M × 2	No scatter/gather processing
INT_MAJ	0/1		Interrupt disabled/enabled
D_REQ	1		Only on the last TCD of the chain.
START	0		No SW request

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a1c-b25-390a5ab02d1

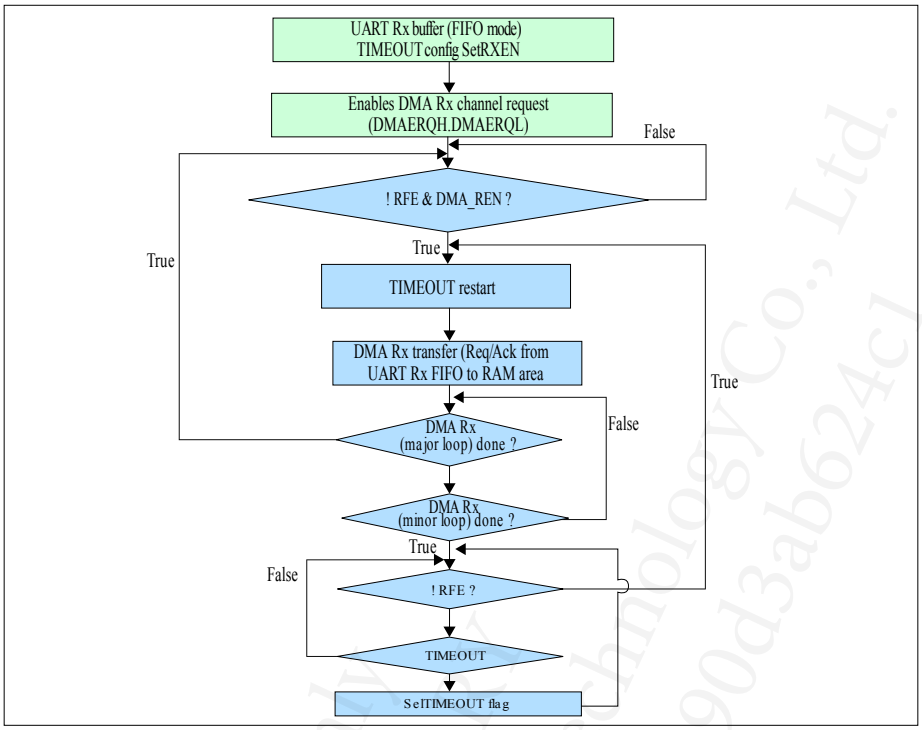


Figure 395. UART – DMA Rx (concept scheme)

48.4.5.7 Use cases and limitations

In LIN master mode, both the DMA channels (TX and RX) must be enabled in case DMA is required.

In UART mode the DMA capability can be used only if the UART Tx/Rx buffers are configured as FIFOs.

DMA and CPU operating modes are mutually exclusive for the data/frame transfer on a UART or LIN node. After a DMA transfer finishes, the CPU can manage subsequent accesses.

Error management must always be executed via the CPU enabling the related error interrupt sources. DMA does not support for error management. Error management means checking status bits, handling IRQs, and potentially canceling DMA transfers.

The DMA programming model must be coherent with the TCD setting defined in this document.

48.5 LINFlexD register descriptions

The following points should be considered for LINFlexD registers:

- Reset values are with-slave/without-slave (master only) format
- If not specified, then the bit can be configured for both master only and master/slave format; in other words, for both values of (generic) slave = 0 and (generic) slave = 1.

Table 320. Register offsets based on master slave configuration and number of filters available

Register	Master only mode	Master Slave mode			
		Number of filters = 0	Number of filters = 4	Number of filters = 8	Number of filters = 16
LIN Control register 1 (LINCR1) -	0x00-0x3C	0x00-0x3C	0x00-0x3C	0x00-0x3C	0x00-0x3C

Table continues on the next page...

Table 320. Register offsets based on master slave configuration and number of filters available (continued)

Register	Master only mode	Master Slave mode			
Buffer Data Register Most Significant(BDRM)					
Identifier Filter Enable Register(IFER) - Identifier Filter Mode register (IFMR)	Reserved/ 0x40-0x48 Transfer error generated	Reserved/ 0x40-0x48 Transfer error generated	0x40-0x48	0x40-0x48	0x40-0x48
Identifier Filter Control registers(IFCR0-3)	Not available	Not available	0x4C-0x58	0x4C-0x58	0x4C-0x58
Identifier Filter Control registers(IFCR4-7)	Not available	Not available	Not available	0x5C-0x68	0x5C-0x68
Identifier Filter Control registers (IFCR8-15)	Not available	Not available	Not available	Not available	0x6C-0x88
Global Control register (GCR) - DMA Rx Enable register (DMARXE)	0x4C-0x5C	0x4C-0x5C	0x5C-0x6C	0x6C-0x7C	0x8C-0x9C
Transfer errors for outside memory map register locations					
	Transfer error generated after 0x60	Transfer error generated after 0x60	Transfer error generated after 0x70	Transfer error generated after 0x80	Transfer error generated after 0xA0

48.5.1 LINFlexD memory map

LINFlexD_0 base address: 401C_8000h

LINFlexD_1 base address: 401C_C000h

LINFlexD_2 base address: 402B_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	LIN Control Register 1 (LINCR1)	32	RW	0000_0082h
4h	LIN Interrupt Enable Register (LINIER)	32	RW	0000_0000h
8h	LIN Status Register (LINSR)	32	W1C	0000_0040h
Ch	LIN Error Status Register (LINESR)	32	W1C	0000_0000h
10h	UART Mode Control Register (UARTCR)	32	RW	0000_0000h
14h	UART Mode Status Register (UARTSR)	32	W1C	0000_0000h
18h	LIN Time-Out Control Status Register (LINTCSR)	32	RW	0000_0200h
1Ch	LIN Output Compare Register (LINOCR)	32	RW	0000_FFFFh
20h	LIN Time-Out Control Register (LINTOCR)	32	RW	0000_0E2Ch
24h	LIN Fractional Baud Rate Register (LINFBRR)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
28h	LIN Integer Baud Rate Register (LINIBRR)	32	RW	0000_0000h
2Ch	LIN Checksum Field Register (LINCFR)	32	RW	0000_0000h
30h	LIN Control Register 2 (LINCR2)	32	RW	0000_6000h
34h	Buffer Identifier Register (BIDR)	32	RW	0000_0000h
38h	Buffer Data Register Least Significant (BDRL)	32	RW	0000_0000h
3Ch	Buffer Data Register Most Significant (BDRM)	32	RW	0000_0000h
4Ch	Global Control Register (GCR)	32	RW	0000_0000h
50h	UART Preset Timeout Register (UARTPTO)	32	RW	0000_0FFFh
54h	UART Current Timeout Register (UARTCTO)	32	RO	0000_0000h
58h	DMA Tx Enable Register (DMATXE)	32	RW	0000_0000h
5Ch	DMA Rx Enable Register (DMARXE)	32	RW	0000_0000h

48.5.2 LIN Control Register 1 (LINCR1)

Offset

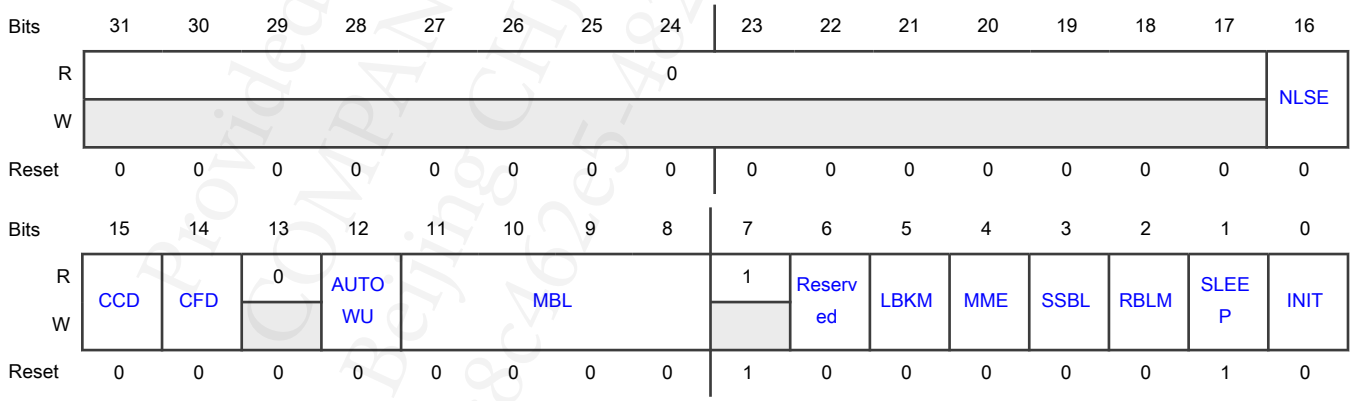
Register	Offset
LINCR1	0h

Function

Provides control bits used to configure features of the LINFlexD.

When accessing the LINCR1 register, each reserved bit must be written to its original reset value.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
ca4626e5-4821-48a8-b251-39d6ab6a1a1a

Fields

Field	Function
31-17 —	Reserved
16 NLSE	LIN State Capture Enable on Bit Error Enables capture of LIN state, LINSR[LINS] whenever bit error flag occur, that is, LINESR[BEF] set to 1. If this bit is set to 1, internal LIN state is not captured when LINCR2[IOBE]=1 and the state transition to IDLE occurs; and LINSR[LINS] shows error state only. 0b - LIN state, LINSR[LINS] shows the current LIN state and does not capture the LIN state if a bit error occurs. 1b - LIN state, LINSR[LINS] is captured whenever LINESR[BEF] occurs.
15 CCD	Checksum Calculation Disable You can read this field at any time and write to it only in Initialization mode. 0b - Hardware performs the checksum calculation. When this bit is reset, the LINCFR becomes read only. 1b - Checksum calculation disabled. When this field is set, the LINCFR is read/write. You can program this register to send a software calculated checksum/CRC (provided CFD is reset).
14 CFD	Checksum Field Disable You can read this field at any time and write to it only in Initialization mode. 0b - Checksum field is sent after the required number of data bytes are sent 1b - No checksum field is sent in the frame
13 —	Reserved
12 AUTOWU	Auto Wakeup You can read this field at any time and write to it only in Initialization mode. This bit is utilized in UART mode also. 0b - Sleep bit is cleared by software only 1b - Sleep bit gets cleared by hardware whenever WUF bit of LINSR is set
11-8 MBL	Master Break Length Chooses the length of the Sync break that the master generates. You can read this field at any time and write to it only in Initialization mode. 0000b - 10-bit break length 0001b - 11-bit break length 0010b - 12-bit break length 0011b - 13-bit break length

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0100b - 14-bit break length 0101b - 15-bit break length 0110b - 16-bit break length 0111b - 17-bit break length 1000b - 18-bit break length 1001b - 19-bit break length 1010b - 20-bit break length 1011b - 21-bit break length 1100b - 22-bit break length 1101b - 23-bit break length 1110b - 36-bit break length 1111b - 50-bit break length
7 —	Reserved
6 —	Reserved
5 LBKM	Loop Back mode See "Loop back mode" in Test mode. You can read this field at any time and write to it only in Initialization mode. This bit is utilized in UART mode also. 0b - Loop Back mode disabled 1b - Loop Back mode enabled
4 MME	Master Mode Enable You can read this field at any time and write to it only in Initialization mode. 0b - Slave mode 1b - Master mode
3 SSBL	Slave Mode Sync Break Length You can read this field at any time and write to it only in Initialization mode. 0b - 11-bit break length 1b - 10-bit break length
2 RBLM	Receiver Buffer Locked mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
	You can read this field at any time and write to it only in Initialization mode. This bit is utilized in UART mode also. 0b - Receiver buffer not locked. Next incoming message overwrites the old one. 1b - Receiver buffer locked against overrun. After the buffer is full, the next incoming message will be discarded if buffer is not released (UARTSR[RMB]).
1 SLEEP	Sleep Mode Request Write a 1 to this field to request LINFlexD to enter Sleep mode. This bit is cleared by software or hardware (if AUTOWU bit in LINCR1 and WUF bit in LINSR are set) to exit sleep mode. This bit is utilized in UART mode also.
0 INIT	Initialization Mode Request Write a 1 to this field to request LINFlexD to enter Initialization mode. On clearing this bit (and if LINCR1[SLEEP]=0), LINFlexD enters Normal mode. This bit is utilized in UART mode also.

48.5.3 LIN Interrupt Enable Register (LINIER)

Offset

Register	Offset
LINIER	4h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SZIE	OCIE	BEIE	CEIE	HEIE	0		FEIE	BOIE	LSIE	WUIE	Reserved	TOIE	DRIE	DTIE	HRIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 SZIE	Stuck at Zero Interrupt Enable An interrupt is generated if this bit is set and the Stuck at Zero Flag (SZF) in LINESR or UARTSR is set. 0b - No interrupt 1b - Interrupt enabled
14 OCIE	Output Compare Interrupt Enable 0b - No interrupt 1b - Interrupt generated when OCF bit in LINESR or UARTSR is set
13 BEIE	Bit Error Interrupt Enable 0b - No interrupt 1b - Interrupt generated when BEF bit in LINESR is set
12 CEIE	Checksum Error Interrupt Enable An interrupt is generated if this bit is set and the Checksum Error Flag (CEF) is set in LINESR. 0b - No interrupt 1b - Interrupt enabled
11 HEIE	Header Error Interrupt Enable An interrupt is generated when this bit is set and either of the following flags are set: SFEF, SDEF, or IDPEF. **NOTE** If generic slave = 0, then this bit will always read a 0 and cannot be programmed. 0b - No interrupt 1b - Interrupt enabled
10-9 —	Reserved
8 FEIE	Frame Error Interrupt Enable 0b - No interrupt 1b - Interrupt generated if Frame Error Flag (FEF) bit is set in LINESR or UARTSR
7 BOIE	Buffer Overrun Error Interrupt Enable An interrupt is generated if this bit is set and the Buffer Overrun Flag (BOF) is set in LINESR or UARTSR. 0b - No interrupt 1b - Interrupt enabled
6 LSIE	LIN State Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Interrupt is generated only when entering the above fields. You can use this interrupt for debugging purposes. The interrupt has no status flag. 0b - No interrupt 1b - Interrupt generated when entering the following states: Sync Del, Sync Field, Identifier field, Checksum
5 WUIE	Wakeup Interrupt Enable If WUIE=1 and the WUF in LINSR or UARTSR is set then an interrupt is generated. 0b - No interrupt 1b - Interrupt enabled
4 —	Reserved
3 TOIE	Timeout Interrupt Enable An interrupt is generated if this bit is set and UARTSR[TO]=1 (in UART mode). 0b - No interrupt 1b - Interrupt enabled
2 DRIE	Data Reception Complete Interrupt Enable An interrupt is generated when this bit is set and Data Received flag (DRF) in LINSR or UARTSR is set. 0b - No interrupt 1b - Interrupt enabled
1 DTIE	Data Transmitted Interrupt enable An interrupt is generated when this bit is set and Data Transmitted flag (DTF) in LINSR or UARTSR is set. 0b - No interrupt 1b - Interrupt enabled
0 HRIE	Header Received Interrupt An interrupt is generated when this bit is set and the Header Received flag (HRF) in LINSR is set. **NOTE** If generic slave = 0, then this bit will always read a 0, and cannot be programmed. 0b - No interrupt 1b - Interrupt enabled

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
c8c462e5-4821-48a8-b251-320d3ab624c1

48.5.4 LIN Status Register (LINSR)

Offset

Register	Offset
LINSR	8h

Function

Indicates the LINFlexD state

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												0	RDC		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LINS				0		RMB	DRBNE	RXBUSY	RDI	WUF	0		DRF	DTF	HRF
W	1						W1C	W1C			W1C			W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 —	Reserved
18-16 RDC	Receive Data Byte Count Contains the number of entries (bytes) in the Receive data buffer in LIN mode. You can use this field for debug purposes. 000b - 1 byte 001b - 2 bytes 010b - 3 bytes 011b - 4 bytes 100b - 5 bytes 101b - 6 bytes

Table continues on the next page...

Table continued from the previous page...

Field	Function
	110b - 7 bytes 111b - 8 bytes
15-12 LINS	LIN State In UART mode, Idle, Init, Sleep, and Data Transmission/Reception states are flagged by the LIN status bits. **NOTE** The value of this field does not change by any write operation to it. Writing Fh to this field clears the Rx interrupt, only when set because of the LIN state event. 0000b - Sleep mode: LINFlexD is in Sleep mode, to save power consumption. 0001b - Init mode: LINFlexD is in Initialization mode. 0010b - Idle mode: This mode is entered when: SLEEP bit and INIT bit are reset by software; Wakeup pulse has been received on RX pin (AUTOWU set); Previous frame transmission/ reception has been completed/aborted 0011b - Sync break: In Slave mode, if a falling edge is detected followed by a dominant state, it indicates that the sync break reception is ongoing. In Master mode, this indicates that the sync break transmission is ongoing. In slave mode, upon any error LIN state could be either Idle or Rec_Break, depending on last bit detected on LIN_RX. If the last bit detected is dominant then Rec_Break, otherwise Idle. 0100b - Sync Del: In Slave mode, valid sync break has been detected (10 bit or 11 bit). Waiting for a rising edge. In Master mode, Sync break transmission has been completed, sync delimiter transmission is ongoing. 0101b - Sync Field: In Slave mode, a valid sync Del has been detected (recessive state for at least one bit time). Receiving sync field. In Master mode, sync field transmission ongoing. 0110b - Identifier Field: In Slave mode, a valid sync field has been received. Receiving ID field. In Master mode, identifier transmission is ongoing. 0111b - Header Reception/Transmission: In Slave mode, a valid header has been received and Identifier field is available in the BIDR. In Master mode, header transmitted. 1000b - Data Reception/Data Transmission: In Receiver mode, reception ongoing. In Transmitter mode, response transmission ongoing. 1001b - Checksum: Data transmission/reception completed, checksum transmission/reception ongoing.
11-10 —	Reserved
9 RMB	Release Message Buffer 0b - Buffer data is free and is reset by hardware when it in Initialization mode. 1b - Buffer data ready to be read by software. This bit must be cleared by software after reading the data received in the buffer.
8	Data Reception Buffer Not Empty

Table continues on the next page...

Table continued from the previous page...

Field	Function
DRBNE	LINFlexD writes a 1 to this field as soon as the first byte of response has been received and stored in BDRL (when there is at least one data byte in reception buffer). Software must clear it after reading all the buffers. LINFlexD writes a 0 to this field in Initialization mode. This flag could be checked by software in case of a response timeout event.
7 RXBUSY	Receiver Busy In Slave mode after header reception, if DIR bit is reset and reception starts, then this bit is set. In this case, you cannot write a 1 to LINCR2[DTRQ]. 0b - Receiver idle 1b - Reception ongoing
6 RDI	Receiver Data Input Reflects the current status of the Rx pin **NOTE** After reset is released, RDI reflects the actual value of Rx pin.
5 WUF	Wakeup flag This bit is set by hardware when a falling edge is detected on the Rx pin. • When slave is in Sleep mode • When master is in Sleep mode or Idle mode LINFlexD writes a 0 to this field in Initialization mode.
4-3 —	Reserved
2 DRF	Data Reception Completed Flag This bit is set by hardware and indicates that data reception completed. This flag must be cleared by software. LINFlexD writes a 0 to this field in Initialization mode. **NOTE** In case framing error or checksum error occurs then this flag is not set.
1 DTF	Data Transmission Completed Flag This bit is set by hardware and indicates that data transmission completed. This flag must be cleared by software. LINFlexD writes a 0 to this field in Initialization mode. **NOTE** For LIN mode, in case a bit error occurs (and IOBE is reset) then this flag is not set.
0 HRF	Header Received flag This bit is set when the header reception is completed. It must be cleared by software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit is set only when: • All filters are inactive and Bypass filter (BF) bit is set • No match in any filter and Bypass filter (BF) bit is set • TX filter match At end of frame or frame aborted, if HRF is still set, it gets reset by hardware. This bit is also reset by hardware in Initialization mode. **NOTE** If generic slave = 0, then this bit will always read a 0, and cannot be programmed.

48.5.5 LIN Error Status Register (LINESR)

Offset

Register	Offset
LINESR	Ch

Function

See Errors for a detailed description of all errors.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SZF	OCF	BEF	CEF	SFEF	SDEF	IDPEF	FEF	BOF	0						NF
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C							W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 SZF	Stuck At Zero Flag This bit is set when there is a stuck-at-zero timeout error. It must be reset by software.
14 OCF	Output Compare Flag This bit must be cleared by software. If this bit is set, MODE bit in LINTCSR is cleared, and the IOT bit of LINTCSR is set, then LINFlexD moves to Idle state. If LINTCSR[MODE]=0, then LINflexD writes a 0 to OCF in Initialization mode. If LINTCSR[MODE]=1, then OCF maintains its value regardless of the LIN state. 0b - No output compare event occurred 1b - In master mode, LINESR[OCF] flag is set when counter LINTCSR[CNT] has matched the content of LINOCR[OC2]. In slave mode, LINESR[OCF] is set when the content of the counter LINTCSR[CNT] has matched the content of LINOCR[OC1] or LINOCR[OC2].
13 BEF	Bit Error Flag LINFlexD writes a 1 to this field when a bit error occurs. It should be cleared by software. LINFlexD writes a 0 to this field in Initialization mode.
12 CEF	Checksum Error Flag LINFlexD writes a 1 to this field if the received checksum does not match the hardware-calculated checksum. It must be cleared by software. This error will never occur if CCD or CFD bit of LINCR1 is set. LINFlexD writes a 0 to this field in Initialization mode.
11 SFEF	Sync Field Error Flag LINFlexD writes a 1 to this field when the received Sync Field is inconsistent. It must be cleared by software. LINFlexD writes a 0 to this field in Initialization mode. If generic slave = 0, then this bit will always read a 0, and cannot be programmed.
10 SDEF	Sync Delimiter Error Flag TLINFlexD writes a 1 to this field when the delimiter is too short (in other words, less than one bit time). It should be cleared by software. LINFlexD writes a 0 to this field in Initialization mode. **NOTE** If generic slave = 0, then this bit will always read a 0, and cannot be programmed.
9 IDPEF	ID Parity Error Flag TLINFlexD writes a 1 to this field when an error in the ID parity occurs. It should be cleared by software. **NOTE** Header Error Interrupt is triggered for SFEF or SDEF or IDPEF flag is set and HEIE is set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If generic slave = 0, then this bit will always read a 0, and cannot be programmed. For generic slave = 1,this bit is reset by hardware in Initialization mode.
8 FEF	Framing Error Flag LINFlexD writes a 1 to this field when a framing error (invalid stop bit) occurs. It should be cleared by software. LINFlexD writes a 0 to this field in Initialization mode.
7 BOF	Buffer Overrun Flag This bit is set by hardware when there is a new byte received and RMB bit is not cleared. It can be cleared by software. LINFlexD writes a 0 to this field in Initialization mode.
6-1 —	Reserved
0 NF	Noise Flag This bit is set by hardware when noise is detected in the received character. It should be cleared by software. LINFlexD writes a 0 to this field in Initialization mode.

48.5.6 UART Mode Control Register (UARTCR)

Offset

Register	Offset
UARTCR	10h

Function

NOTE

When accessing the UARTCR register, reserved bits must always be written to zero.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MIS	CSP			OSR				ROSE	NEF			DTU_PCE...	SBUR		Reserved
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TDFL_TFC			RDFL_RFC			RFBM	TFBM	WL1	PC1	RxEn	TxEn	PC0	PCE	WL0	UART
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MIS	Monitor Idle State Controls what UARTCTO monitors. Register bit can be read in any mode, written only in initialization mode when UART bit is set. 0b - UARTCTO monitors the number of bits to be received. 1b - UARTCTO monitors the idle state of the reception line.
30-28 CSP	Configurable Sample Point Decides the sample point during reduced oversampling. CSP can take the following range of values for a given oversampling rate: OSR / CSP: 4 — 2,3 5 — 2, 3, 4 6 — 3, 4, 5 8 — NA Register bit can be read in any mode, written only in initialization mode when UART bit is set.
27-24 OSR	Over Sampling Rate Configures the number of samples taken for a bit when reduced oversampling is enabled. In most cases, OSR can be 4, 5, 6 or 8. When idle state monitoring is enabled (MIS=1), OSR can be only 4 or 8. 4 or 8. Register bit can be read in any mode, written only in initialization mode when UART bit is set.
23	Reduced Over Sampling Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
ROSE	Register bit can be read in any mode, written only in initialization mode when UART bit is set. 0b - Each bit is over sampled sixteen times. 1b - OSR determines the oversampling rate.
22-20 NEF	Number of expected frames Configures the number of expected frames in UART reception mode. If the DTU bit is set, then the UART timeout counter will be reset after the configured number of frames have been received. Register bit can be read in any mode, written only when UART bit is set. Register bit is a read/write field.
19 DTU_PCETX	Disable Timeout in UART mode This bit can be programmed in Initialization mode only when the UART bit is set. There is an internal counter that gives the number of frames received. **In Buffer mode:** This timer gets reset whenever the number of bytes received (rec_byte_cnt) is equal to RDFL. At this point, the DRF bit is set. Also, the num_of_frames (number of frames received) and rec_byte_cnt are reset. When num_of_frames equals NEF, Disable Timeout is generated. But this clears num_of_frames also. Thus the timer restarts. The value of counter is 0 when it restarts. **In FIFO mode:** When num_of_frames equals NEF, Disable Timeout is generated which clears num_of_frames and causes the timer to restart. The value of counter is 0 when it restarts. Notes: • Disable Timeout causes only the Timer to restart which enables Timeout again. • Timer reset means it resets and starts counting again and the Timeout is enabled. 0b - Timeout has to be handled by software 1b - Timeout in UART mode is disabled after the configured number of data frames are received
18-17 SBUR	Stop Bits In UART Reception Mode When the UART is used for transmission and reception, you need to set the same number of stop bits in GCR and SBUR. When the UART is used only as receiver, it is enough to set SBUR bits only. Register bit can be read in any mode, written only in Initialization mode when UART bit is set. 00b - 1 stop bit 01b - 2 stop bits 10b - 3 stop bits 11b - reserved
16 —	Reserved This reserved field is writable but do not write any value to it other than its reset value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-13 TDFL_TFC	Transmitter Data Field Length/TX FIFO Counter TDFL defines the number of bytes to be transmitted in UART buffer mode (TFBM = 0). Bit 15 is reserved and not implemented. When the UART data length is configured as halfword (WL = 10 or WL = 11), only the configurations TDFL = x01 or TDFL = x11 are allowed. x00 : 1 byte x01 : 2 bytes x10 : 3 bytes x11 : 4 bytes Register bit is a read/write field. TFC contains the number of entries (bytes) of the Tx FIFO in UART FIFO mode (TFBM = 1). TFCx is available for debug purposes. 000 : empty 001 : 1 byte 010 : 2 bytes 011 : 3 bytes 100 : 4 bytes Others : reserved TDFL_TFC can be programmed and is significant only when the UART bit is set. When you have connected a debugger, TDFL_TFC decrements if you read TxFIFO through the debugger. If the data path functions normally, LINFlexD clears the RFC/TFC counters internally.
12-10 RDFL_RFC	Reception Data Field Length/RX FIFO Counter RDFL defines the number of bytes to be received in UART buffer mode (RFBM = 0). Bit 19 is reserved and not implemented. When the UART data length is configured as halfword (WL = 10 or WL = 11), only the configurations RDFL = x01 or RDFL = x11 are allowed. x00 : 1 byte x01 : 2 bytes x10 : 3 bytes x11 : 4 bytes RFC contains the number of entries (bytes) of the Rx FIFO in UART FIFO mode (RFBM = 1). RFCx is available for debug purposes. 000 : Empty 001 : 1 byte 010 : 2 bytes 011 : 3 bytes 100 : 4 bytes

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Others - Reserved RDFL_RFC can be programmed and is significant only when the UART bit is set. **NOTE** In buffer mode, RDFL_RFC must be programmed to be greater than or equal to NEF (number of expected frames). In FIFO mode, there is no such constraint. When you have connected a debugger, RDFL_RFC decrements if you read RxFIFO through the debugger. If the data path functions normally, LINFlexD clears the RFC/TFC counters internally.
9 RFBM	Rx FIFO/Buffer Mode Register bit can be read in any mode, written only in initialization mode when UART bit is set. 0b - Rx Buffer mode enabled 1b - Rx FIFO mode enabled (mandatory in DMA Rx mode)
8 TFBM	Tx FIFO/Buffer Mode Register bit can be read in any mode, written only in initialization mode when UART bit is set. 0b - Tx Buffer mode enabled 1b - Tx FIFO mode enabled (mandatory in DMA Tx mode)
7 WL1	Word Length In UART Mode Works with WL0 to configure word length as shown in the following table. Register bit can be read in any mode, written only in initialization mode when UART bit is set.

WL1	WL0	Description
0	0	7 bits data + parity
0	1	8 bits data when PCE = 0 or 8 bits data + parity when PCE = 1
1	0	15 bits data + parity
1	1	16 bits data when PCE = 0 or 16 bits data + parity when PCE = 1

6 PC1	Parity Control Works with PC0 to configure parity as shown in the following table. Register bit can be read in any mode, written only in initialization mode when UART bit is set.

PC1	PC0	Description
0	0	Parity sent is Even
0	1	Parity sent is Odd

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>PC1</th><th>PC0</th><th>Description</th></tr>
<tr><td>1</td><td>0</td><td>A logical 0 is always transmitted/ checked as parity bit</td></tr>
<tr><td>1</td><td>1</td><td>A logical 1 is always transmitted/ checked as parity bit</td></tr>
</table>
</td></tr>
<tr><td>5
RxEn</td><td>Receiver Enable
This bit can be programmed only when the UART bit is set.
0b - Receiver disabled
1b - Receiver enabled</td></tr>
<tr><td>4
TxEn</td><td>Transmitter Enable
This bit can be programmed only when UART bit is set.
0b - Transmitter disabled
1b - Transmitter enabled, transmission starts only when this bit is set and BDRL[DATA0] is programmed</td></tr>
<tr><td>3
PC0</td><td>Parity Control
Works with PC1 to configure parity. Register bit can be read in any mode, written only in initialization mode when UART bit is set.</td></tr>
<tr><td>2
PCE</td><td>Parity Control Enable
Register bit can be read in any mode, written only in initialization mode when UART bit is set.
0b - Parity transmit/check disabled
1b - Parity transmit/check enabled</td></tr>
<tr><td>1
WL0</td><td>Word Length in UART mode
Works with WL1 to configure word length. Register bit can be read in any mode, written only in initialization mode when UART bit is set.</td></tr>
<tr><td>0
UART</td><td>UART Mode
Register bit can be read in any mode, written only in initialization mode.
0b - LIN mode
1b - UART mode</td></tr>
</table>

Provided under NDA only.
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-b251-390d3ab624c

48.5.7 UART Mode Status Register (UARTSR)

Offset

Register	Offset
UARTSR	14h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SZF	OCF	PE				RMB	FEF	BOF	RDI	WUF	RFNE	TO	DRFRFE	DTFTFF	NF
W	W1C	W1C	W1C				W1C	W1C	W1C		W1C		W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 SZF	Stuck At Zero Flag LINFlexD writes a 1 to this field when LINFlexD detects 100 dominant bits. It must be cleared by software. An interrupt is generated if the SZIE bit in LINIER is set. This bit will reflect the same value as in LINESR, when not in Initialization mode and the UART bit is set.
14 OCF	Output Compare Flag An interrupt will be generated if the OCIE bit in LINIER is set. This bit will reflect the same value as in LINESR, when not in Initialization mode and the UART bit is set. **NOTE** For Baud rate above 1 Mbit/s, this flag is not usable. 0b - No output compare event occurred 1b - The content of the counter has matched the content of LINOCR
13-10 PE	Parity Error Flag Indicates whether a parity error occurred in the corresponding byte. No interrupt is generated if this error occurs.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit will reflect the same value as in LINESR, when not in initialization mode and UART bit set. Parity is checked only after complete frame is received. Following are the conditions when WL bits = 01 or 11 (either PE0 or PE2 is only set). • When PE0 is set, it indicates Parity error in either the first or second received byte. • When PE2 is set, it indicates Parity error in either the third or fourth received byte. 0000b - No parity error 0001b - Parity error in the corresponding received byte
9 RMB	Release Message Buffer This bit must be cleared by software. This bit reflects the same value as in LINSR, when not in Initialization mode and the UART bit is set. 0b - Buffer data is free 1b - Buffer data ready for software to read
8 FEF	Framing Error Flag LINFlexD writes a 1 to this field when a framing error (invalid stop bit) occurs. It must be cleared by software. It generates an interrupt if the FEIE bit of LINIER is set. This bit will reflect the same value as in LINESR, when not in Initialization mode and the UART bit is set. 0b - No framing error 1b - Framing error occurs
7 BOF	FIFO/Buffer overrun flag This bit is set by hardware when there is a new byte received and the RMB bit is not cleared in UART buffer mode. In UART FIFO mode this bit is set when there is a new byte and the Rx FIFO is full. In UART FIFO mode, once Rx FIFO is full, the new received message will be discarded irrespective of the new value of the RBLM bit. In UART Rx Buffer mode, if RBLM is set then the new message received will be discarded; if RBLM is reset then the new message will overwrite the buffer. It can be cleared by software writing a 1. An interrupt is generated if the BOIE bit of LINIER is set. This bit will reflect the same value as in LINESR, when not in Initialization mode and the UART bit is set.
6 RDI	Receiver Data Input signal This bit reflects the current status of the RX pin when UART bit is set.
5 WUF	Wakeup flag This bit is set by hardware when a falling edge is detected on the RX pin in sleep mode. It should be cleared by software. An interrupt will be generated if the WUIE bit in LINIER is set. This bit will reflect the same value as in LINSR when not in Initialization mode and the UART bit is set.
4 RFNE	Receive FIFO Not Empty

Table continues on the next page...

Table continued from the previous page...

Field	Function
	RFNE bit is set by hardware in UART FIFO mode (RFBM = 1), when there is at least one data byte present in the receive FIFO. RFNE is a read-only bit for debugging purposes. This flag can be used by software in case of a timeout event.
3 TO	Timeout This bit is set by hardware when a UART timeout occurs — in other words, the value of UARTCTO becomes equal to the preset value of the timeout (UARTPTO register setting). TO should be cleared by software. The SR bit should be used to reset the receiver fsm to Idle state in case of UART TIMEOUT for UART reception, depending on the application, in both buffer and FIFO mode. An interrupt will be generated when LINIER.TOIE bit is set on the Error interrupt line in UART mode.
2 DRFRFE	Data Reception Completed Flag /Rx FIFO Empty Flag DRF is set by hardware in UART buffer mode (RFBM = 0) and indicates that the number of bytes programmed in RDFL have been received. DRF should be cleared by software. An interrupt will be generated if the DRIE bit in LINIER is set. DRF bit is set when the configured number of valid stop bits are received for the last frame (number of frames is configurable by RDFL bits). DRF is set irrespective of framing error in case framing error is in the last STOP bit configured, parity error or overrun error. This bit will reflect the same value as in LINSR when not in Initialization mode and the UART bit is set. RFE is set by hardware in UART FIFO mode (RFBM = 1) when the RX FIFO is empty. RFE is a read-only bit for debugging purposes. It is internally used by the DMA RX interface.
1 DTFTFF	Data Transmission Completed Flag/ TX FIFO Full Flag DTF is set by hardware in UART buffer mode (TFBM = 0) and indicates that data transmission is completed. DTF should be cleared by software. An interrupt will be generated if the DTIE bit in LINIER is set. This bit will reflect the same value as in LINSR when not in Initialization mode and the UART bit is set. TFF is set by hardware in UART FIFO mode (TFBM = 1) when TX FIFO is full. TFF is a read-only bit for debugging purposes. It is internally used by the DMA TX interface.
0 NF	Noise flag This bit is set by hardware when noise is detected in the received character. It should be cleared by software. This bit will reflect the same value as in LINESR when not in Initialization mode and the UART bit is set. During reduced oversampling (ROSE bit = 1), it is enabled only when OSR = 8.

48.5.8 LIN Time-Out Control Status Register (LINTCSR)

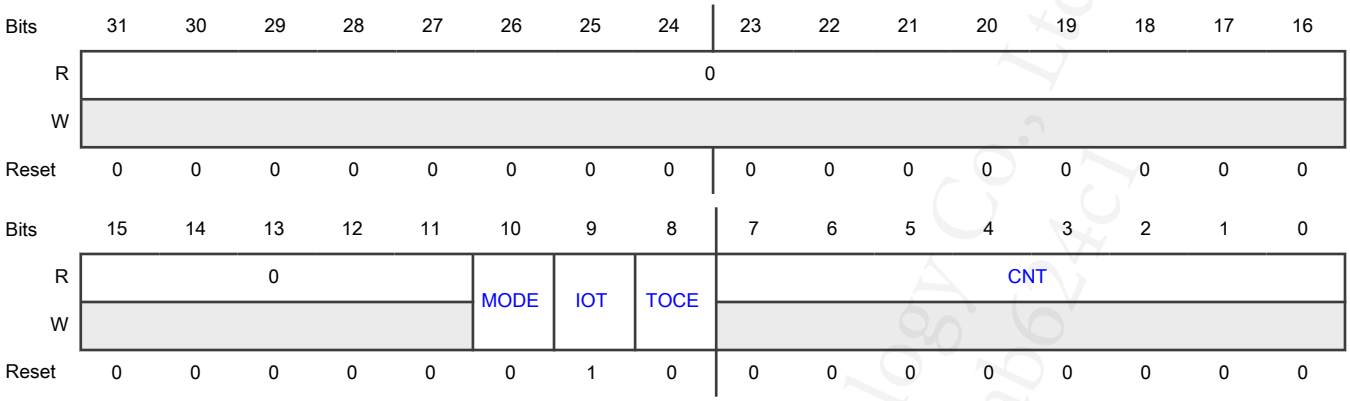
Offset

Register	Offset
LINTCSR	18h

Function

This register contains control and status bits for timeout feature.

Diagram



Fields

Field	Function
31-11 —	Reserved
10 MODE	Time-out counter mode This bit can be configured only during initialization. Register bit can be read in any mode, written only in initialization mode. 0b - LIN mode 1b - Output compare mode
9 IOT	Idle on timeout Register bit can be read in any mode, written only in initialization mode. This feature is applicable only when MODE bit in LINTCSR is cleared. 0b - LIN state machine does not reset to Idle on timeout 1b - LIN state machine resets to Idle on timeout event
8 TOCE	Time-out counter enable TOCE is always configurable by software in Initialization mode. If LIN state is other than INIT and if timer is configured in LIN mode, then hardware takes control of TOCE. 0b - Time-out counter disable. OCF flag is not set on an output compare event. 1b - Time-out counter enable. OCF flag is set if an output compare event occurs.
7-0 CNT	Counter Value These bits reflect the value of a counter used for timeout. **NOTE** For proper functionality of this counter, LINIBRR should be >= 5.

Provided under NDA only
COMPANY PROPRIETARY
Copyright (c) 2018-2021 Beijing CHIPANALOG Automotive Technology Co., Ltd.
Agreement No. 20210825-1251-390d3ab624fc1

48.5.9 LIN Output Compare Register (LINOCR)

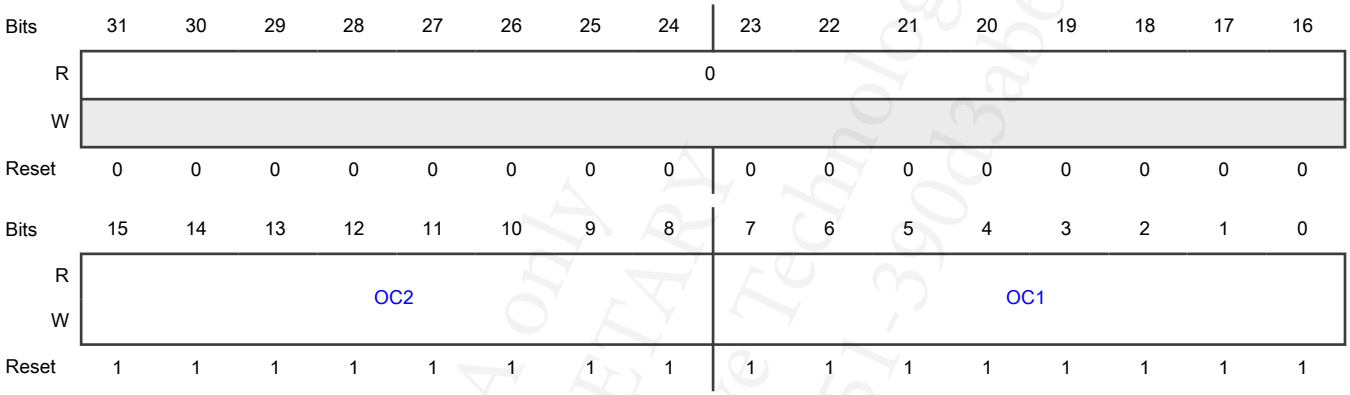
Offset

Register	Offset
LINOCR	1Ch

Function

This register contains the value to be compared to the LINTCSR:CNT value. This register is writable by software only in Output Compare Mode.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 OC2	Output compare value 2
7-0 OC1	Output compare value 1

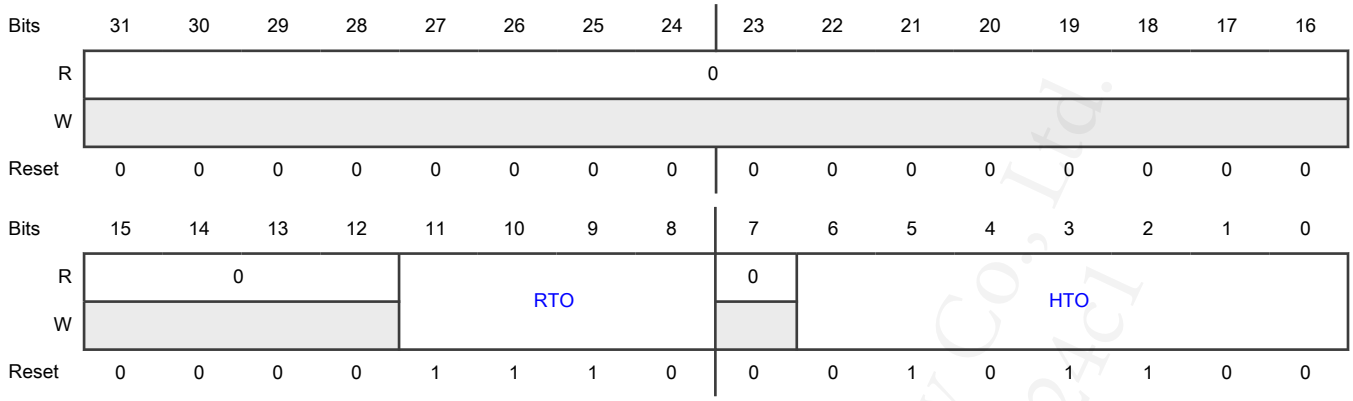
48.5.10 LIN Time-Out Control Register (LINTOCR)

Offset

Register	Offset
LINTOCR	20h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e6-4821-48a8-b251-39003ab62451

Diagram



Fields

Field	Function
31-12 —	Reserved
11-8 RTO	Response timeout value This is the response timeout duration (in bit time) for 1 byte. The reset value is 0Eh = 14, corresponding to $T_{Response\_Maximum} = 1.4 \times T_{Response\_Nominal}$
7 —	Reserved
6-0 HTO	Header timeout value This register contains the header timeout duration (in bit time). This register can be written only for Slave mode. For slave, these bits should be programmed without considering 11 bits of break.

48.5.11 LIN Fractional Baud Rate Register (LINFBRR)

Offset

Register	Offset
LINFBRR	24h

Function

This register consists of bits that decide the fractional part of the LIN Baud Rate. It can be programmed only in Initialization mode.

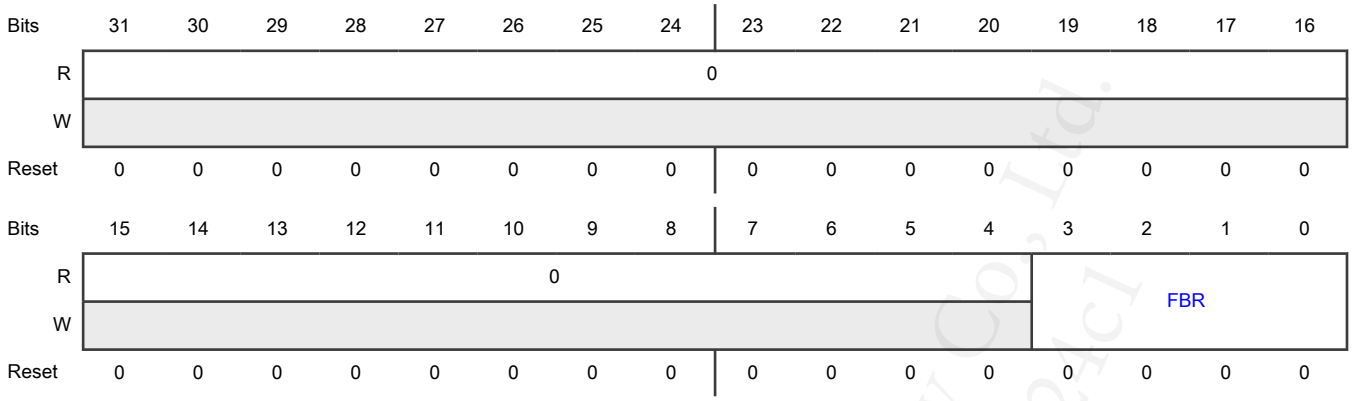
NOTE

When LASE bit is set, this register should be read only after AUTOSYNC_COMP bit in LINSR register is set to obtain the correct value.

NOTE

This register cannot be used when reduced oversampling is enabled (ROSE bit = 1)

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 FBR	Fractional Baud rates Register bit can be read in any mode, written only in initialization mode. 0000b - Fraction(LDIV) = 0 0001b - Fraction(LDIV) = 1/16 0010b - Fraction(LDIV) = 2/16 0011b - Fraction(LDIV) = 3/16 0100b - Fraction(LDIV) = 4/16 0101b - Fraction(LDIV) = 5/16 0110b - Fraction(LDIV) = 6/16 0111b - Fraction(LDIV) = 7/16 1000b - Fraction(LDIV) = 8/16 1001b - Fraction(LDIV) = 9/16 1010b - Fraction(LDIV) = 10/16 1011b - Fraction(LDIV) = 11/16 1100b - Fraction(LDIV) = 12/16 1101b - Fraction(LDIV) = 13/16 1110b - Fraction(LDIV) = 14/16 1111b - Fraction(LDIV) = 15/16

Provided by IHS Markit
COMPANY PROPRIETARY
Beijing CAR Automotive Technology Co., Ltd.
c8c26524821-48a8-b251-390d8ab624c1

48.5.12 LIN Integer Baud Rate Register (LINIBRR)

Offset

Register	Offset
LINIBRR	28h

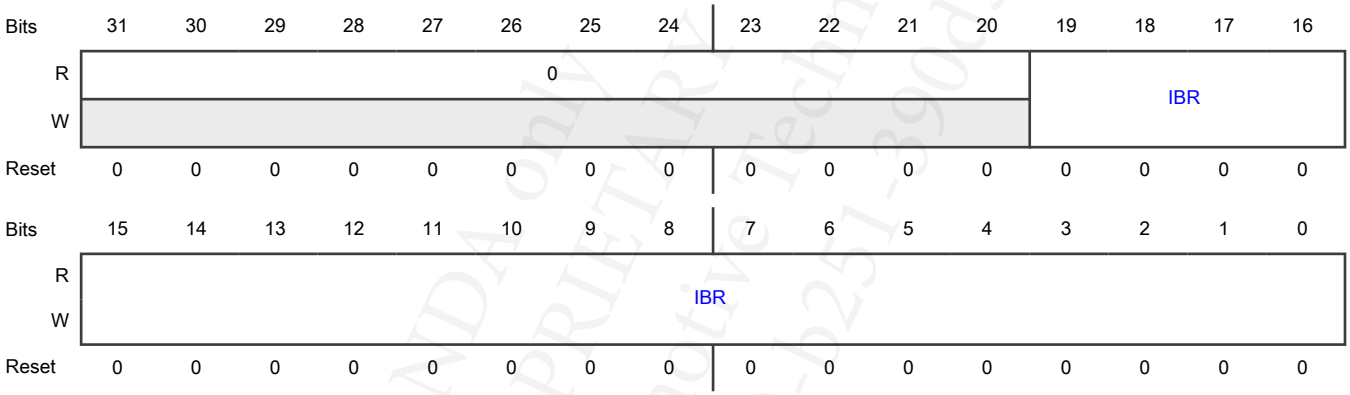
Function

This register consists of control bits that decide the baud rate along with the LINFBRR. It can be programmed only in Initialization mode.

NOTE

When LASE bit is set, this register should be read only after AUTOSYNC_COMP bit in LINSR register is set to obtain the correct value.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-0 IBR	Integer Baud rates These bits along with the fractional baud rate bits decide the LIN baud rate. IBR = 0h: LIN clock disabled IBR = 1h: Mantissa (LDIV) = 1 ... IBR = FFFFEh: Mantissa (LDIV) = 1048574 IBR = FFFFFh: Mantissa (LDIV) = 1048575 Register bit can be read in any mode, written only in initialization mode.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c0462e5-4821-48a8-b251-a9003ab0240d

48.5.13 LIN Checksum Field Register (LINCFR)

Offset

Register	Offset
LINCFR	2Ch

Function

NOTE

There is a delay between 4 to 6 clock cycles of LIN_XBAR_CLK for the internal checksum's value (which is clocked with LIN_CLK/16 * LDIV) to reflect on LINCFR.

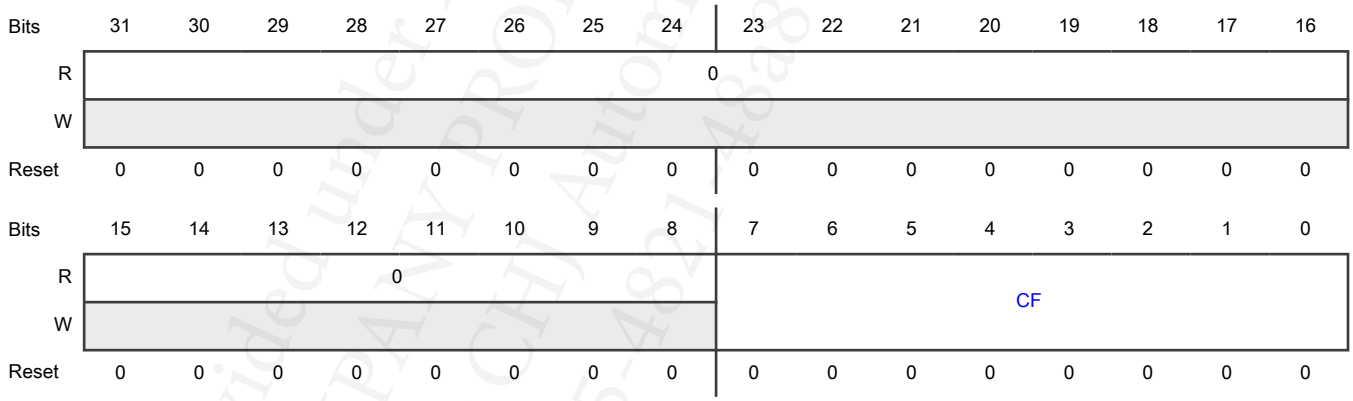
This register consists of checksum bits.

NOTE

Due to synchronization structures between the two input clocks to LINFlexD, the write to this register is delayed between 2 to 3 clock cycles and read is delayed between 4 to 6 clock cycles of LIN_XBAR_CLK .

CFD	CCD	LINCHKSUM read/write	Checksum sent
1	1	read/write	None
1	0	read-only	None
0	1	read/write	Programmed checksum
0	0	read-only	Calculated checksum

Diagram



Fields

Field	Function
31-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 CF	Checksum bits When the CCD bit is reset these bits are read-only and are calculated by hardware. When the CCD bit is set, these bits can be written by software.

48.5.14 LIN Control Register 2 (LINCR2)

Offset

Register	Offset
LINCR2	30h

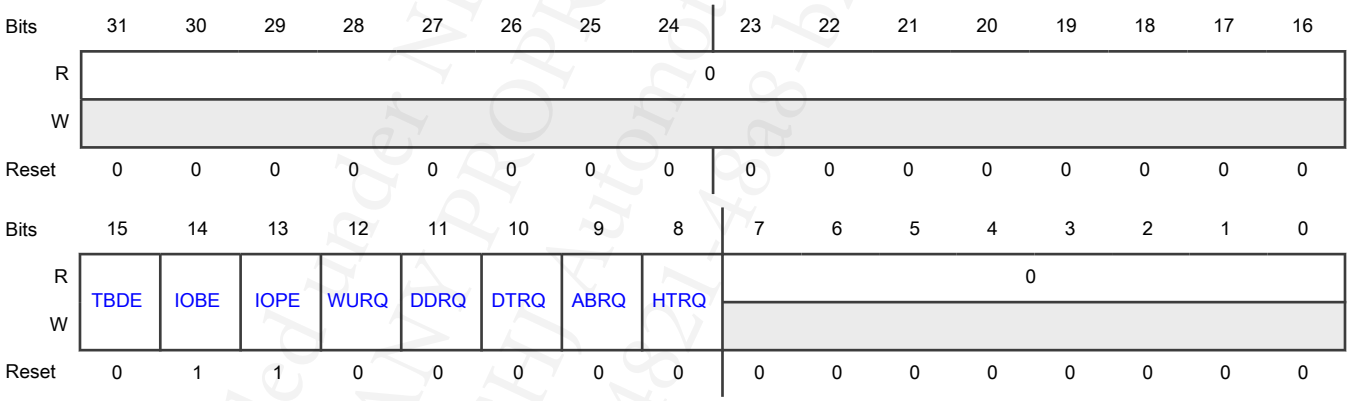
Function

This register includes control status bits related to buffer operations.

NOTE

- When accessing the LINCR2 register, each reserved bit should be written to its original reset value.
- In Normal mode, write 1 to only those fields that you must configure and write 0 to all the other non-reserved fields. Because only the module clears these fields, writing 0 does not change their current values.

Diagram



Fields

Field	Function
31-16 —	Reserved When accessing the LINCR2 register, each reserved bit should be written to its original reset value.
15 TBDE	Two Bit delimiter bit This bit can be set in Initialization mode only.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Register bit can be read in any mode, written only in initialization mode. 0b - Delimiter length in break field is 1 bit 1b - Delimiter length in break field is 2 bits
14 IOBE	Idle on Bit Error This bit can be set in Initialization mode only. Register bit can be read in any mode, written only in initialization mode. 0b - Bit Error does not reset LIN state machine 1b - Bit Error resets LIN state machine
13 IOPE	Idle on Identifier Parity Error This bit can be set in Initialization mode only. Register bit can be read in any mode, written only in initialization mode. **NOTE** If generic slave = 0, then this bit will always read a '0', and cannot be programmed. 0b - Parity Error does not reset LIN state machine 1b - Parity Error resets LIN state machine
12 WURQ	Wakeup Generate Request Setting this bit will generate a wakeup pulse. It is reset by hardware when the wakeup character has been transmitted. The character sent during wakeup is copied from BDRL (DATA0). Note that this bit cannot be set in Sleep mode — software has to exit Sleep mode before setting this bit. Bit Error is not checked when transmitting the wakeup request. Register bit can be read/set by software
11 DDRQ	Data Discard request Set by software to stop data reception if the frame does not concern the node. This bit is reset by hardware once LINFlexD ignores the response and moves to Idle state. For LIN slave this bit can be set only when HRF bit is set and Identifier is software-filtered. Register bit can be read/set by software
10 DTRQ	Data Transmission Request Set by software in slave mode to request the transmission of the LIN Data field stored in the Buffer data register. This bit can be set only when the HRF bit is set (to ensure that data transmission is requested only after a header reception). Cleared by hardware when the request has been completed, or on abort request or error condition. In Master mode, this bit is set by hardware when the DIR bit is set and header transmission is complete. Register bit can be read/set by software

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 ABRQ	Abort Request Set by software to abort the current transmission. Cleared by hardware when the transmission has been aborted. LINFlexD aborts the transmission at the end of the current bit. This bit can abort a wakeup request also and can be used in UART mode also. Register bit can be read/set by software
8 HTRQ	Header Transmission Request Set by software to request the transmission of the LIN Header. Cleared by hardware when the request has been completed or on abort request. This bit has no effect in UART mode. **NOTE** In master mode, if both HTRQ and ABRQ are set at the same time then ABRQ has no effect. Similarly, in slave mode after header reception, if DTRQ and ABRQ are simultaneously set then ABRQ has no effect.
7-0 —	Reserved When accessing the LINCR2 register, each reserved bit should be written to its original reset value.

48.5.15 Buffer Identifier Register (BIDR)

Offset

Register	Offset
BIDR	34h

Function

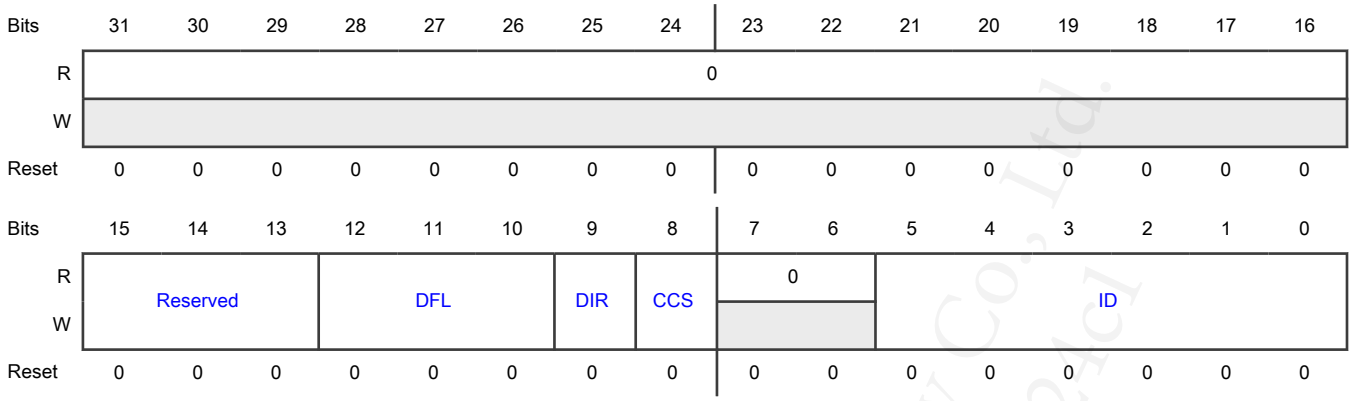
This register contains the bits which provide information about the identifier of the transaction and other related information.

NOTE

All the fields (ID, CSS, DIR, DFL) of the BIDR register must be updated when an ID filter (enabled) in Slave mode (Tx or Rx) matches the ID received.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
c8c462e5-4e9-48a8-b251-390cab624c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15-13 —	Reserved
12-10 DFL	Data Field Length Number of data bytes in the response part of the frame. DFL = Number of data bytes - 1
9 DIR	Direction This bit controls the direction of the data field. 0b - LINFlexD receives the data and copy them in the BDR registers 1b - LINFlexD transmits the data from the BDR registers
8 CCS	Classic Checksum This bit controls the type of checksum applied on the current message. 0b - Enhanced Checksum covering Identifier and Data fields. This is compatible with LIN specification rev. 2.0 and higher. 1b - Classic Checksum covering Data filed only. This is compatible with LIN specification 1.3 and lower.
7-6 —	Reserved
5-0 ID	Identifier Identifier part of the identifier field without the identifier parity. This field can be written only in Master mode (MME = '1').

Provided under NDA only
COMPANY PROPRIETARY
Beijing C.T.A Automotive Technology Co., Ltd.
c8462e54-41ca8-b251-390d8ab624c1

48.5.16 Buffer Data Register Least Significant (BDRL)

Offset

Register	Offset
BDRL	38h

Function

This register is a part of an 8-byte data buffer. See Table 310 for register access in UART mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATA3								DATA2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA1								DATA0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 DATA3	Data Byte 3 Data byte 3 of the data field.
23-16 DATA2	Data Byte 2 Data byte 2 of the data field.
15-8 DATA1	Data Byte 1 Data byte 1of the data field.
7-0 DATA0	Data Byte 0 Data byte 0 of the data field.

48.5.17 Buffer Data Register Most Significant (BDRM)

Offset

Register	Offset
BDRM	3Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390da3b624c1

Function

This register is a part of an 8-byte data buffer. See Table 311 for register access in UART mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DATA7								DATA6							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA5								DATA4							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 DATA7	Data Byte 7 Data byte 7 of the data field.
23-16 DATA6	Data Byte 6 Data byte 6 of the data field.
15-8 DATA5	Data Byte 5 Data byte 5 of the data field.
7-0 DATA4	Data Byte 4 Data byte 4 of the data field.

48.5.18 Global Control Register (GCR)

Offset

Register	Offset
GCR	4Ch

Function

This register is read-only in Normal mode and can be programmed only in Initialization mode. This is a global control register — in other words, the register configuration will be applied in LIN mode as well as in UART mode.

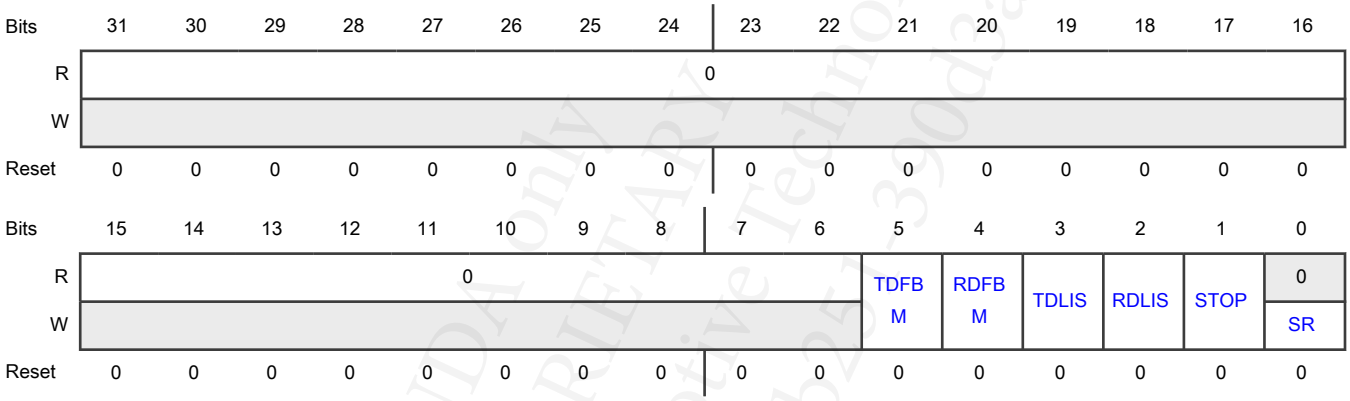
The address offset depends on the no_of_filters. See chip configuration details for the number of filters used in this device.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8162e5-4821-48a8-b251-390d3ab624c1

Table 321. Register fields reset by SR

Register	Comment
LINSR	All fields except RXBUSY and AUTOSYNC_COMP are reset
LINESR	All fields are reset
LINTCSR	Only CNT[0:7] is reset
UARTSR	All fields except RFE & TFF are reset
UARTCR	Only TFC & RFC are reset
UARTCTO	All fields are reset

Diagram



Fields

Field	Function
31-6 —	Reserved
5 TDFBM	Transmit data first bit MSB This bit controls the first bit of transmit data (payload only) as MSB/LSB in both UART and LIN modes. Register bit can be read in any mode, written only in initialization mode. 0b - The first bit of transmitted data is LSB — in other words, the first bit transmitted is mapped on LSB bit (BDR (0), BDR (8), BDR (16), BDR (24)) 1b - The first bit of transmitted data is MSB — in other words, the first bit transmitted is mapped on MSB bit (BDR (7), BDR (15), BDR (23), BDR (31))
4 RDFBM	Received data first bit MSB This bit controls the first bit of received data (payload only) as MSB/LSB both in UART and LIN modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Register bit can be read in any mode, written only in initialization mode. 0b - The first bit of received data is LSB — in other words, the first bit received is mapped on LSB bit (BDR (0), BDR (8), BDR (16), BDR (24)) 1b - The first bit of received data is MSB — in other words, the first bit received is mapped on MSB bit (BDR (7), BDR (15), BDR (23), BDR (31))
3 TDLIS	Transmit data level inversion selection This bit controls the data inversion of transmitted data (payload only) in both UART and LIN modes. Register bit can be read in any mode, written only in initialization mode. 0b - Transmitted data is not inverted 1b - Transmitted data is inverted
2 RDLIS	Received data level inversion selection This bit controls the data inversion of received data (payload only) in both UART and LIN modes. Register bit can be read in any mode, written only in initialization mode. 0b - Received data is not inverted 1b - Received data is inverted
1 STOP	1/2 stop bit configuration This bit controls the number of stop bit transmitted data in both UART and LIN modes. The stop bit is configured for all the fields (Delimiter, Sync, ID, Checksum, Payload). Register bit can be read in any mode, written only in initialization mode. 0b - 1 stop bit 1b - 2 stop bits
0 SR	Soft reset SR executes a soft reset of the LINFlexD controller (FSMs, FIFO pointers, counters, timers, status and error registers) without modifying the configuration registers when a 1 write operation is performed. This bit should be cleared by software to perform further operations (this bit is not cleared by hardware). Register bit can be written only be software in initialization mode. Table 321 describes the register fields reset by SR.

48.5.19 UART Preset Timeout Register (UARTPTO)

Offset

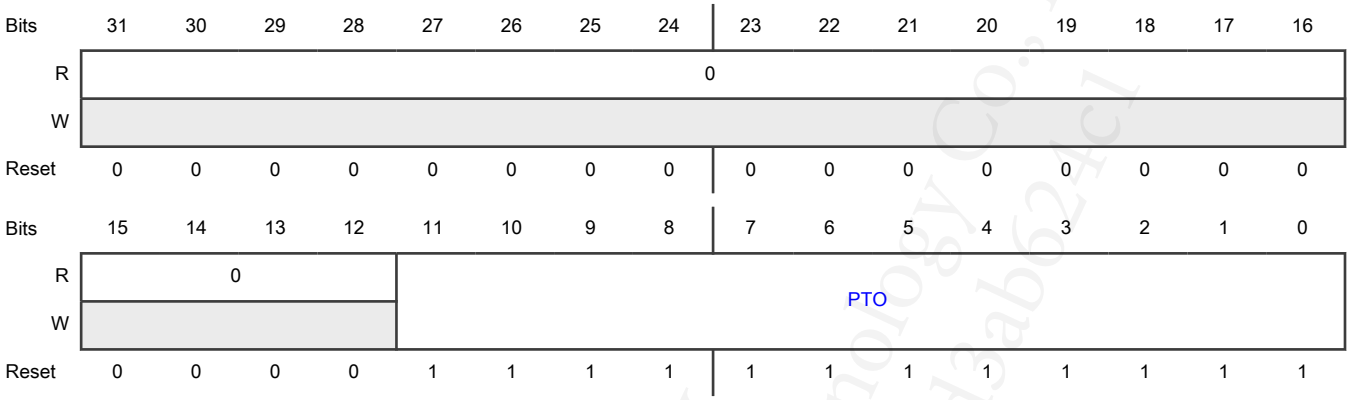
Register	Offset
UARTPTO	50h

Function

This register contains the preset value of the timeout register in UART mode and is programmed according to the number of bits to be received or to monitor the idle state of the reception line. This register can be written by software any time.

The address offset depends on no_of_filters. See chip configuration details for the number of filters used in the device.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 PTO	Preset Timeout PTO defines the preset value of timeout counter. A zero-value is forbidden, otherwise the UARTSR[TO] status bit is immediately set. Also see register UARTCTO.

48.5.20 UART Current Timeout Register (UARTCTO)

Offset

Register	Offset
UARTCTO	54h

Function

This register contains the current timeout value in UART mode, and is used in conjunction with the UARTPTO register (see UART Preset Timeout Register (UARTPTO)) to monitor the number of bits received by UART or to monitor the idle state of the reception line. UART timeout works in both CPU and DMA modes.

The timeout counter:

- Starts at zero and counts upward
- Is clocked with LIN_CLK / (16 * LDIV) synchronized to LIN_XBAR_CLK (when ROSE = 0)
- Is clocked with LIN_CLK / (OSR * IBRR) synchronized to LIN_XBAR_CLK (when ROSE = 1)
- Is automatically enabled when UARTCR[RXEN] = 1

NOTE

Due to this synchronization there is a delay between 4 to 6 clock cycles of LIN_XBAR_CLK for the internal counter's value (which is clocked with LIN_CLK/16 * LDIV when ROSE = 0 or LIN_CLK/OSR * IBRR when ROSE = 1) to reflect on UARTCTO.

It is reset when:

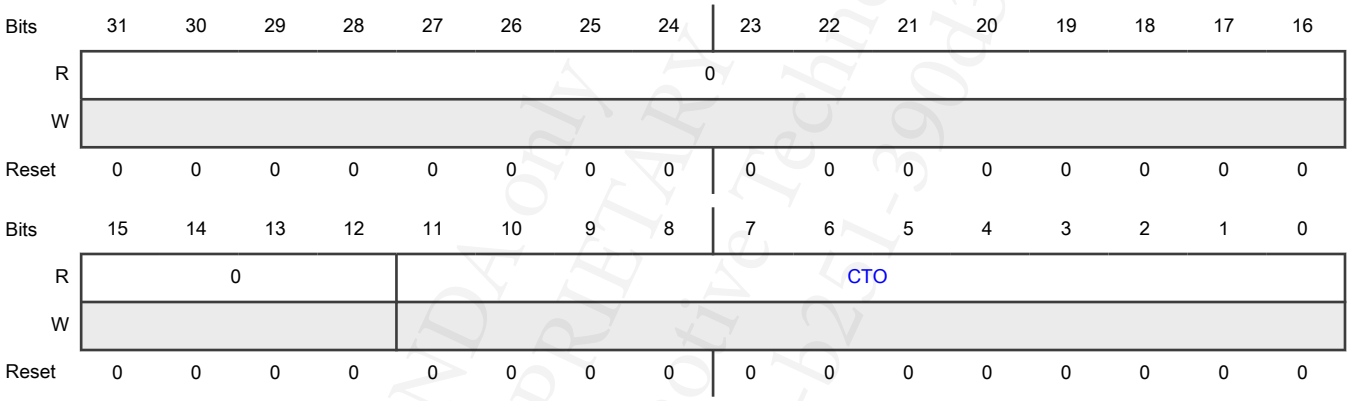
- Number of frames received is equal to NEF bits
- UARTCTO becomes equal to UARTPTO
- Whenever UARTPTO is written
- When DRF is set and DTU bit = 1

The address offset depends on no_of_filters. See chip configuration details for the number of filters used in the device.

NOTE

Write access to this RO register generates transfer error.

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 CTO	Current Timeout CTO defines the current value of the timeout counter. CTO is a read-only field. CTO is reset every time UARTPTO is re-initialized, or UARTCTO = UARTPTO, or by hard/soft reset. When the CTO value matches the preset value (UARTPTO), the status bit UARTSR[TO] is set.

48.5.21 DMA Tx Enable Register (DMATXE)

Offset

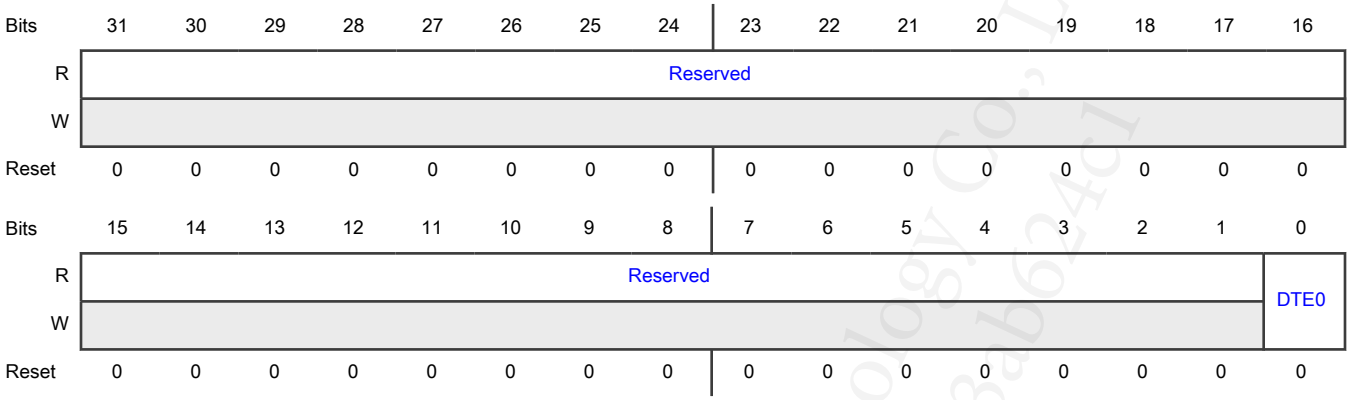
Register	Offset
DMATXE	58h

Function

This register enables the DMA TX interface. This register can be written and read by software anytime.

The address offset depends on no_of_filters. See chip configuration details for the number of filters used in the device.

Diagram



Fields

Field	Function
31-1 —	Reserved
0-0 DTEn	DMA Tx channel enable The number of DTE bits varies, and is equal to DMA_TX_CH_NUM. When DMATXE = 0x00000000, the DMA TX interface FSM is forced (soft reset) in Idle state. **NOTE** See chip configuration details for the value of DMA_TX_CH_NUM used in this device. **NOTE** The value of n is 0. 0b - nth DMA Tx channel disabled 1b - nth DMA Tx channel enabled

48.5.22 DMA Rx Enable Register (DMARXE)

Offset

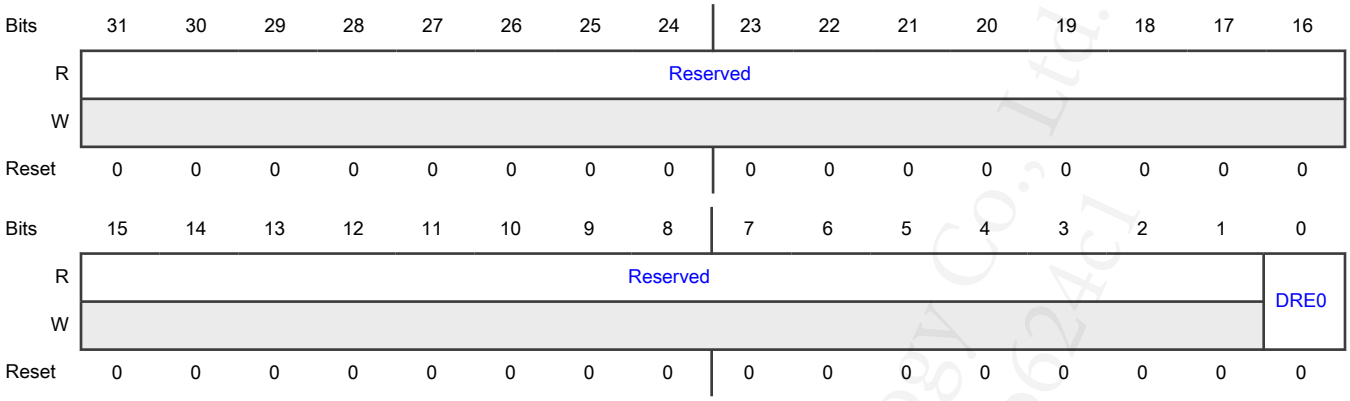
Register	Offset
DMARXE	5Ch

Function

This register enables the DMA RX interface. This register can be written and read by software any time.

The address offset depends on no_of_filters. See device configuration chapter for the number of filters used in the device.

Diagram



Fields

Field	Function
31-1 —	Reserved
0-0 DREn	DMA Rx channel enable The number of DRE bits varies, and is equal to DMA_RX_CH_NUM. When DMARXE = 0x00000000, the DMA RX interface FSM is forced (soft reset) in Idle state. **NOTE** See chip configuration details for the value of DMA_RX_CH_NUM used in this device. **NOTE** The value of n is 0. 0b - nth DMA Rx channel disabled 1b - nth DMA Rx channel enabled

48.6 Programming considerations

The next subsections describe the various configurations in which the LINFlexD module can be used.

LINFlexD writes a 1 to multiple fields (such as status fields). It also writes a 0 to multiple other fields (such as LINCR2[HTRQ] and LINCR2[ABRQ]). Software must check the status of the bits before initiating the next sequence to ensure that LINFlexD completed its work. This ensures that LINFlexD and software are working in sync.

48.6.1 Master node

LINFlexD acts as master when LINCR1[MME]=1.

48.6.1.1 Transmitter

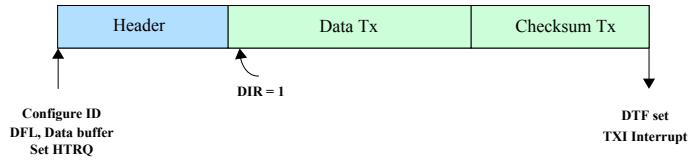


Figure 396. Master node – transmitter configuration

48.6.1.2 Receiver

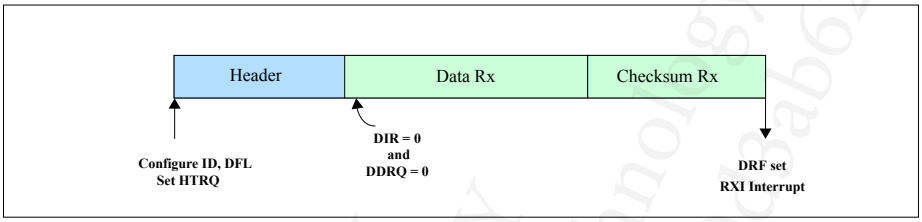


Figure 397. Master node – receiver configuration

48.6.1.3 Transmitter, bit error

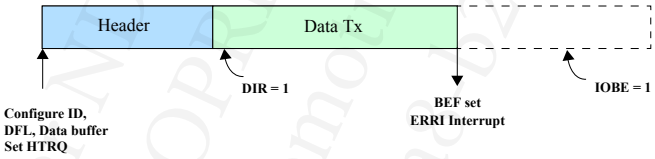


Figure 398. Master node – transmitter, bit error configuration

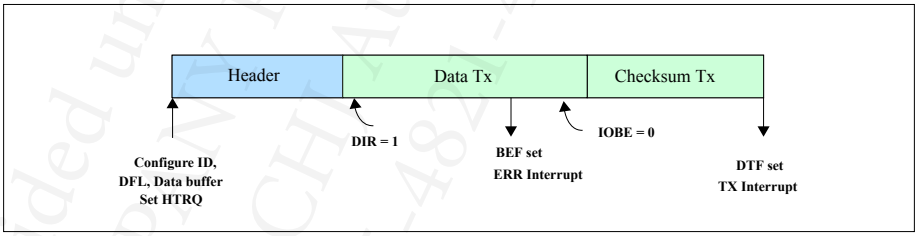


Figure 399. Master node – transmitter, checksum error configuration

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHANGEMOTIVE Technology Co., Ltd.
c8c462e5-482f-48ad-8251-3903ab624c1

48.6.1.4 Receiver, checksum error

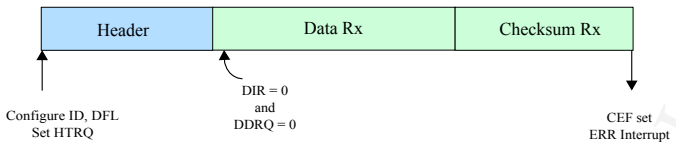


Figure 400. Master node – receiver, checksum error configuration

48.6.2 Slave node

48.6.2.1 Transmitter (no identifier filters)

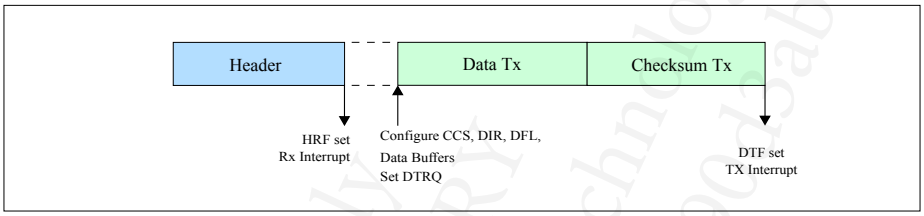


Figure 401. Slave node – transmitter (no identifier filters) configuration

48.6.2.2 Receiver (no identifier filters)

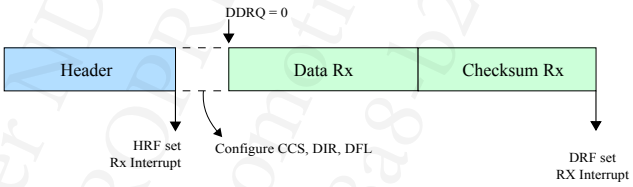


Figure 402. Slave node – receiver (no identifier filters) configuration

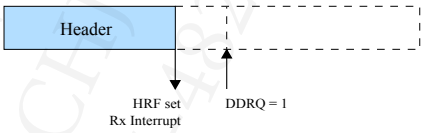


Figure 403. Slave node – receiver (no identifier filters) configuration

48.6.2.3 No filters, transmitter, bit error

The following figure shows Slave node - no filters, transmitter, bit error configuration.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c84625432148 2021-3-9 03:51:39

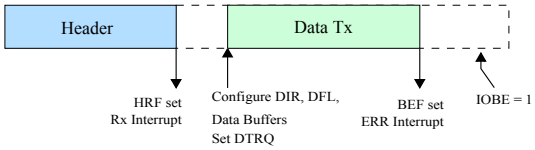


Figure 404. Slave node – no filters, transmitter, bit error configuration

48.6.2.4 No filters, receiver, checksum error

The following figure shows Slave node - no filters, receiver, checksum error configuration.

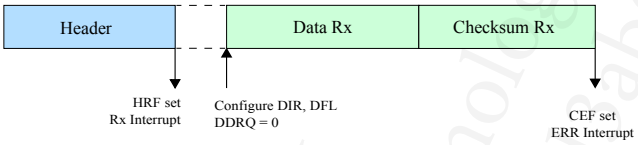


Figure 405. Slave node – no filters, receiver, checksum error configuration

48.6.3 Timeout

Master Node: Response (during reception only) and frame timeout are checked.

Slave Node: Header, Response (during reception only), and frame timeout are checked.

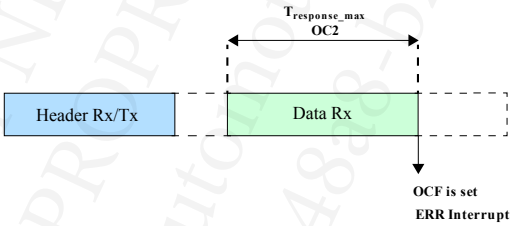


Figure 406. Response timeout

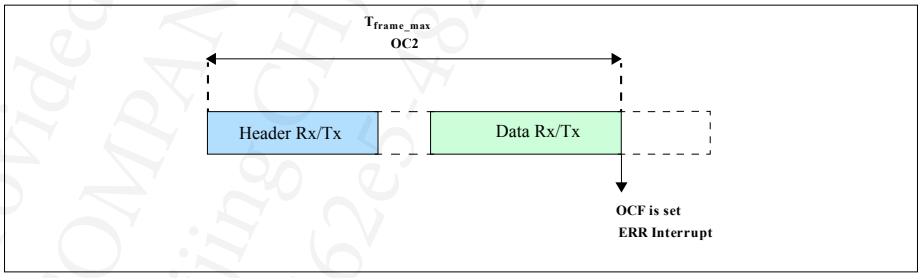


Figure 407. Frame timeout

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4621-4848-b521-390dab624c1

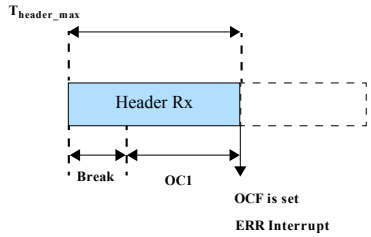


Figure 408. Header timeout

48.6.4 UART mode

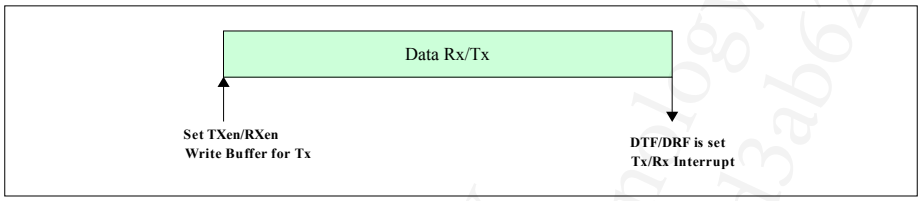


Figure 409. UART mode configuration

48.6.5 Interrupts

This section describes interrupt handling and interrupt concentration. Interrupt listed in Table 322 are corresponding to status registers LINSR, LINESR, and UARTSR. Corresponding interrupt enables bit are present in LINIER register. Instead of having multiple interrupt output lines, these interrupts are concentrated into three output signals as shown in Figure 410.

Table 322. Interrupts

Interrupt event	Event flag	Enable control bit	Interrupt vector
Stuck at zero	SZF	SZIE	Error
Output compare	OCF	OCIE	Error
Bit error	BEF	BEIE	Error
Checksum error	CEF	CEIE	Error
Header error	SFEF orSDEF orIDPEF	HEIE	Error
Frame error	FEF	FEIE	Error
Buffer overrun error	BOF	BOIE	Error
UART timeout error	TO	TOIE	Error
LIN state	Sync Del, Sync Field,Identifier field or Checksum Field	LSIE	Rx
Wakeup	WUF	WUIE	Rx

Table continues on the next page...

Table 322. Interrupts (continued)

Interrupt event	Event flag	Enable control bit	Interrupt vector
Data reception complete	DRF	DRIE	Rx
Data transmitted	DTF	DTIE	Tx
Header received	HRF	HRIE	Rx
Header received	HRF	HRIE	Tx (if there is a filter match for Tx identifier)

The following figure shows an interrupt flow diagram.

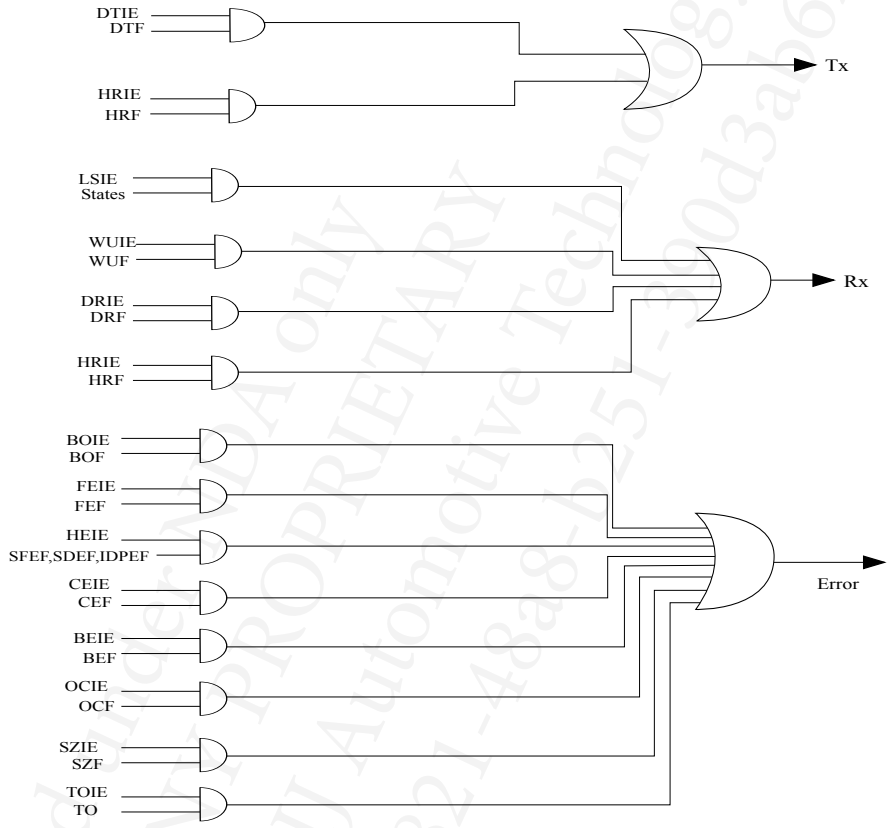


Figure 410. Interrupt diagram

Rx interrupt due to LIN state event is cleared by writing LINSR[LINS] field with 0xF when LSIE is enabled. In loopback mode, Rx interrupt is set earlier than Tx interrupt.

48.6.6 LINFlexD clock tolerance

48.6.6.1 Faster receiver tolerance

In this case, the receiver has a higher baud rate than the transmitter, thus the stop bit sampling starts already in the last transmitted payload bit. To ensure the correct noise and framing error free reception bit, the samples RS8, RS9, and RS10 must be located in the transmitted stop bit as shown in the following figure.

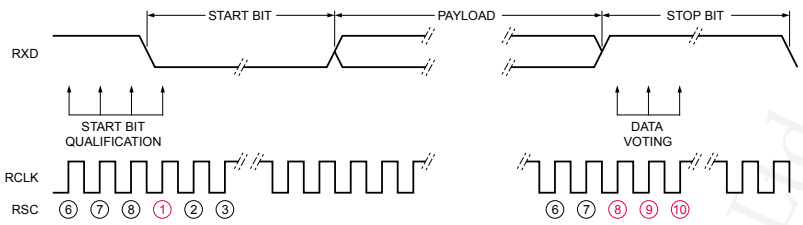


Figure 411. Faster receiver

48.6.6.2 Slower receiver tolerance

In this case, the receiver has a slower baud rate than the transmitter. Therefore, the stop bit sampling is still running while transmission of the next start bit is already occurring. To ensure the correct noise and framing error free reception bit , the samples RS8, RS9, and RS10 must be located in the transmitted stop bit as shown in the following figure.

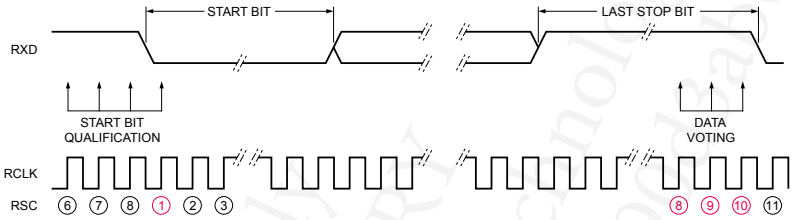


Figure 412. Slower receiver

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39c0ba6b24c1

Chapter 49
Serial Peripheral Interface (SPI)

49.1 Chip-specific SPI information

49.1.1 Module instance information

This chip contains six instances of SPI. The following table shows the configuration of each instance.

Description	SPI_0	SPI_1	SPI_2	SPI_3	SPI_4	SPI_5
Extended (32-bit) SPI mode support	1	1	1	1	1	1
CSI and DSI mode support	0	0	0	0	0	0
TX FIFO	5	5	5	5	5	5
RX FIFO	5	5	5	5	5	5
CMD FIFO	5	5	5	5	5	5
CTAR registers (transfer attribute configuration)	6	6	6	6	6	6
PCS signals	8	5	5	5	5	5
Mode	Master/Slave	Master/Slave	Master/Slave	Master/Slave	Master/Slave	Master/Slave

49.2 Introduction

The SPI module provides a synchronous serial bus for communication between a chip and an external peripheral device.

49.2.1 Block diagram

The following figure represents the module's block diagram.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-9251-3900d3ab624f:1

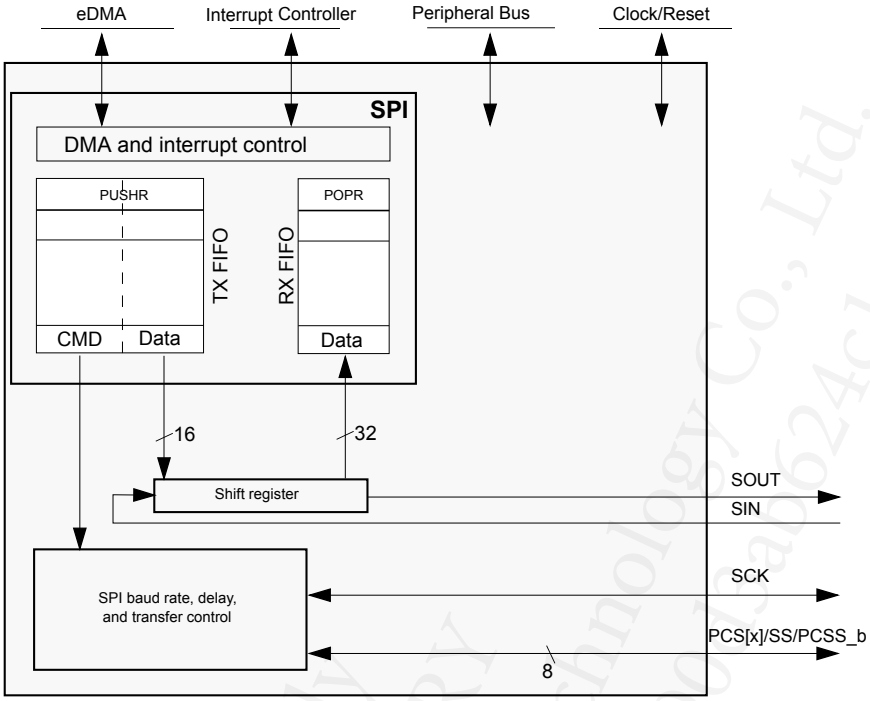


Figure 413. SPI block diagram

49.2.2 Features

The module supports the following features:

- Full-duplex, three-wire synchronous transfers
- Master mode
- Slave mode
- Data streaming operation in the Slave mode with continuous slave selection
- Buffered transmit operation using transmit first in first out (TX FIFO) with a depth of 5 entries
- Support for 8- or 16-bit accesses to PUSHR_SLAVE[TXDATA]
- Buffered receive operation using the Receive FIFO (RX FIFO) with a depth of 5 entries
- TX and RX FIFOs can be disabled individually for low-latency updates to SPI queues
- Visibility into TX and RX FIFOs for ease of debugging
- Programmable transfer attributes on a per-frame basis:
  - 6 transfer attribute registers
  - 6 extended transfer attribute registers
  - Serial clock (SCK) with programmable polarity and phase
  - Various programmable delays
  - Programmable serial frame size: 4 to 32
    - The continuous selection format can support SPI frames longer than 32 bits.
  - Continuously held chip-select capability
  - Parity control

- 8 peripheral chip selects (PCSes), expandable to 2 ^ (8 – 1) with an external demultiplexer
- Deglitching support for up to 8 peripheral chip selects (PCSes) with external demultiplexer
- DMA support for adding entries to TX FIFO and removing entries from RX FIFO when:
  - TX FIFO is not full (TFFF)
  - RX FIFO is not empty (RFDF)
  - Command (CMD) FIFO is not full (CMDFFF)
- Interrupt conditions:
  - End of queue reached (EOQF)
  - TFFF
  - CMDFFF
  - Transfer of current frame complete (TCF)
  - Transfers due from current command frame complete (CMDTCF)
  - Attempt to transmit with an empty transmit FIFO (TFUF)
  - RFDF
  - Frame received while RX FIFO is full (RFOF)
  - SPI Parity Error (SPEF)
  - Data present in TX FIFO while CMD FIFO is empty (TFIWF)
- Global interrupt request line
- Modified SPI transfer formats for communication with slower peripheral devices

49.2.3 Interface configurations

49.2.3.1 SPI configuration

The SPI configuration allows the module to send and receive serial data. This configuration allows the module to operate as a basic SPI block with internal FIFOs supporting the external queue operation. Transmitted data and received data are included in separate FIFOs. The host CPU or a DMA controller reads the received data from the RX FIFO and writes transmit data to the transmit FIFO.

For queued operations, the SPI queues can be included in system RAM, external to the module. A DMA controller or host CPU accomplishes data transfers between the queues and the module FIFOs. The following figure shows a system example with DMA, SPI, and external queues in system RAM.

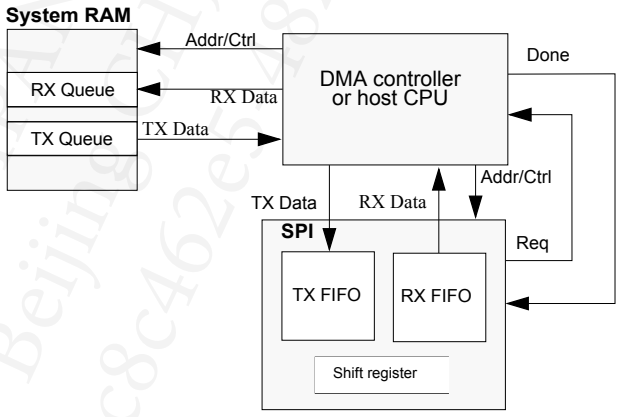


Figure 414. SPI with queues and DMA

49.2.4 Modes of operation

The module supports the following modes of operation that can be divided into two categories:

- Module-specific modes:
  - Master mode
  - Slave mode
  - Module Disable mode
- Chip-specific modes:
  - Debug mode

The module enters module-specific modes when the host writes to a module register. The signals external to the module control the chip-specific modes. The chip may enter these modes in parallel to the module-specific modes.

49.2.4.1 Master mode

The Master mode allows the module to initiate and control serial communication. In this mode, the module controls the following signals, which the module configures as outputs:

- SCK
- SOUT
- PCS[*x*]

49.2.4.2 Slave mode

The Slave mode allows the module to communicate with SPI bus masters. In this mode, the module responds to externally controlled serial transfers. The module configures the SCK signal and the PCS[0]/SS_b signals as inputs, which are driven by the SPI bus master.

49.2.4.3 Module Disable mode

The Module Disable mode can be used for chip-power management. In this mode, you can stop the clock to the non-memory mapped logic in the module.

49.2.4.4 Debug mode

The Debug mode is used for system development and debugging. MCR[FRZ] controls module behavior when the chip is in this mode.

- If you write a 1 to this field, the module stops all serial transfers .
- If you write a 0 to the field, this mode has no effect on the module.

49.3 Module signal descriptions

This table describes the signals on the boundary of the module that may connect off chip (in alphabetical order).

Table 323. Module signal descriptions

Signal	Master mode	Slave mode	I/O
PCS0/SS_b	Peripheral Chip Select 0 (O)	Slave Select (I)	I/O
PCS1	Peripheral Chip Select 1	Unused	O

Table continues on the next page...

Table 323. Module signal descriptions (continued)

Signal	Master mode	Slave mode	I/O
PCS2	Peripheral Chip Select 2	Unused	O
PCS3	Peripheral Chip Select 3	Unused	O
PCS4	Peripheral Chip Select 4	Unused	O
PCS5/PCSS_b	Peripheral Chip Select 5/Peripheral Chip Select Strobe	Unused	O
PCS[6:7]	Peripheral Chip Selects 6–7	Unused	O
SCK	Serial Clock (O)	Serial Clock (I)	I/O
SIN	Serial Data In	Serial Data In	I
SOUT	Serial Data Out	Serial Data Out	O

49.3.1 PCS0/SS_b

Master mode: Peripheral Chip Select 0 (O)—Selects SPI slave to receive data transmitted from the module

Slave mode: Slave Select (I)—Selects the module to receive data transmitted from the SPI master

NOTE

Do not tie the SPI slave select pin to ground. Otherwise, SPI cannot function properly.

49.3.2 PCS1

Master mode: Peripheral Chip Select 1 (O)—Selects SPI slave to receive data that the module transmits

Slave mode: Unused

49.3.3 PCS2

Master mode: Peripheral Chip Select 2 (O)—Selects SPI slave to receive data that the module transmits

Slave mode: Unused

49.3.4 PCS3

Master mode: Peripheral Chip Select 3 (O)—Selects SPI slave to receive data that the module transmits

Slave mode: Unused

49.3.5 PCS4

Master mode: Peripheral Chip Select 4 (O)—Selects SPI slave to receive data that the module transmits

Slave mode: Unused

49.3.6 PCS5/PCSS_b

Master mode:

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-300d3ab62fc1

- Peripheral Chip Select 5 (O)—Used only when the peripheral-chip-select strobe is disabled (MCR[PCSSE]). Selects SPI slave to receive data transmitted by the module.
- Peripheral Chip Select Strobe (O)—Used only when the peripheral-chip-select strobe is enabled (MCR[PCSSE]). Strobes an off-module peripheral-chip-select demultiplexer, which decodes the module's PCS signals other than PCS5, preventing glitches on the demultiplexer outputs.

Slave mode: Unused

49.3.7 PCS6–PCS7

Master mode: Peripheral Chip Selects 6–7 (O)—Selects an SPI slave to receive data transmitted by the module

Slave mode: Unused

49.3.8 SCK

Master mode: Serial Clock (O)—Supplies a clock signal from the module to SPI slaves

Slave mode: Serial Clock (I)—Supplies a clock signal to the module from the SPI master

49.3.9 SIN

Master mode: Serial Input (I)—Receives serial data

Slave mode: Serial Input (I)—Receives serial data

49.3.10 SOUT

Master mode: Serial Output (O)—Transmits serial data

Slave mode: Serial Output (O)—Transmits serial data

49.4 SPI register descriptions

Register accesses to memory addresses that are reserved or undefined result in a transfer error. Any write access to the POPR and RXFR*n* also results in a transfer error.

NOTE

While the module is in the running state, do not write to:

- CTAREn

49.4.1 SPI memory map

SPI_0 base address: 401D_4000h

SPI_1 base address: 401D_8000h

SPI_2 base address: 401D_C000h

SPI_3 base address: 402C_8000h

SPI_4 base address: 402C_C000h

SPI_5 base address: 402D_0000h

Freescale Confidential Proprietary
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c2

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration Register (MCR)	32	RW	0000_4001h
8h	Transfer Count Register (TCR)	32	RW	0000_0000h
Ch - 20h	Clock and Transfer Attributes Register (in Master mode) (CTAR0 - CTAR5)	32	RW	7800_0000h
Ch	Clock and Transfer Attributes Register (in Slave mode) (CTAR0_SLAVE)	32	RW	7800_0000h
2Ch	Status Register (SR)	32	W1C	0201_0000h
30h	DMA/Interrupt Request Select and Enable Register (RSER)	32	RW	0000_0000h
34h	PUSH TX FIFO Register In Master Mode (PUSHR)	32	RW	0000_0000h
34h	PUSH TX FIFO Register In Slave Mode (PUSHR_SLAVE)	32	RW	0000_0000h
38h	POP RX FIFO Register (POPR)	32	RO	0000_0000h
3Ch - 4Ch	Transmit FIFO Registers (TXFR0 - TXFR4)	32	RO	0000_0000h
7Ch - 8Ch	Receive FIFO Registers (RXFR0 - RXFR4)	32	RO	0000_0000h
11Ch - 130h	Clock and Transfer Attributes Register Extended (CTARE0 - CTARE5)	32	RW	0000_0001h
13Ch	Status Register Extended (SREX)	32	RO	0000_0000h

49.4.2 Module Configuration Register (MCR)

Offset

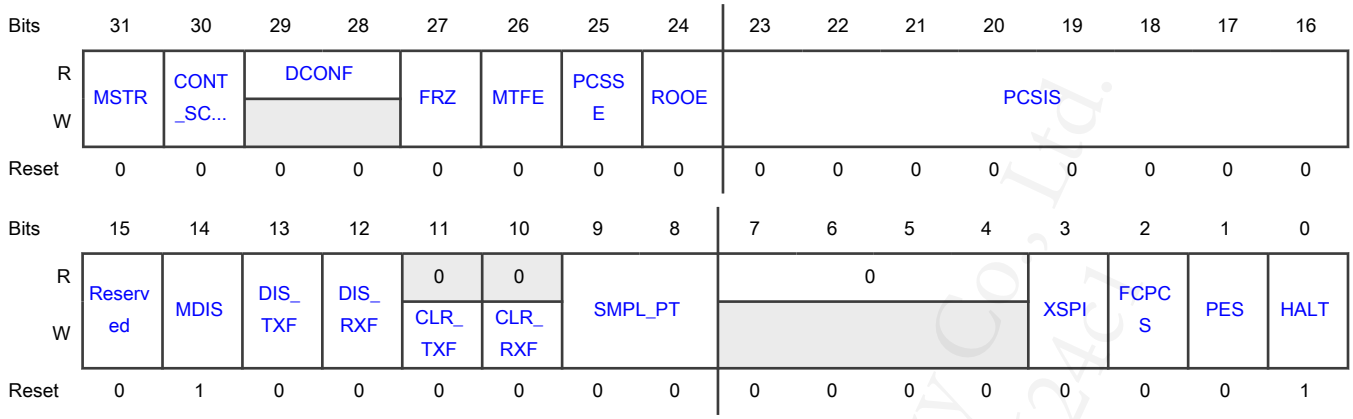
Register	Offset
MCR	0h

Function

This register includes fields to configure various attributes associated with module operations. The HALT and MDIS fields of this register can be changed any time, but the effect takes place only on the next frame boundary. Also, only these two fields can be changed when the module is in the Running state.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4a91-48a8-b251-5900ab624c1

Diagram



Fields

Field	Function
31 MSTR	Master/Slave Mode Select Enables either Master mode (if supported) or Slave mode (if supported) operation 0b - Enables Slave mode 1b - Enables Master mode
30 CONT_SCKE	Continuous SCK Enable Enables the SCK to run continuously 0b - Continuous SCK disabled 1b - Continuous SCK enabled
29-28 DCONF	SPI Configuration Selects among the different configurations of the module 00b - SPI 01b - Reserved 10b - Reserved 11b - Reserved
27 FRZ	Freeze Enables transfers to be stopped on the next frame boundary when the chip enters the Debug mode 0b - Do not halt serial transfers in Debug mode. 1b - Halt serial transfers in Debug mode.
26 MTFE	Modified Transfer Format Enable

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cc4625c4-821-4da8-b251-390a3ab624c1

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>Enables a modified transfer format to be used

NOTE
If the value of this field is 1, with continuous SCK enabled (MCR[CONT_SCKE] = 1) in Master mode, write 0 to CTAR[LSBFE] for correct operations while receiving unequal length frames. If PUSHR[CONT] is also set for back-to-back frame transfer, configure the frame size of the first frame as less than or equal to the frame size of the next frame. In this scenario, you must ensure that for all received frames, the bits are read equal to their respective frame sizes and any extra bits during the POP operation are masked.

0b - Modified SPI transfer format disabled
1b - Modified SPI transfer format enabled</td></tr>
<tr><td>25
PCSSE</td><td>Peripheral Chip Select Strobe Enable
Enables the PCS5/PCSS_b to operate as a PCS strobe output signal

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table><tr><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>SPI_0_MCR</td><td>—</td></tr><tr><td>—</td><td>SPI_1_MCR</td></tr><tr><td>—</td><td>SPI_2_MCR</td></tr><tr><td>—</td><td>SPI_3_MCR</td></tr><tr><td>—</td><td>SPI_4_MCR</td></tr><tr><td>—</td><td>SPI_5_MCR</td></tr></table>
0b - PCS5/PCSS_b is used as the PCS[5] signal.
1b - PCS5/PCSS_b is used as an active-low PCSS signal.</td></tr>
<tr><td>24
ROOE</td><td>Receive FIFO Overflow Overwrite Enable
In case of RX FIFO overflow, configure the module to ignore the incoming serial data or overwrite existing data. If the RX FIFO is full and new data is received, the data from the transfer, generating the overflow, is ignored or shifted to the Shift register.

0b - Incoming data is ignored.
1b - Incoming data is shifted to the Shift register.</td></tr>
<tr><td>23-16</td><td>Peripheral Chip Select x Inactive State</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>PCSIS</td><td>Determines the inactive state of PCSx when the module is in Master mode. This field has no effect when the module is in Slave mode. The Slave Select input to the module in Slave mode is always Active Low.

NOTE
The effect of this field only takes place when the module is enabled. Ensure that the field is configured correctly before enabling the SPI interface.

NOTE
This field is not supported in every instance. The following table includes only supported registers.

<table>
<tr><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SPI_0_MCR</td><td>—</td></tr>
<tr><td>SPI_1_MCR[20–16]</td><td>SPI_1_MCR[23–21]</td></tr>
<tr><td>SPI_2_MCR[20–16]</td><td>SPI_2_MCR[23–21]</td></tr>
<tr><td>SPI_3_MCR[20–16]</td><td>SPI_3_MCR[23–21]</td></tr>
<tr><td>SPI_4_MCR[20–16]</td><td>SPI_4_MCR[23–21]</td></tr>
<tr><td>SPI_5_MCR</td><td>—</td></tr>
</table>
00000000b - The inactive state of PCSx is low.
00000001b - The inactive state of PCSx is high.</td></tr>
<tr><td>15
—</td><td>Reserved</td></tr>
<tr><td>14
MDIS</td><td>Module Disable
Allows the clock to be stopped to the non-memory mapped logic in the module, effectively putting it in a software-controlled, power-saving state. The reset value of this field is parametrized with a default reset value of 1. When the module is used in the Slave mode, it is recommended to retain 0 as the value of this field.
0b - Enables the module clocks
1b - Allows external logic to disable the module clocks</td></tr>
<tr><td>13
DIS_TXF</td><td>Disable Transmit FIFO
When the TX FIFO is disabled, the transmit part of the module operates as a simplified, double-buffered SPI. This field can be written to only when the value of the MDIS field is 0.
0b - TX FIFO is enabled.
1b - TX FIFO is disabled.</td></tr>
</table>

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 DIS_RXF	Disable Receive FIFO When the RX FIFO is disabled, the receive part of the module operates as a simplified double-buffered SPI. This field can only be written when the value of the MDIS field is 0. 0b - RX FIFO is enabled. 1b - RX FIFO is disabled.
11 CLR_TXF	Clear TX FIFO Flushes the TX FIFO. Writing a 1 to CLR_TXF clears the TX FIFO counter. The value of this field is always read as 0. 0b - Do not clear the TX FIFO counter. 1b - Clear the TX FIFO counter.
10 CLR_RXF	Clear RX FIFO Flushes the RX FIFO. Writing a 1 to this field clears the RX counter. The value of this field is always read as 0. **NOTE** After every RX FIFO clear operation (MCR[CLR_RXF] = 0b1), followed by an RX FIFO overflow (SR[RFOF] = 0b1) scenario, you must immediately perform a single POP from the RX FIFO and discard the read data. The POP and discard operations must be completed before receiving a new incoming frame. 0b - Do not clear the RX FIFO counter. 1b - Clear the RX FIFO counter.
9-8 SMPL_PT	Sample Point Controls when the module master samples SIN in the modified transfer format. This field is valid only when the value of CTAR*n*[CPHA] is 0. 00b - Zero protocol clock cycles between SCK edge and SIN sample 01b - One protocol clock cycle between SCK edge and SIN sample 10b - Two protocol clock cycles between SCK edge and SIN sample 11b - Reserved
7-4 —	Reserved
3 XSPI	Extended SPI Mode Enables the usage of Command and Transfer Attribute Register Extended (CTARE) registers. These registers allow you to send up to 32-bit SPI frames. Command cycling is also enabled, which allows you to

Table continues on the next page...

Table continued from the previous page...

Field	Function
	send multiple data frames using a single command frame. When MCR[DIS_TXF] is asserted, the Extended SPI mode cannot be used to transmit SPI frames that are more than 16 bits in size. 0b - Normal SPI mode. Frame size can be up to 16 bits. Command cycling is not available in this mode. 1b - Extended SPI mode. Up to 32-bit SPI frames, along with command cycling, are enabled.
2 FCPCS	Fast Continuous PCS Mode Enables the masking of “After SCK (t_{ASC})” and “PCS to SCK (t_{CSC})” delays when operating in the Continuous PCS mode. This masking is not available if Continuous SCK mode is enabled. The individual delay masks are selected via PUSHR[MASC] and PUSHR[MCSC]. The firmware should select appropriate masks when providing continuous frames via the PUSHR. 0b - Normal or slow continuous PCS mode. Masking of delays is disabled. 1b - Fast continuous PCS mode. Delays are masked via control fields in the PUSHR.
1 PES	Parity Error Stop Controls SPI operation when a parity error is detected in a received SPI frame 0b - SPI frame transmission continues. 1b - SPI frame transmission stops.
0 HALT	Halt Starts and stops frame transfers. See Start and Stop of Module transfers for details. 0b - Starts transfers 1b - Stops transfers

49.4.3 Transfer Count Register (TCR)

Offset

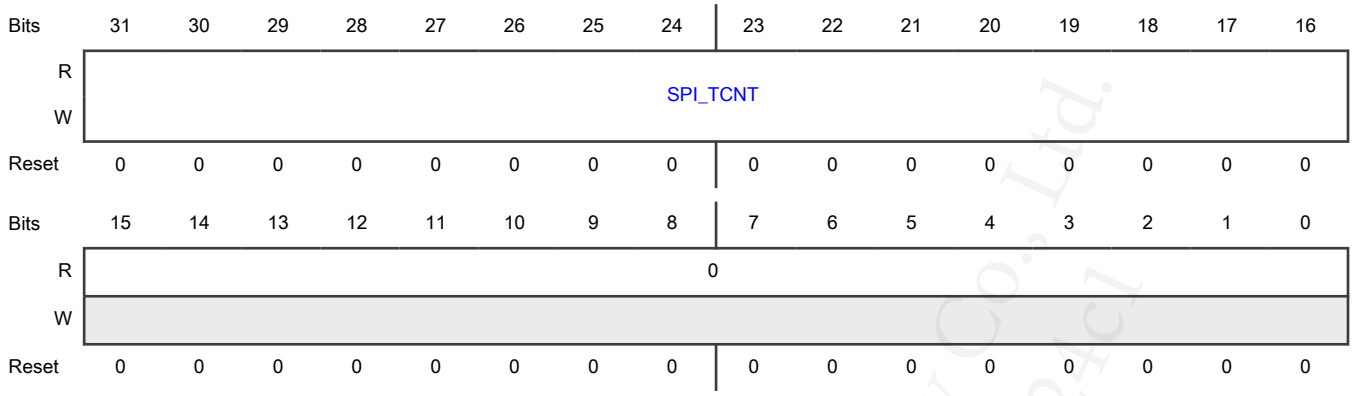
Register	Offset
TCR	8h

Function

This register contains a counter that indicates the number of SPI transfers made. The transfer counter is intended to assist in queue management. You must not write to the TCR when the module is in the running state.

Preliminary Information NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462f5-4821-48a8-b251-390d8bb24c

Diagram



Fields

Field	Function
31-16 SPI_TCNT	SPI Transfer Counter Counts the number of SPI transfers that the module performs This field increments every time the last bit of an SPI frame is transmitted, and a value written to this field presets the counter to that value. The field is reset to 0 at the beginning of a frame if the value of PUSHR[CTCNT] is 1. If the counter is not overwritten or cleared, it eventually overflows at value 65535 to 0.
15-0 —	Reserved

49.4.4 Clock and Transfer Attributes Register (in Master mode) (CTAR0 - CTAR5)

Offset

Register	Offset
CTAR0	Ch
CTAR1	10h
CTAR2	14h
CTAR3	18h
CTAR4	1Ch
CTAR5	20h

Function

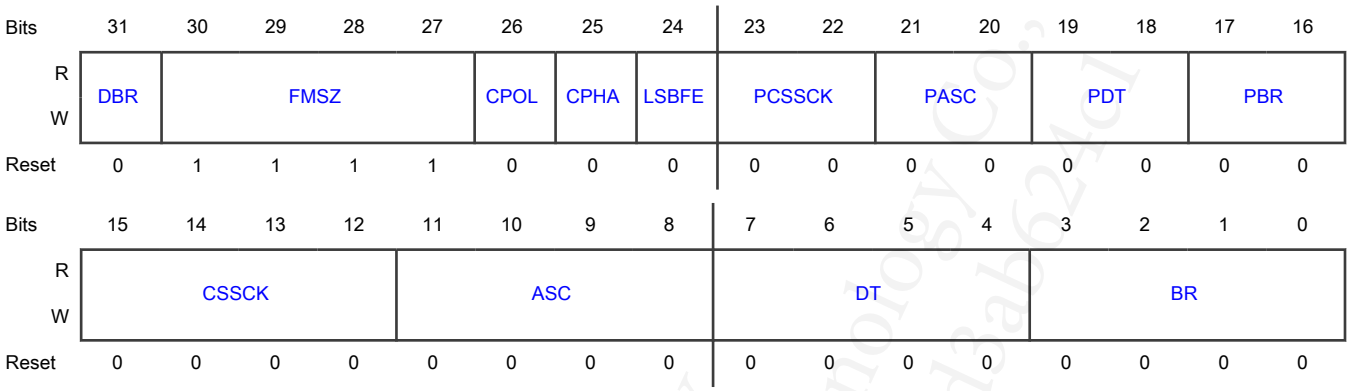
This set of registers is used to define different transfer attributes. You must not write to the CTARs when the module is in the running state.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8462e5-4821-48a2-b251-390d3ab624c1

In Master mode, the CTARs define combinations of transfer attributes such as frame size, clock phase and polarity, data bit ordering, baud rate, and various delays. In Slave mode, a subset of the fields in CTAR0 is used to configure the slave transfer attributes.

When the module is configured as an SPI master, the CTAS field in the command portion of the TX FIFO entry selects which of the CTARs is used. When the module is configured as an SPI bus slave, it uses the CTAR0 register.

Diagram



Fields

Field	Function
31 DBR	Double Baud Rate Doubles the effective baud rate of the Serial Communications Clock (SCK). This field is used only in Master mode. It effectively halves the baud rate division ratio, supporting faster frequencies, and odd division ratios for SCK. If the value of this field is 1, the duty cycle of the SCK depends on the value in the baud rate prescaler and the CPHA field, as listed in the following table. See the description of the VR field in this register for details on how to compute the baud rate.

Table 324. SPI SCK duty cycle

DBR	CPHA	PBR	SCK duty cycle
0	Any	Any	50/50
1	0	00	50/50
1	0	01	33/66
1	0	10	40/60
1	0	11	43/57
1	1	00	50/50
1	1	01	66/33
1	1	10	60/40
1	1	11	57/43

0b - The baud rate is computed normally with a 50/50 duty cycle.

1b - The baud rate is doubled with the duty cycle depending on the baud rate prescaler.

Table continued from the previous page...

Field	Function
30-27 FMSZ	Frame Size The number of bits transferred per frame is equal to the FMSZ value plus 1. Regardless of the transmission mode, the minimum valid frame size value is 4.
26 CPOL	Clock Polarity Selects the inactive state of the SCK. This field is used in both Master and Slave modes. For successful communication between serial devices, the devices must have identical clock polarities. When the continuous selection format is selected, switching between clock polarities without stopping the module can cause errors in the transfer because of the peripheral device interpreting the switch of clock polarity as a valid clock edge. **NOTE** In case of Continuous SCK mode, when the module goes in low-power mode (disabled), the inactive state of SCK is not guaranteed. 0b - The inactive state value of SCK is low. 1b - The inactive state value of SCK is high.
25 CPHA	Clock Phase Selects which edge of SCK causes data to change and which edge causes data to be captured. This field is used in both Master and Slave modes. For successful communication between serial devices, the devices must have identical clock phase settings. In Continuous SCK mode, the field value is ignored and the transfers are done considering 1 as the value of this field. 0b - Data is captured on the leading edge of SCK and changed on the following edge. 1b - Data is changed on the leading edge of SCK and captured on the following edge.
24 LSBFE	LSB First Specifies whether the LSB or MSB of the frame is transferred first. 0b - Data is transferred to MSB first. 1b - Data is transferred to LSB first.
23-22 PCSSCK	PCS to SCK Delay Prescaler Selects the prescaler value for the delay between assertion of PCS and the first edge of the SCK. See the CSSCK field description for information on how to compute the PCS to SCK delay. See PCS to SCK Delay (t_{CSC}) for details. 00b - PCS to SCK prescaler value is 1. 01b - PCS to SCK prescaler value is 3. 10b - PCS to SCK prescaler value is 5. 11b - PCS to SCK prescaler value is 7.
21-20 PASC	After SCK Delay Prescaler

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Selects the prescaler value for the delay between the last edge of SCK and the negation of PCS. See the ASC field description for information on how to compute the "After SCK Delay". See After SCK Delay (t_{ASC}) for details. 00b - Delay after Transfer Prescaler value is 1. 01b - Delay after Transfer Prescaler value is 3. 10b - Delay after Transfer Prescaler value is 5. 11b - Delay after Transfer Prescaler value is 7.
19-18 PDT	Delay after Transfer Prescaler Selects the prescaler value for the delay between the negation of the PCS signal at the end of a frame and the assertion of PCS at the beginning of the next frame. This field is only used in Master mode. See the DT field description for details on how to compute the "Delay after Transfer". See Delay after Transfer (t_{DT}) for details. 00b - Delay after Transfer Prescaler value is 1. 01b - Delay after Transfer Prescaler value is 3. 10b - Delay after Transfer Prescaler value is 5. 11b - Delay after Transfer Prescaler value is 7.
17-16 PBR	Baud Rate Prescaler Selects the prescaler value for the baud rate. This field is used only in Master mode. The baud rate is the frequency of the SCK. The protocol clock is divided by the prescaler value before the baud rate selection takes place. See the BR field description for details on how to compute the baud rate. 00b - Baud Rate Prescaler value is 2. 01b - Baud Rate Prescaler value is 3. 10b - Baud Rate Prescaler value is 5. 11b - Baud Rate Prescaler value is 7.
15-12 CSSCK	PCS to SCK Delay Scaler Selects the scaler value for the PCS to SCK delay. This field is used only in master mode. The PCS to SCK Delay is the delay between the assertion of PCS and the first edge of the SCK. The delay is a multiple of the protocol clock period, and it is computed according to the following equation: $t_{CSC} = (1/f_P) \times PCSSCK \times CSSCK$. The following table lists the delay scaler values.

Table 325. Delay scaler encoding

Field value	Delay scaler value
0000	2
0001	4
0010	8

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Table 325. Delay scaler encoding (continued)

Field value	Delay scaler value
0011	16
0100	32
0101	64
0110	128
0111	256
1000	512
1001	1024
1010	2048
1011	4096
1100	8192
1101	16384
1110	32768
1111	65536

	See PCS to SCK Delay (t_{CSC}) for details.
11-8 ASC	After SCK Delay Scaler Selects the scaler value for the "After SCK Delay". This field is used only in master mode. The "After SCK Delay" is the delay between the last edge of SCK and the negation of PCS. The delay is a multiple of the protocol clock period, and it is computed according to the following equation: $t_{ASC} = (1/f_P) \times PASC \times ASC$ See Delay Scaler Encoding table in CTARn[CSSCK] bit field description for scaler values. See After SCK Delay (t_{ASC}) for details.
7-4 DT	Delay After Transfer Scaler Selects the "Delay after Transfer" scaler. This field is used only in Master mode. The "Delay after Transfer" is the time between the negation of the PCS signal at the end of a frame and the assertion of PCS at the beginning of the next frame. In the continuous serial communications clock operation, the DT value is fixed to one SCK clock period. The "Delay after Transfer" is a multiple of the protocol clock period, and it is computed according to the following equation: $t_{DT} = (1/f_P) \times PDT \times DT$ See Delay Scaler Encoding table in the CTAR*n*[CSSCK] field description for scaler values.
3-0	Baud Rate Scaler

Table continues on the next page...

Table continued from the previous page...

Field	Function
BR	Selects the scaler value for the baud rate. This field is used only in master mode. The prescaled protocol clock is divided by the baud rate scaler to generate the frequency of the SCK. The baud rate is computed according to the following equation: SCK baud rate = (f_P/PBR) x [(1+DBR)/BR] The following table lists the baud rate scaler values.

Table 326. Baud rate scaler

CTARn[BR]	Baud rate scaler value
0000	2
0001	4
0010	6
0011	8
0100	16
0101	32
0110	64
0111	128
1000	256
1001	512
1010	1024
1011	2048
1100	4096
1101	8192
1110	16384
1111	32768

49.4.5 Clock and Transfer Attributes Register (in Slave mode) (CTAR0_SLAVE)

Offset

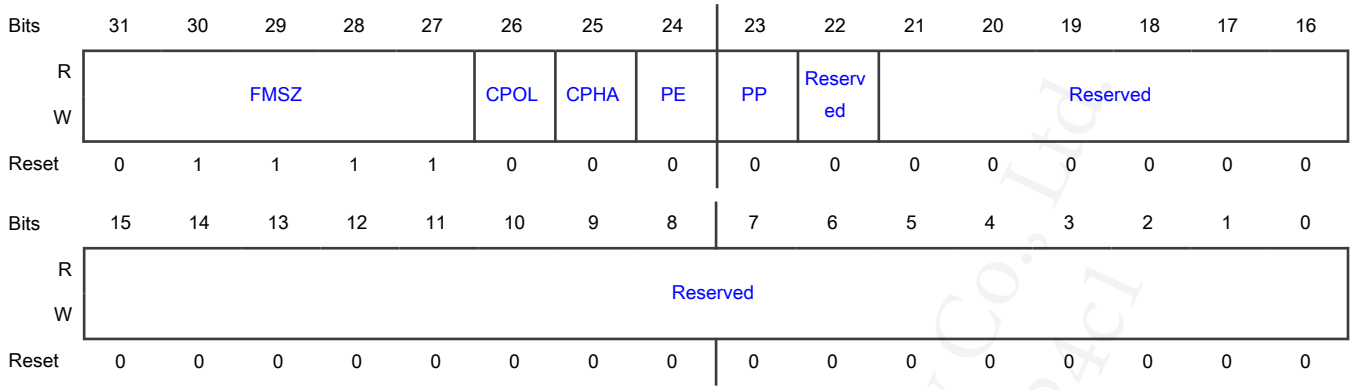
Register	Offset
CTAR0_SLAVE	Ch

Function

When the module is configured as an SPI bus slave, the CTAR0 register is used.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-27 FMSZ	Frame Size The number of bits transferred per frame is equal to the value of this field plus 1. Note that the minimum valid value of frame size is 4.
26 CPOL	Clock Polarity Selects the inactive state of the SCK. **NOTE** In case of Continuous SCK mode, when the module goes in low-power mode (disabled), the inactive state of SCK is not guaranteed. 0b - The inactive state value of SCK is low. 1b - The inactive state value of SCK is high.
25 CPHA	Clock Phase Selects which edge of SCK causes data to change and which edge causes data to be captured. This field is used in both Master and Slave modes. For successful communication between serial devices, the devices must have identical clock phase settings. In Continuous SCK mode, the field value is ignored and the transfers are done considering 1 as the value for this field. 0b - Data is captured on the leading edge of SCK and changed on the following edge. 1b - Data is changed on the leading edge of SCK and captured on the following edge.
24 PE	Parity Enable Enables parity bit transmission and reception for the frame 0b - No parity bit included/checked 1b - Parity bit is transmitted instead of last data bit in frame; and parity is checked for the received frame
23 PP	Parity Polarity Controls polarity of the parity bit transmitted and checked

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Even parity: the number of 1 bits in the transmitted frame is even. SR[SPEF] is set if the number of 1 bits is odd in the received frame. 1b - Odd parity: the number of 1 bits in the transmitted frame is odd. SR[SPEF] is set if the number of 1 bits is even in the received frame.
22 —	Reserved
21-0 —	Reserved

49.4.6 Status Register (SR)

Offset

Register	Offset
SR	2Ch

Function

This register includes status and flag fields, which reflect the status of the module and indicate the occurrence of events that can generate interrupts or DMA requests. The software can clear the fields of this register by writing a 1, and writing a 0 does not effect the fields. The register may not be writeable in the Module Disable mode because of the use of power-saving mechanisms.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TCF	TXRXS	0	EOQF	TFUF	0	TFFF	BSYF	CMDTCF	0	SPEF	0	RFOF	TFIWF	RFDF	CMDFF
W	W1C			W1C	W1C		W1C		W1C		W1C		W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXCTR				TXNXTPTR				RXCTR				POPNXTPTR			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Transfer Complete Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
TCF	Indicates that all bits in a frame are shifted out. This field remains set until it is cleared by writing a 1 to it. 0b - Transfer is incomplete. 1b - Transfer is complete.
30 TXRXS	TX and RX Status Reflects the run status of the module 0b - Transmit and receive operations are disabled when the module is in the Stopped state. 1b - Transmit and receive operations are enabled when the module is in the Running state.
29 —	Reserved
28 EOQF	End of Queue Flag Indicates that the last entry in a queue had been transmitted when the module was in Master mode. This field is set when the TX FIFO entry has the EOQ field set in the command halfword and the end of the transfer is reached. The field remains set until cleared by writing a 1 to it. When this field is set, the TXRXS field is automatically cleared. 0b - EOQ is not set in the executing command. 1b - EOQ is set in the executing SPI command.
27 TFUF	Transmit FIFO Underflow Flag Indicates an underflow condition in the TX FIFO. The transmit underflow condition is detected only for SPI blocks operating in Slave mode and SPI configuration. This field is set when the TX FIFO of the module operating in SPI Slave mode is empty and an external SPI master initiates a transfer. The field remains set until cleared by writing a 1 to it. 0b - No TX FIFO underflow occurred. 1b - TX FIFO underflow had occurred.
26 —	Reserved
25 TFFF	Transmit FIFO Fill Flag Indicates whether there is an available location to be filled in the FIFO. Either DMA or an interrupt can be used to add another entry to the FIFO. Note that this field is set if at least one location is free in the FIFO. The field can be cleared by writing a 1 to it or by acknowledgement from the DMA controller to the TX FIFO full request, when the TX FIFO is full. If FIFO is filled manually, and not by DMA, see Transmit FIFO fill interrupt or DMA request **NOTE** The reset value of this field is 0 when the module is disabled(MCR[MDIS]=1). 0b - TX FIFO is full.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - TX FIFO is not full.
24 BSYF	Busy Flag Indicates that the current command frame is being used for transmitting multiple data frames. This field is valid only when the value of MCR[XSPI] is 1, and is not set for the last data frame of a cyclic command transfer or when the value of CTARE[DTCP] is 1. See CMD FIFO buffering mechanism for details. 0b - No cyclic command transfer is in progress. 1b - Cyclic command transfer is in progress. Current data frame is not the last data frame for on-going cyclic command transfer.
23 CMDTCF	Command Transfer Complete Flag Indicates that the last data frame for the current cyclic command has been transmitted. Therefore, this field is set only for the last data frame of a cyclic command transfer or when the value of CTARE[DTCP] is 1. The field remains set until it is cleared by writing a 1 to it. 0b - Data transfer by current command is incomplete. 1b - Data transfer by current command is complete.
22 —	Reserved
21 SPEF	SPI Parity Error Flag Indicates that SPI frame with parity error had been received. The field remains set until it is cleared by writing a 1 to it. 0b - No parity error occurred. 1b - Parity error had occurred.
20 —	Reserved
19 RFOF	Receive FIFO Overflow Flag Indicates an overflow condition in the RX FIFO. The field is set when the RX FIFO and Shift register are full and a transfer is initiated. The field remains set until it is cleared by writing a 1 to it. 0b - No RX FIFO overflow occurred. 1b - RX FIFO overflow has occurred.
18 TFIWF	Transmit FIFO Invalid Write Flag Indicates data write on TX FIFO when CMD FIFO is empty. Without a command, the data entries present in TX FIFO are invalid. This field remains set until it is cleared by writing a 1 to it. 0b - No invalid data is present in TX FIFO. 1b - Invalid data is present in TX FIFO because CMD FIFO is empty.

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 RFDF	Receive FIFO Drain Flag Indicates whether there is an available location to be drained from the FIFO. Either DMA or an interrupt can be used to read from the FIFO. Note that this field is set if at least one location can be read from the FIFO. The field can be cleared by writing a 1 to it or by acknowledgement from the DMA controller when the RX FIFO is empty. 0b - RX FIFO is empty. 1b - RX FIFO is not empty.
16 CMDFFF	Command FIFO Fill Flag Indicates whether there is an available location to be filled in the FIFO. Either a DMA request or an interrupt indication can be used to add another entry to the FIFO. Note that this field is set if at least one location is free in the FIFO. The CMDFFF is cleared by writing a 1 to it or by acknowledgement from the DMA controller to the CMD FIFO full request. 0b - CMD FIFO is full. 1b - CMD FIFO is not full.
15-12 TXCTR	TX FIFO Counter Indicates the number of valid entries in the TX FIFO. This field increments every time the PUSHR is written and decrements every time SPI command is executed and the SPI data is transferred to the Shift register.
11-8 TXNXTPTR	Transmit Next Pointer Indicates which TX FIFO entry is transmitted during the next transfer. This field updates every time SPI data is transferred from the TX FIFO to the Shift register. **NOTE** If TX FIFO is cleared by writing 1 to MCR[CLR_TXF], the value of the TXNXTPTR field does not change immediately to 0. Only when the next transfer starts, the field reflects the latest value, TXNXTPTR = 1.
7-4 RXCTR	RX FIFO Counter Indicates the number of entries in the RX FIFO. This field decrements every time the POPR is read and increments every time data is transferred from the Shift register to the RX FIFO.
3-0 POPNXTPTR	Pop Next Pointer Contains a pointer to the RX FIFO entry to be returned when the POPR is read. This field is updated when the POPR is read.

49.4.7 DMA/Interrupt Request Select and Enable Register (RSER)

Offset

Register	Offset
RSER	30h

Function

This register controls DMA and interrupt requests. You must not write to the register when the module is in the Running state.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TCF_RE	CMDFF_...	Reserved	EOQF_RE	TFUF_RE	Reserved	TFFF_RE	TFFF_DI...	CMDTCF_...	Reserved	SPEF_RE	Reserved	RFOF_RE	TFIWF_RE	RFDF_RE	RFDF_DI...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDFF_...	Reserved	0													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 TCF_RE	Transmission Complete Request Enable Enables SR[TCF] to generate an interrupt request. 0b - TCF interrupt requests are disabled. 1b - TCF interrupt requests are enabled.
30 CMDFFF_RE	Command FIFO Fill Flag Request Enable Enables SR[CMDFFF] to generate a request. The CMDFFF_DIRS field of this register selects between generating an interrupt request or a DMA request. 0b - CMDFFF interrupts or DMA requests are disabled. 1b - CMDFFF interrupts or DMA requests are enabled.
29 —	Always write the reset value to this field.
28 EOQF_RE	Finished Request Enable Enables SR[EOQF] to generate an interrupt request 0b - EOQF interrupt requests are disabled. 1b - EOQF interrupt requests are enabled.
27 TFUF_RE	Transmit FIFO Underflow Request Enable Enables SR[TFUF] to generate an interrupt request. 0b - TFUF interrupt requests are disabled. 1b - TFUF interrupt requests are enabled.
26	Always write the reset value to this field.

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
25 TFFF_RE	Transmit FIFO Fill Request Enable Enables SR[TFFF] to generate a request. The TFFF_DIRS field of this register selects between generating an interrupt request or a DMA request. 0b - TFFF interrupts or DMA requests are disabled. 1b - TFFF interrupts or DMA requests are enabled.
24 TFFF_DIRS	Transmit FIFO Fill DMA or Interrupt Request Select Selects between generating a DMA request or an interrupt request. When SR[TFFF] and RSER[TFFF_RE] are set, this field selects between generating an interrupt request or a DMA request. 0b - TFFF flag generates interrupt requests. 1b - TFFF flag generates DMA requests.
23 CMDTCF_RE	Command Transmission Complete Request Enable Enables SR[CMDTCF] to generate an interrupt request 0b - CMDTCF interrupt requests are disabled. 1b - CMDTCF interrupt requests are enabled.
22 —	Always write the reset value to this field.
21 SPEF_RE	SPI Parity Error Request Enable Enables SR[SPEF] to generate an interrupt request 0b - SPEF interrupt requests are disabled. 1b - SPEF interrupt requests are enabled.
20 —	Always write the reset value to this field.
19 RFOF_RE	Receive FIFO Overflow Request Enable Enables SR[RFOF] to generate an interrupt request 0b - RFOF interrupt requests are disabled. 1b - RFOF interrupt requests are enabled.
18 TFIWF_RE	Transmit FIFO Invalid Write Request Enable Enables SR[TFIWF] to generate an interrupt request 0b - TFIWF interrupt requests are disabled. 1b - TFIWF interrupt requests are enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
17 RFDF_RE	Receive FIFO Drain Request Enable Enables SR[RFDF] to generate a request. The RFDF_DIRS field of this register selects between generating an interrupt request or a DMA request. 0b - RFDF interrupt or DMA requests are disabled. 1b - RFDF interrupt or DMA requests are enabled.
16 RFDF_DIRS	Receive FIFO Drain DMA or Interrupt Request Select Selects between generating a DMA request or an interrupt request. If SR[RFDF] is set, and the value of the RFDF_RE field in this register is 1, the RFDF_DIRS field selects between generating an interrupt request or a DMA request. 0b - Interrupt request 1b - DMA request
15 CMDFFF_DIRS	Command FIFO Fill DMA or Interrupt Request Select Selects between generating a DMA request or an interrupt request. If SR[CMDFFF] is set, and the value of the CMDFFF_RE field in this register is 1, the CMDFFF_DIRS field selects between generating an interrupt request or a DMA request. 0b - CMDFFF flag generates interrupt requests. 1b - CMDFFF flag generates DMA requests.
14 —	Always write the reset value to this field.
13-0 —	Reserved

49.4.8 PUSH TX FIFO Register In Master Mode (PUSHR)

Offset

Register	Offset
PUSHR	34h

Function

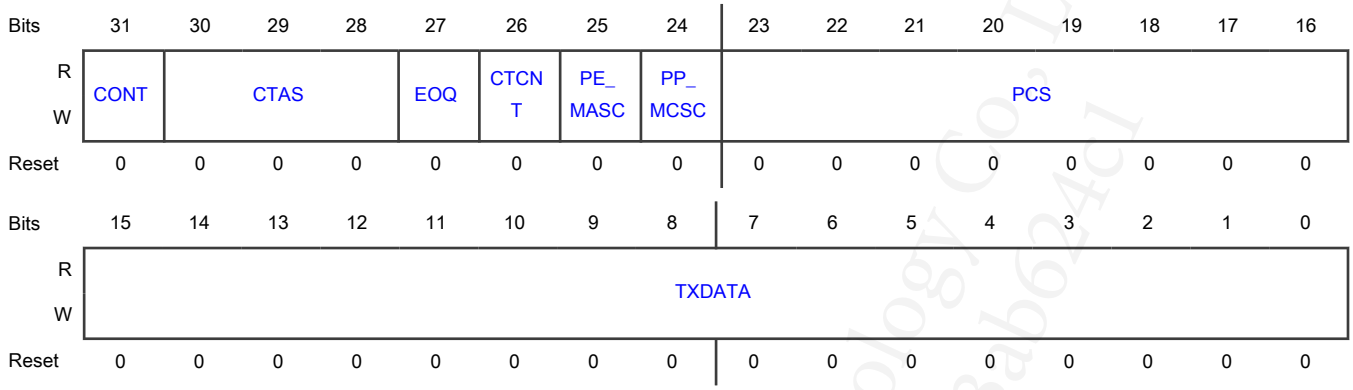
Specifies data to be transferred to the TX FIFO and CMD FIFO. You must write 16 bits of data into the TXDATA field. An 8- or 16-bit write access to the TXDATA field transfers 16 bits of data bus to the TX FIFO. A write access to the command fields transfers the 16 bits of command information to the CMD FIFO. In Master mode, the register transfers 16 bits of data to the TX FIFO and 16 bits of command information to the CMD FIFO.

If Extended SPI Mode is disabled (MCR[XSPI]), the TX FIFO and CMD FIFO must be filled simultaneously. In other words, you must perform write accesses to both the data and command fields for every PUSHR operation. With Extended SPI Mode disabled and both the TX FIFO and CMD FIFO written to and read from simultaneously, they behave as a single 32-bit FIFO. When Extended SPI mode is enabled (MCR[XSPI]), the TX FIFO and CMD FIFO can be written to independently.

A read access of PUSHR returns the top-most TX FIFO and CMD FIFO entries concatenated.

When the module is disabled, writing to this register does not update the FIFO. Therefore, any reads performed while the module is disabled return the last PUSHR write performed while the module was still enabled.

Diagram



Fields

Field	Function
31 CONT	Continuous Peripheral Chip Select Enable Selects a continuous selection format. This field is used in SPI Master mode. The field enables the selected PCS signals to remain asserted between transfers. 0b - Return PCSn signals to their inactive state between transfers. 1b - Keep PCSn signals asserted between transfers.
30-28 CTAS	Clock and Transfer Attributes Select Selects which CTAR to use in Master mode to specify the transfer attributes for the associated SPI frame. In SPI Slave mode, CTAR0 is used. See the chip-specific section for details to determine how many CTARs this chip has. You should not program a value in this field for a register that is not present. 000b - CTAR0 001b - CTAR1 010b - CTAR2 011b - CTAR3 100b - CTAR4 101b - CTAR5 110b - Reserved 111b - Reserved
27 EOQ	End of Queue The host software uses this field to signal to the module that the current SPI transfer is the last in a queue. At the end of the transfer, SR[EOQF] is set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - SPI data is not the last data to transfer. 1b - SPI data is the last data to transfer.
26 CTCNT	Clear Transfer Counter Writes a 0 to TCR[TCNT], and the TCNT field is cleared before the module starts transmitting the current SPI frame. 0b - Do not clear TCR[TCNT] 1b - Clear TCR[TCNT]
25 PE_MASC	Parity Enable or Mask TASC Delay in Current Frame PE – This bit enables parity bit transmission and parity reception check for the SPI frame. MASC - The current frame has the “after SCK” delay masked if this bit is asserted. See Fast Continuous Selection Format for more details. **NOTE** This field is used as Mask T_{ASC} in the Fast Continuous PCS mode when MCR[FCPCS] is set. 0b - PE - No parity bit is included/checked. MASC - T_{ASC} delay is not masked and the current frame has the "After SCK delay". 1b - PE - Parity bit is transmitted instead of the last data bit in the frame; parity is checked for the received frame. MASC - T_{ASC} delay is masked in the current frame.
24 PP_MCSC	Parity Polarity or Mask TCSC Delay in Next Frame PP: It controls the polarity of the parity bit transmitted and checked. MCSC: The next frame has the “PCS to SCK” delay masked if this bit is asserted. See Fast Continuous Selection Format for details. **NOTE** This field is used as Mask T_{CSC} in the Fast Continuous PCS mode when MCR[FCPCS] is set. 0b - PP - Even Parity: the number of 1 bits in the transmitted frame is even. SR[SPEF] is set if the number of 1 bits is odd in the received frame. MCSC - T_{CSC} delay is not masked and the next frame has the PCS to SCK delay. 1b - PP - Odd Parity: the number of 1 bits in the transmitted frame is odd. SR[SPEF] is set if the number of 1 bits is even in the received frame. MCSC - T_{ASC} delay is masked in the next frame.
23-16 PCS	PCS Selects which PCS signals are to be asserted for the transfer. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Table continues on the next page...

Table continued from the previous page...

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td></td><td>
<table>
<tr><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>SPI_0_PUSHR</td><td>—</td></tr>
<tr><td>SPI_1_PUSHR[20–16]</td><td>SPI_1_PUSHR[23–21]</td></tr>
<tr><td>SPI_2_PUSHR[20–16]</td><td>SPI_2_PUSHR[23–21]</td></tr>
<tr><td>SPI_3_PUSHR[20–16]</td><td>SPI_3_PUSHR[23–21]</td></tr>
<tr><td>SPI_4_PUSHR[20–16]</td><td>SPI_4_PUSHR[23–21]</td></tr>
<tr><td>SPI_5_PUSHR</td><td>—</td></tr>
</table>
00000000b - Negate the PCS[x] signal.
00000001b - Assert the PCS[x] signal.
</td></tr>
<tr><td>15-0
TXDATA</td><td>Transmit Data
Holds SPI data to be transferred according to the associated SPI command</td></tr>
</table>

49.4.9 PUSH TX FIFO Register In Slave Mode (PUSHR_SLAVE)

Offset

Register	Offset
PUSHR_SLAVE	34h

Function

Specifies the data to be transferred to the TX FIFO in Slave mode. An 8- or 16-bit write access to PUSHR transfers the 16-bit TXDATA field to the TX FIFO.

If the value of MCR[XSPI] is 1, up to 32-bit DSPI frames may be queued for transmission or reception.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 TXDATA	Transmit Data Holds SPI data to be transferred according to the associated SPI command

49.4.10 POP RX FIFO Register (POPR)

Offset

Register	Offset
POPR	38h

Function

This register is used to read the RX FIFO. An 8- or 16-bit access to the register has the same effect on the RX FIFO as 32-bit read accesses. A write to this register generates a transfer error.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca2467e5-5482-48a8-b251-390d3ab624c1

Fields

Field	Function
31-0 RXDATA	Received Data Contains the SPI data from the RX FIFO entry to which the pop next data pointer points

49.4.11 Transmit FIFO Registers (TXFR0 - TXFR4)

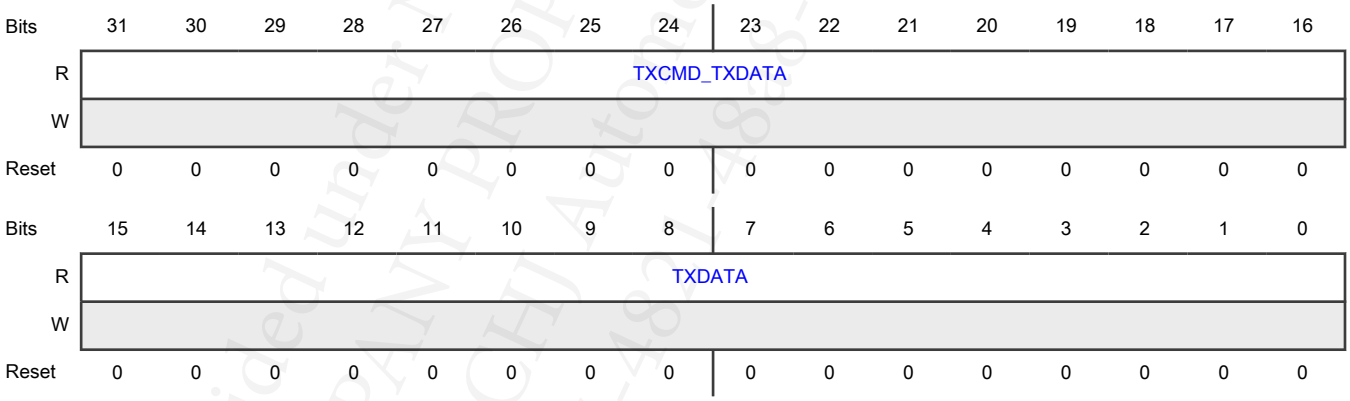
Offset

Register	Offset
TXFR0	3Ch
TXFR1	40h
TXFR2	44h
TXFR3	48h
TXFR4	4Ch

Function

These registers provide visibility into the TX FIFO for debugging purposes. Each register is an entry in the TX FIFO. The registers can only be read and not modified. Reading the TXFR*x* registers does not alter the state of the TX FIFO. Also, reading these registers is invalid if the module is operating in the Extended SPI mode.

Diagram



Fields

Field	Function
31-16 TXCMD_TXDATA	Transmit Command or Transmit Data In Master mode, the TXCMD field contains the command that sets the transfer attributes for the SPI data. In Slave mode, this field is reserved.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 TXDATA	Transmit Data Contains the SPI data to be shifted out.

49.4.12 Receive FIFO Registers (RXFR0 - RXFR4)

Offset

Register	Offset
RXFR0	7Ch
RXFR1	80h
RXFR2	84h
RXFR3	88h
RXFR4	8Ch

Function

These read-only registers provide visibility into the RX FIFO for debugging purposes. Each register is an entry in the RX FIFO. Reading the RXFR*x* registers does not alter the state of the RX FIFO. However, the value of MCR[MDIS] must be 0 when RXFR is read.

NOTE

Accessing a RXFR*n* register causes all subsequent SPI register accesses to be incorrect. Therefore, you must only read received data through SPI_POPR.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under IDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e54821e8a8-3251-390d3ab624c1

Fields

Field	Function
31-0 RXDATA	Receive Data Contains the received SPI data

49.4.13 Clock and Transfer Attributes Register Extended (CTARE0 - CTARE5)

Offset

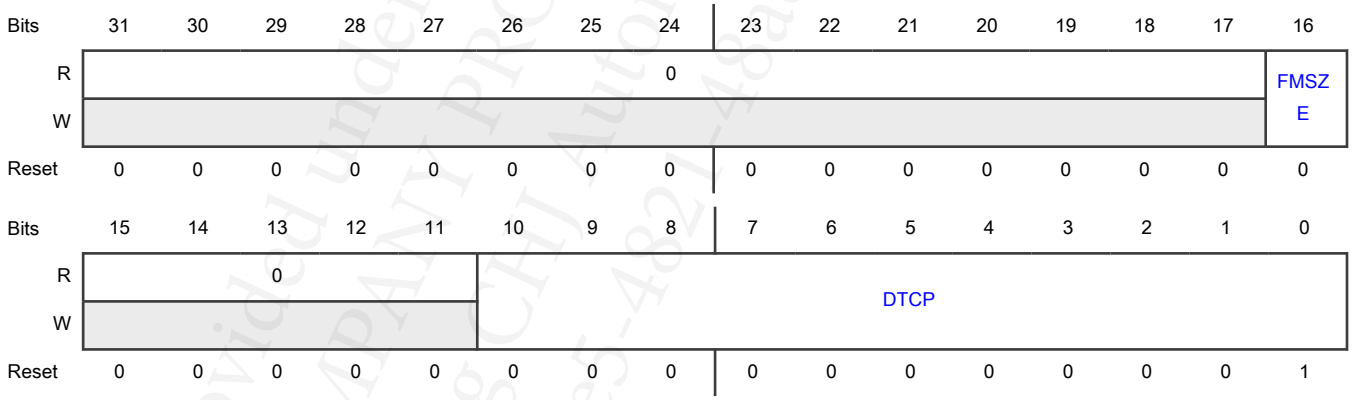
Register	Offset
CTARE0	11Ch
CTARE1	120h
CTARE2	124h
CTARE3	128h
CTARE4	12Ch
CTARE5	130h

Function

CTARE registers are used to define the extended transfer attributes for an SPI frame. These registers are valid only when Extended SPI mode is enabled (MCR[XSPI]).

When the module is configured as a SPI master, the CTAS field in CMD FIFO entry selects which of the CTARE registers is used.

Diagram



Fields

Field	Function
31-17	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4848-8251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
16 FMSZE	Frame Size Extended This field is valid only when MCR[XSPI] is set. This field concatenated with CTAR[FMSZ] defines the Frame size of the SPI frames to be transmitted. Effective frame size would be the concatenation of {CTARE[FMSZE], CTAR[FMSZ]} plus 1. 0b - Default Mode. Up to 16 bit SPI frames can be transfered. 1b - Up to 32 bit SPI frames can be transferred. Each Frame transfer will be a result of 2 simultaneous TX FIFO Pop Operation.
15-11 —	Reserved
10-0 DTCP	Data Transfer Count Preload This field is valid only when SPIx_MCR[XSPI] is set. This field defines the number of data frames (whose size is defined by CTARE[FMSZE] and CTAR[FMSZ]) to be transmitted using the Command frame which selected this SPIx_CTARE register. The value 0 is reserved and should not be written in this field. The default value of this field is 1.

49.4.14 Status Register Extended (SREX)

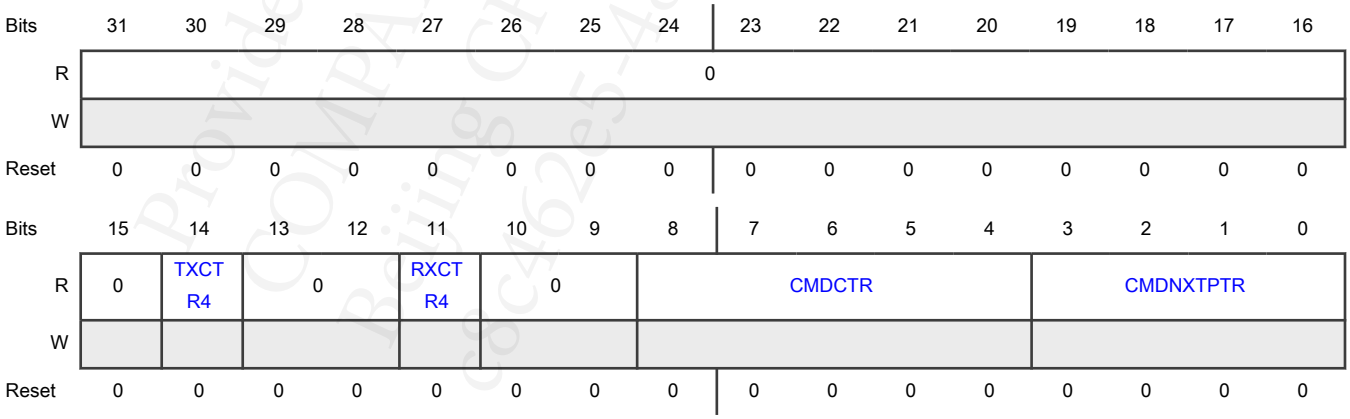
Offset

Register	Offset
SREX	13Ch

Function

The register contains status fields. The fields reflect the status of the module and indicate the occurrence of events. This register is not writeable.

Diagram



Fields

Field	Function
31-15 —	Reserved
14 TXCTR4	TX FIFO Counter[4] This bit is an extension of SR[TXCTR]. The concatenated field {TXCTR4, TXCTR} indicates the number of valid entries in the TX FIFO. This field is incremented every time the PUSHR is written. And this field is decremented every time an SPI command is executed and the SPI data is transferred to the Shift register.
13-12 —	Reserved
11 RXCTR4	RX FIFO Counter[4] This bit is an extension of SR[RXCTR]. The concatenated field {RXCTR4, RXCTR} indicates the number of entries in the RX FIFO. This field is decremented every time the POPR is read. And this field is incremented every time data is transferred from the Shift register to the RX FIFO.
10-9 —	Reserved
8-4 CMDCTR	CMD FIFO Counter Indicates the number of entries in the CMD FIFO. The CMDCTR is incremented every time the command part of PUSHR is written. The CMDCTR is decremented every time a SPI command is executed (all data frames due to current command frame have been transmitted).
3-0 CMDNXTPTR	Command Next Pointer Indicates which CMD FIFO Entry is used during the next transfer. The CMDNXTPTR field is updated every time SPI data due to current command have been transmitted.

49.5 Functional description

The module supports full-duplex, synchronous serial communications between chips and peripheral devices. The SPI configuration transfers data serially using a shift register and a selection of programmable transfer attributes.

The module includes the following configuration:

- SPI configuration in which the module operates as a basic SPI or a queued SPI

MCR[DCONF] determines the module configuration. SPI configuration is selected when the value of MCR[DCONF] is 0.

The CTAR*n* registers hold clock and transfer attributes. The SPI configuration allows to select which CTAR to use, on a frame-by-frame basis, by setting a field in the SPI command. MCR[XSPI] further allows the usage of CTARE*n* registers. This allows you to send multiple data frames using a single command frame.

Typical master-to-slave connections are shown in the following figure. When a data transfer operation is performed, data is serially shifted to a predetermined number of bit positions. Because the modules are linked, data is exchanged between the master and the slave. The data that was included in the master shift register is now in the shift register of the slave, and vice versa. At the end of a transfer, if the value of SR[TCF] is 1, it indicates a completed frame transfer.

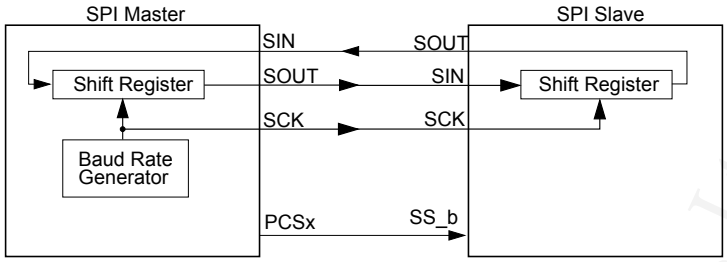


Figure 415. Serial protocol overview

Generally, more than one slave device can be connected to the module master, and 8 PCS signals of the module masters can be used to select which of the slaves to communicate with.

The SPI configuration shares transfer protocol and timing properties that are described independently of the configuration in Transfer formats. The transfer rate and delay settings are described in Module baud rate and clock delay generation.

49.5.1 Starting and stopping module transfers

The module has two operating states: Stopped and Running. Both the states are independent of their configuration. The default state of the module is Stopped. In this state, the module initiates no serial transfers in the Master mode and responds to no transfers in the Slave mode. The Stopped state is also a safe state for writing to the various configuration registers of the module, without causing undetermined results. In the Running state, serial transfers take place.

SR[TXRXS] indicates the state of the module. The value of this field is 1 if the module is in the Running state.

The module starts or transitions to the Running state when all of the following conditions are true:

- The value of SR[EOQF] = 0.
- The chip is not in the Debug mode or the value of MCR[FRZ] = 0.
- The value of MCR[HALT] = 0.

The module stops or transitions from the Running state to the Stopped state after the current frame, when any one of the following conditions exist:

- The value of SR[EOQF] = 1.
- The chip is in Debug mode and the value of MCR[FRZ] = 1.
- The value of MCR[HALT] = 1.

State transitions from running to stopped occur on the next frame boundary if a transfer is in progress, and occur immediately if no transfers are in progress.

49.5.2 SPI configuration

The SPI configuration transfers data serially using a shift register and a selection of programmable transfer attributes. The module is in SPI configuration when the value of MCR[DCONF] is 0. The SPI frames can be 32-bit long. The host CPU or a DMA controller transfers the external SPI data to the module RAM queues and to a TX FIFO buffer. The received data is stored in the entries of the RX FIFO buffer. The host CPU or the DMA controller transfers the received data from the RX FIFO to a memory external to the module. The operation of FIFO buffers is described in the following sections:

- TX FIFO buffering mechanism
- CMD FIFO buffering mechanism
- RX FIFO buffering mechanism

The interrupt and DMA request conditions are described in Interrupts/DMA requests.

The SPI configuration supports two module-specific modes—Master mode and Slave mode. In the Master mode, the module initiates and controls the transfer according to the fields of the executing SPI command. In the Slave mode, the module responds only to transfers initiated by a bus master external to it, and the SPI command field is reserved.

49.5.2.1 Master mode

In the SPI Master mode, the module initiates serial transfers by controlling the SCK and the PCS signals. The executing SPI command determines which CTARs will be used to set the transfer attributes and which PCS signals to assert. The command field also contains various bits that help with queue management and transfer protocol. See PUSH TX FIFO Register In Master Mode (PUSHR) for details on the SPI command fields. The data in the executing TX FIFO entry is loaded into the shift register and shifted out on the SOUT pin. In the SPI Master mode, each SPI frame to be transmitted has a command associated with it, allowing for transfer attribute control on a frame by frame basis. In the Extended SPI Master mode, multiple SPI frames can have a single command associated with them allowing efficient SPI frame transfers requiring common transfer attributes. In addition, the Extended SPI mode allows larger frame sizes of up to 32 bits.

49.5.2.2 Slave mode

In the SPI Slave mode, the module responds to transfers that an SPI bus master initiates. It does not initiate transfers. Certain transfer attributes such as clock polarity, clock phase, and frame size must be configured for successful communication with SPI master. The SPI Slave mode transfer attributes are configured in the CTAR0 and CTARE0 registers. The data shifts out with MSB first, and the module does not support shifting out of LSB.

49.5.2.3 FIFO disable operation

The FIFO disable mechanisms allow SPI transfers without using the TX FIFO, CMD FIFO, or RX FIFO. The module operates as a double-buffered, simplified SPI when the FIFOs are disabled. The transmit and receive sides of the FIFOs are disabled separately. Writing a 1 to MCR[DIS_TXF] disables the TX FIFO and CMD FIFO, and writing a 1 to MCR[DIS_RXF] disables the RX FIFO.

The FIFO disable mechanisms are transparent to you and to the host software. Transmit data and commands are written to the PUSHR and received data is read from the POPR.

When the TX FIFO and CMD FIFO are disabled:

- SR[TFFF], SR[TFUF], SR[CMDFFF], SREX[CMDCTR], and SR[TXCTR] behave as if there is a one-entry FIFO.
- The contents of TXFRs, SR[TXNXTPTR], and SREX[CMDNXTPTR] are undefined.

Similarly, when the RX FIFO is disabled, SR[RFDF], SR[RFOF], and SR[RXCTR] behave as if there is a one-entry FIFO, but the contents of the RXFR registers and SR[POPNXTPTR] are undefined.

49.5.2.4 TX FIFO buffering mechanism

The TX FIFO functions as a buffer of SPI data for transmission. It holds 5 words, each consisting of SPI data. The number of entries in the TX FIFO is chip-specific. SPI data is added to the TX FIFO by writing to the data field of PUSHR. TX FIFO entries can only be removed from the TX FIFO if they are shifted out or by flushing the TX FIFO.

SR[TXCTR] indicates the number of valid entries in the TX FIFO. The field updates every time an 8- or 16-bit write takes place on PUSHR[TXDATA] or SPI data is transferred into the Shift register from the TX FIFO.

SR[TXNXTPTR] indicates the TX FIFO entry that will be transmitted during the next transfer. The TXFR*n* registers are invalid in the Extended SPI mode because the TX FIFO and CMD FIFO can be used independently. The TXNXTPTR field increments every time SPI data is transferred from the TX FIFO to the Shift register. The maximum value of this field is equal to the maximum implemented TXFR number and it rolls over after reaching the maximum.

49.5.2.4.1 Filling the TX FIFO

Host software or other intelligent blocks can add (push) entries to the TX FIFO and CMD FIFO by writing to the PUSHR. When the TX FIFO is not full, the value of DR[TFFF] is set to 1, and the value is set to 0 when TX FIFO is full and the DMA controller indicates that a write to PUSHR is complete. You can also write a 1 to clear the field, which can generate a DMA request or an interrupt request. See Transmit FIFO fill interrupt or DMA request for details.

The module ignores attempts to push data to a full TX FIFO, and the state of the TX FIFO does not change and no error condition occurs.

49.5.2.4.2 Draining the TX FIFO

The module removes (drains) the TX FIFO entries by shifting SPI data out through the Shift register. It transfers entries from the TX FIFO to the Shift register, and the entries are shifted out as long as there are valid entries in the TX FIFO. Every time an entry is transferred from the TX FIFO to the Shift register, the TX FIFO counter decrements by 1. When the value of MCR[XSPI] is 1, and if the frame size of SPI data to be transmitted is more than 16 bits, then it causes two data entries to be popped from TX FIFO simultaneously. These entries are transferred to the Shift register. The first of the two popped entries forms the 16 least significant bits of the SPI frame to be transmitted. Such an operation also causes the TX FIFO counter to decrement by 2. At the end of a transfer, the value of SR[TCF] is set to indicate the completion of a transfer. The TX FIFO is flushed by writing a 1 to MCR[CLR_TXF].

If an external bus master initiates a transfer with a module slave while the slave's TX FIFO is empty, the value of SR[TFUF] is configured as 1. See Transmit FIFO underflow interrupt request for details.

49.5.2.5 CMD FIFO buffering mechanism

The CMD FIFO functions as a buffer of SPI command used for SPI data transmission. When Extended SPI Mode (MCR[XSPI]) is disabled, the TX FIFO and CMD FIFO must be filled together, i.e. write enables should be given for both the Data and Command fields while performing a PUSHR operation.When Extended SPI Mode (MCR[XSPI]) is enabled, the TX FIFO and CMD FIFO can be filled independently.

The CMD FIFO holds 5 words, each representing SPI command fields. The number of entries in the CMD FIFO is chip-specific. SPI command is added to the CMD FIFO by writing to the command field of PUSHR. CMD FIFO entries can only be removed from the CMD FIFO by being shifted out (to help transmit SPI data) or by flushing the CMD FIFO.

When Extended SPI Mode (MCR[XSPI]) is disabled, every CMD FIFO entry has a corresponding single TX FIFO entry attached to it because both these FIFO's are filled simultaneously.

When Extended SPI Mode (MCR[XSPI]) is enabled, every CMD FIFO entry can have multiple TX FIFO entries attached to it. Thus a single CMD FIFO entry can be used to transmit multiple TX FIFO entries. The CTARE[DTCP] field decides the number of SPI Data Frames having frame size as {FMSZE, FMSZ} to be transmitted using the current Command Entry. The CTAR/CTARE registers pointed by the CTAS field in the Command frame gives the FMSZ and FMSZE fields respectively. The time for which a command entry is in use is known as a Command Cycle. The Busy Flag SR[BSYF] is asserted for the duration of the Command Cycle except for the last SPI frame in the Command Cycle.

The CMD FIFO Counter field (CMDCTR) in the SPI Status Register extended(SREX) indicates the number of valid entries in the CMD FIFO. The CMDCTR field is updated every time a 8- or 16-bit write takes place on the lower half of SPI_PUSHR or SPI data is transferred into the shift register from the TX FIFO.

The TXFRn Registers are invalid in the Extended SPI Mode, since the TX FIFO and CMD FIFO can be used independently. The CMDNXTPTR field indicates which CMD FIFO Entry will be used during the next command cycle. The CMDNXTPTR field is incremented every time the last SPI data in the command cycle is transferred from the TX FIFO to the shift register and it rolls over after reaching the maximum.

49.5.2.6 RX FIFO buffering mechanism

The RX FIFO functions as a buffer for data received on the SIN pin. The FIFO holds 5 received SPI data frames. The number of entries in the RX FIFO is chip-specific. SPI data is added to the RX FIFO at the completion of a transfer when the received data in the Shift register is transferred into the RX FIFO. SPI data is removed (popped) from the RX FIFO by reading the POPR. RX FIFO entries can only be removed from the RX FIFO by reading the POPR or by flushing the RX FIFO.

SR[RXCTR] indicates the number of valid entries in the RX FIFO. The field is updated every time the POPR is read or SPI data is copied from the Shift register to the RX FIFO.

SR[POPNXTPTR] points to the RX FIFO entry that is returned when the POPR is read. The field contains the positive offset from RXFR0 in a number of 32-bit registers. For example, POPNXTPTR = 2 means that the RXFR2 contains the received SPI data that

is returned after the POPR is read. The POPNXTPTR field increments every time the POPR is read. The maximum value of the field is equal to the maximum implemented RXFR number and it rolls over after reaching the maximum.

49.5.2.6.1 Filling the RX FIFO

The RX FIFO is filled with the received SPI data from the Shift register. While the RX FIFO is not full, SPI frames from the Shift register are transferred to the RX FIFO. Every time a SPI frame is transferred to the RX FIFO, the RX FIFO counter is incremented by 1.

If the RX FIFO and Shift register are full and a transfer is initiated, the value of SR[RFOF] equals 1, indicating an overflow condition. Depending on the state of MCR[ROOE], the data from the transfer that generated the overflow is either ignored or shifted to the Shift register. If the value of the ROOE field is 1, the incoming data is shifted in to the Shift register. If the value of ROOE is 0, the incoming data is ignored.

49.5.2.6.2 Draining the RX FIFO

Host CPU or a DMA can remove (pop) entries from the RX FIFO by reading the POPR. A read of the POPR decrements the RX FIFO counter by 1. Attempts to pop data from an empty RX FIFO are ignored and the RX FIFO counter remains unchanged. The data, read from the empty RX FIFO, is undetermined.

When the RX FIFO is not empty, the value of SR[RFDF] is 1. The value of the RFDF field turns 0 when the RX FIFO is empty and the DMA controller indicates that a read from POPR is complete or by writing a 1 to it.

49.5.3 Module baud rate and clock delay generation

The SCK frequency and the delay values for serial transfer are generated by dividing the system clock frequency by a prescaler and a scaler with the option for doubling the baud rate. The following figure shows conceptually how the SCK signal is generated.



Figure 416. Communications clock prescalers and scalers

49.5.3.1 Baud rate generator

The baud rate is the frequency of the SCK. The protocol clock is divided by a prescaler (PBR) and scaler (BR) to produce SCK with the possibility of halving the scaler division. The DBR, PBR, and BR fields in the CTARs select the frequency of SCK by the formula in the BR field description. The following table shows an example of how to compute the baud rate.

Table 327. Baud rate computation example

f_P	PBR	Prescaler	BR	Scaler	DBR	Baud rate
100 MHz	0b00	2	0b0000	2	0	25 Mb/s
20 MHz	0b00	2	0b0000	2	1	10 Mb/s

NOTE

The clock frequencies mentioned in the preceding table are provided as an example. See the Clocking chapter for the frequency used to drive this module in the chip.

49.5.3.2 PCS to SCK delay (t_{CSC})

The "PCS to SCK delay" is the duration from assertion of the PCS signal to the first SCK edge. See Figure 418 for an illustration of the PCS to SCK delay. The PCSSCK and CSSCK fields in the CTAR*x* registers select the PCS to SCK delay by the formula in the CSSCK field description. The following table shows an example of how to compute the PCS to SCK delay.

Table 328. PCS to SCK delay computation example

f_{SYS}	PCSSCK	Prescaler	CSSCK	Scaler	PCS to SCK Delay
100 MHz	0b01	3	0b0100	32	0.96 µs

NOTE

The clock frequency mentioned in the preceding table is given as an example. See the Clocking chapter for the frequency used to drive this module in the chip.

49.5.3.3 After SCK delay (t_{ASC})

The "After SCK Delay" is the duration between the last edge of SCK and the negation of PCS. See Figure 418 and Figure 419 for illustrations of this type of delay. The PASC and ASC fields in the CTAR*x* registers select the "After SCK Delay" by the formula in the ASC field description. The following table shows an example of how to compute the "After SCK Delay".

Table 329. After SCK Delay computation example

f_P	PASC	Prescaler	ASC	Scaler	After SCK Delay
100 MHz	0b01	3	0b0100	32	0.96 µs

NOTE

The clock frequency mentioned in the preceding table is given as an example. See the Clocking chapter for the frequency used to drive this module in the chip.

49.5.3.4 Delay after transfer (t_{DT})

The "Delay after Transfer" is the minimum time between negation of the PCS signal for a frame and the assertion of the PCS signal for the next frame. See Figure 418 for an illustration of this type of delay. The PDT and DT fields in the CTAR*x* registers select the "Delay after Transfer" by the formula in the DT field description. The following table shows an example of how to compute the "Delay after Transfer".

Table 330. "Delay after Transfer" computation example

f_P	PDT	Prescaler	DT	Scaler	Delay after Transfer
100 MHz	0b01	3	0b1110	32768	0.98 ms

NOTE

The clock frequency mentioned in the preceding table is given as an example. See the Clocking chapter for the frequency used to drive this module in the chip.

When in Non-Continuous Clock mode, the t_{DT} delay is configured according to the equation specified in the CTAR[DT] field description. When in Continuous Clock mode, the delay is fixed at 1 SCK period.

49.5.3.5 PCSS enable (PCSS_b)

The PCSS_b signal provides a delay to allow the PCS signals to settle after a transition occurs, thereby avoiding glitches. When the module is in the Master mode and the value of MCR[PCSSE] is 1, PCSS_b provides a signal for an external demultiplexer to decode peripheral chip selects other than PCS5 into glitch-free PCS signals. The following figure shows the timing of the PCSS_b signal relative to PCS signals.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CUA Automotive Technology Co., Ltd.
c8622e4a91484a8a9351-90a3ab04c1

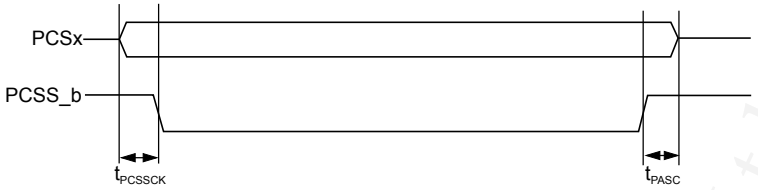


Figure 417. PCSS timing

CTAR[PCSSCK] selects the delay between the assertion of the PCS signals and the assertion of PCSS_b based on the following formula:

$$t_{\mathrm{PCSSCK}} = \frac{1}{f_{\mathrm{P}}} \times \mathrm{PCSSCK}$$

At the end of the transfer, CTAR[PASC] selects the delay between PCSS_b negation and PCS negation based on the following formula:

$$t_{\mathrm{PASC}} = \frac{1}{f_{\mathrm{P}}} \times \mathrm{PASC}$$

The following table shows an example of how to compute the t_{pcssck} delay.

Table 331. PCSS assert computation example

f_P	PCSSCK	Prescaler	Delay before transfer
100 MHz	0b11	7	70.0 ns

The following table shows an example of how to compute the t_{pasc} delay.

Table 332. PCSS negate computation example

f_P	PASC	Prescaler	Delay after transfer
100 MHz	0b11	7	70.0 ns

The PCSS_b signal is not supported when the Continuous SCK mode is enabled.

NOTE

The clock frequency mentioned in the tables above is an example. See the Clocking chapter for details on the frequency used to drive this module in the chip.

49.5.4 Transfer formats

The SPI serial communication is controlled by the SCK signal and the PCS signals. The SCK signal provided by the master device synchronizes shifting and sampling of the data on the SIN and SOUT pins. The PCS signals serve as enable signals for the slave devices.

In Master mode, CTAR*n*[CPOL] and CTAR*n*[CPHA] select the polarity and phase of the serial clock, SCK.

- CPOL - Selects the idle state polarity of the SCK
- CPHA - Selects if the data on SOUT is valid before or on the first SCK edge

Even if the bus slave does not control the SCK signal, in Slave mode, the values of CPOL and CPHA must be identical to the master device settings to ensure proper transmission. In SPI Slave mode, only CTAR0 is used.

The module supports four different transfer formats:

- Classic SPI with CPHA = 0
- Classic SPI with CPHA = 1
- Modified Transfer Format with CPHA = 0
- Modified Transfer Format with CPHA = 1

A modified transfer format is supported to allow for high-speed communication with peripherals that require longer setup times. The module can sample the incoming data later than halfway through the cycle to give the peripheral more setup time. The MCR[MTFE] selects between the classic SPI format and modified transfer format.

In the interface configurations, the module provides the option of having the PCS signals asserted between frames. See Continuous selection format for details.

49.5.4.1 Classic SPI transfer format (CPHA = 0)

The transfer format shown in the following figure is used to communicate with peripheral SPI slave devices where the first data bit is available on the first clock edge. In this format, the master and slave sample their SIN pins on the odd-numbered SCK edges and change the data on their SOUT pins on the even-numbered SCK edges.

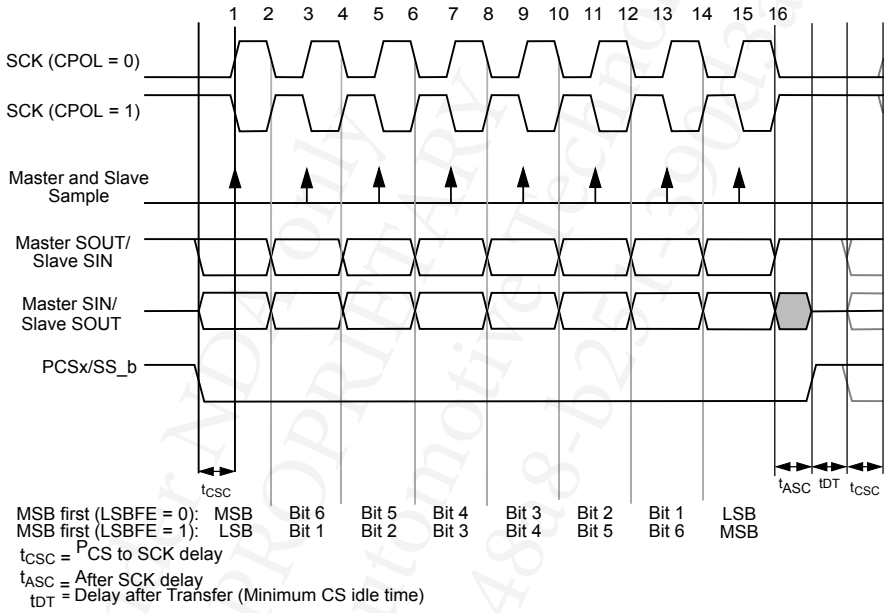


Figure 418. Module transfer timing diagram (MTFE = 0, CPHA = 0, FMSZ = 8)

The master initiates the transfer by placing its first data bit on the SOUT pin and asserting the appropriate PCS signals to the slave device. The slave responds by placing its first data bit on its SOUT pin. After the t_{CSC} delay elapses, the master outputs the first edge of SCK. The master and slave devices use this edge to sample the first input data bit on their serial data input signals. At the second edge of the SCK, the master and slave devices place their second data bit on their serial data output signals. For the rest of the frame, the master and the slave sample their SIN pins on the odd-numbered clock edges and change the data on their SOUT pins for the even-numbered clock edges. After the last clock edge occurs, a delay of t_{ASC} is inserted before the master negates the PCS signals. A delay of t_{DT} is inserted before a new frame transfer can be initiated by the master.

49.5.4.2 Classic SPI transfer format (CPHA = 1)

This transfer format shown in the following figure is used to communicate with peripheral SPI slave devices that require the first SCK edge before the first data bit becomes available on the slave SOUT pin. In this format, the master and slave devices change the data on their SOUT pins on the odd-numbered SCK edges and sample the data on their SIN pins for the even-numbered SCK edges.

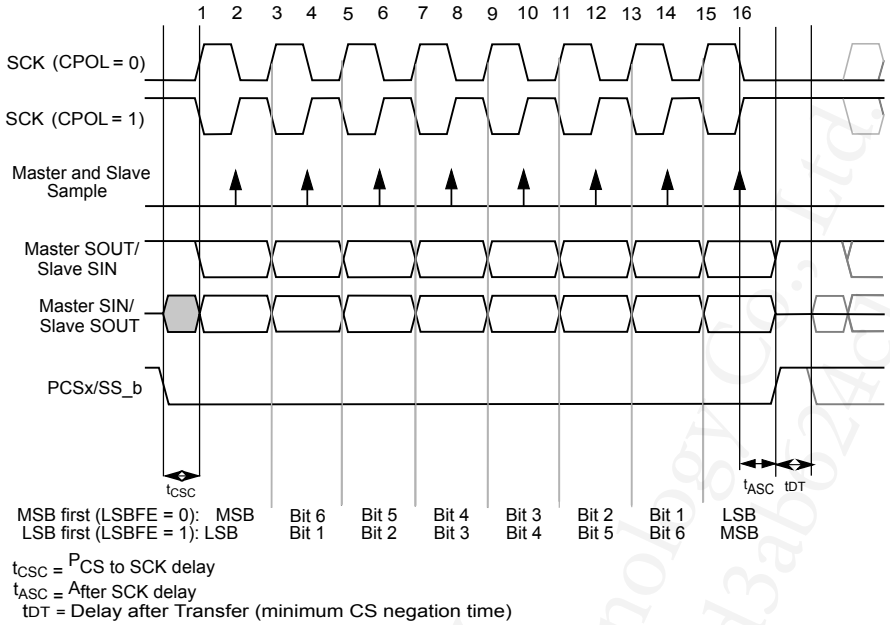


Figure 419. Module transfer timing diagram (MTFE = 0, CPHA = 1, FMSZ = 8)

The master initiates the transfer by asserting the PCS signal to the slave. After the t_{CSC} delay elapses, the master generates the first SCK edge and at the same time places valid data on the master SOUT pin. The slave responds to the first SCK edge by placing its first data bit on its slave SOUT pin.

At the second edge of the SCK, the master and slave sample their SIN pins. For the rest of the frame, the master and the slave change the data for their SOUT pins on the odd-numbered clock edges and sample their SIN pins on the even-numbered clock edges. After the last clock edge occurs, a delay of t_{ASC} is inserted before the master negates the PCS signal. A delay of t_{DT} is inserted before the master can initiate a new frame transfer.

49.5.4.3 Modified SPI transfer format (MTFE = 1, CPHA = 0)

In the modified transfer format, both the master and the slave sample later in the SCK period than in Classic SPI mode to allow the logic to tolerate more delays in device pads and board traces. These delays become a more significant fraction of the SCK period as the SCK period decreases with increasing baud rates.

The master and the slave place data on the SOUT pins at the assertion of the PCS signal. After the PCS to SCK delay elapses, the first SCK edge is generated. The slave samples the master SOUT signal on every odd-numbered SCK edge. The SPI in the Slave mode, when the value of MTFE is 1, also places new data on the slave SOUT on every odd-numbered clock edge. Regular external slave, configured with CPHA=0 format, drives its SOUT output at every even-numbered SCK clock edge.

The SPI master places its second data bit on the SOUT line, one protocol clock after the odd-numbered SCK edge, if the protocol clock frequency to SCK frequency ratio is higher than 3. If this ratio is below 4, the master changes SOUT at the even-numbered SCK edge. MCR[SMPL_PT] selects the point where the master samples the SIN. The master sample point can be delayed by one or two protocol clock cycles. The value of SMPL_PT must be 0 if the protocol to SCK frequency ratio is less than 4. However if this ratio is less than 4, the actual sample point is delayed by one protocol clock cycle automatically by the design.

The timing delays are as follows:

- T_{csc} - PCS to SCK assertion delay
- T_{acs} - After SCK PCS negation delay
- T_{su_ms} - master SIN setup time
- T_{hd_ms} - master SIN hold time
- T_{vd_sl} - slave data output valid time, time between slave data output SCK driving edge and when data becomes valid

- T_{su_sl} - data setup time on slave data input
- T_{hd_sl} - data hold time on slave data input
- T_{sys} - protocol clock period

The following timing diagrams illustrate the SPI operation with MCR[MTFE] = 1 and show the modified transfer format for CPHA = 0 and Fsys/Fsck = 4. Only the condition where CPOL = 0 is illustrated. Solid triangles show the data sampling clock edges, and here are the two possible slave behaviors:

- Signal, marked "SOUT of Ext Slave" presents regular SPI slave serial output.
- Signal, marked "SOUT of SPI Slave" presents SPI in the slave mode with MTFE = 1.

Other MTFE = 1, diagrams show SPI SIN input as being driven by a regular external SPI slave, configured according to SPI master CPHA programming.

NOTE

In the following diagrams, f_{sys} represents the protocol clock frequency from which the baud frequency f_{sck} is derived.

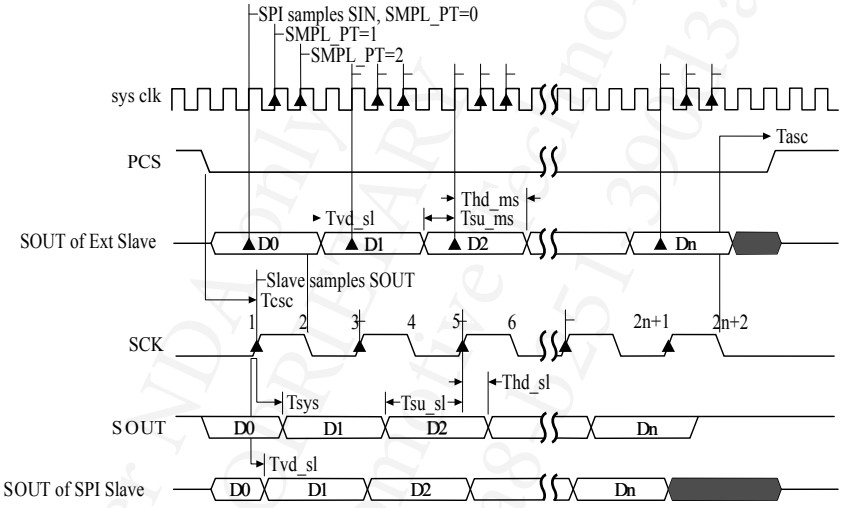


Figure 420. SPI modified transfer format (MTFE=1, CPHA=0, $f_{sck} = f_{sys}/4$)

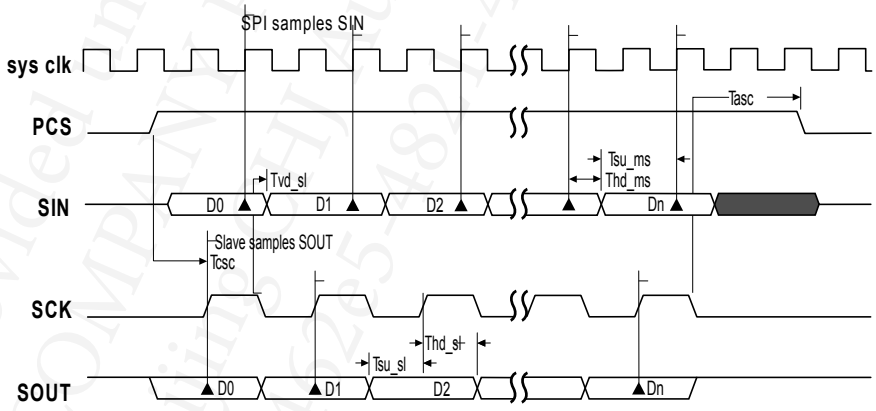


Figure 421. SPI modified transfer format (MTFE=1, CPHA=0, $f_{sck} = f_{sys}/2$)

Provided under NDA ONLY
COMPANY PROPRIETARY
Bright Power Semiconductor Co., Ltd.
c8042554182a8251390f3ab0241

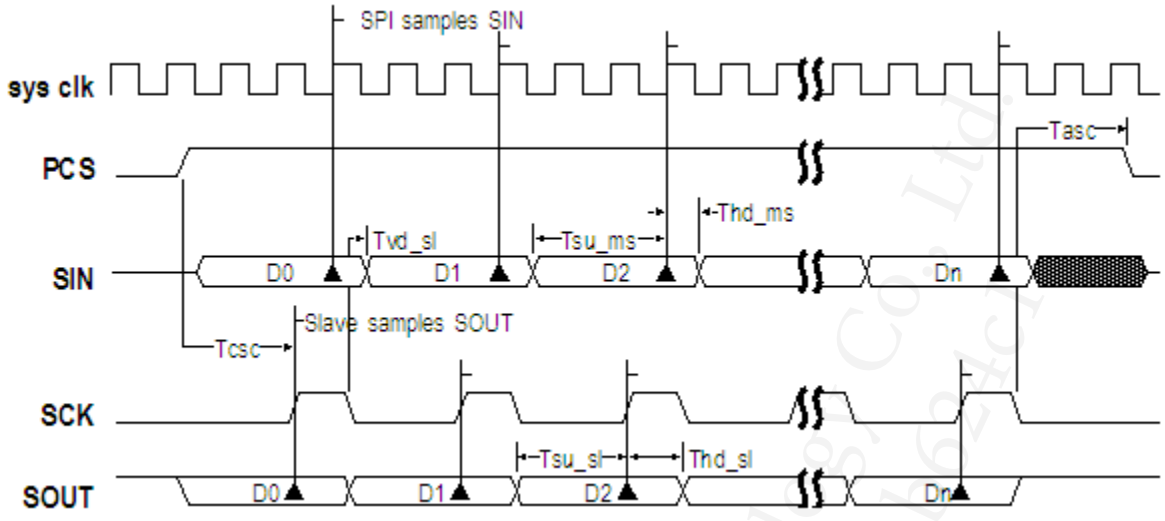


Figure 422. SPI modified transfer format (MTFE=1, CPHA=0, $f_{sck} = f_{sys}/3$)

49.5.4.4 Modified SPI transfer format (MTFE = 1, CPHA = 1)

The following figures show the modified transfer ormat for CPHA = 1. Only the condition where CPOL = 0 is shown. At the start of a transfer, the SPI asserts the PCS signal to the slave device. After the PCS to SCK delay elapses, the master and the slave put data on their SOUT pins at the first edge of SCK. The slave samples the master SOUT signal on the even-numbered edges of SCK. The master samples the slave SOUT signal on the odd-numbered SCK edges starting with the third SCK edge. The slave samples the last bit on the last edge of the SCK. The master samples the last slave SOUT bit on half SCK cycle after the last edge of SCK. No clock edge is visible on the master SCK pin during the sampling of the last bit. The SCK to PCS delay and the after SCK delay must be greater or equal to half of the SCK period.

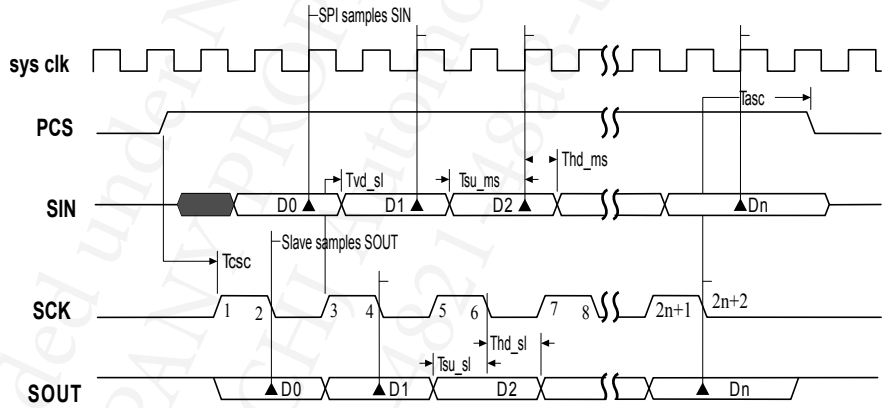


Figure 423. SPI modified transfer format (MTFE=1, CPHA=1, $f_{sck} = f_{sys}/2$)

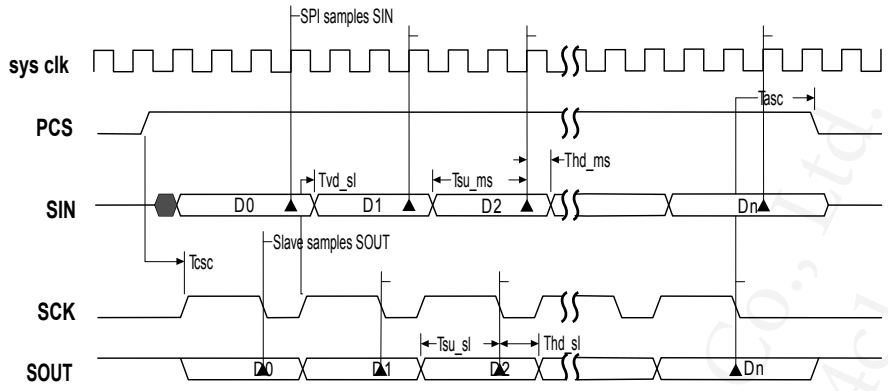


Figure 424. SPI modified transfer format (MTFE=1, CPHA=1, $f_{sck} = f_{sys}/3$)

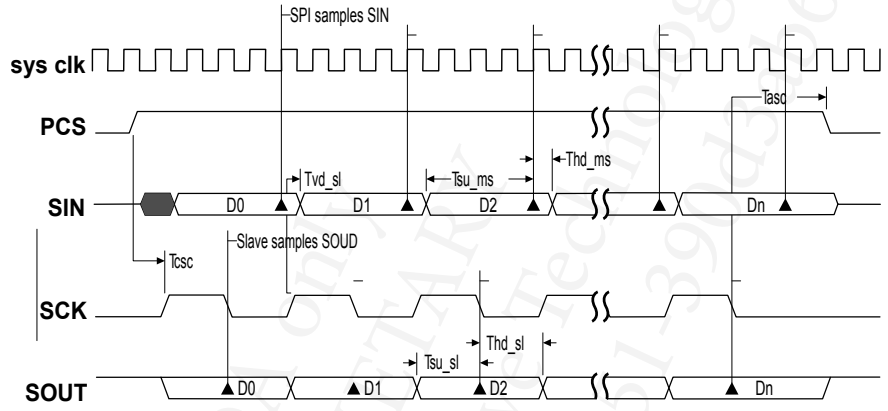


Figure 425. SPI modified transfer format (MTFE=1, CPHA=1, $f_{sck} = f_{sys}/4$)

49.5.4.5 Continuous selection format

Some peripherals must be deselected between every transfer. Other peripherals must remain selected between several sequential serial transfers. The continuous selection format provides the flexibility to handle the following case. The format is enabled for the SPI configuration by setting the CONT bit in the SPI command.

When the value of CONT is 0, the module drives the asserted Chip Select signals to their idle states in between frames. The idle states of these signals are selected by the PCSIS*n* fields in the MCR. The following figure shows the timing diagram for two 4-bit transfers with CPHA = 1 and CONT = 1.

Provided by IHS Markit
COMPANY CONFIDENTIAL
Beijing CHJ Information Technology Co., Ltd.
c8c462e5-4821-4c8-b251-39015b6241

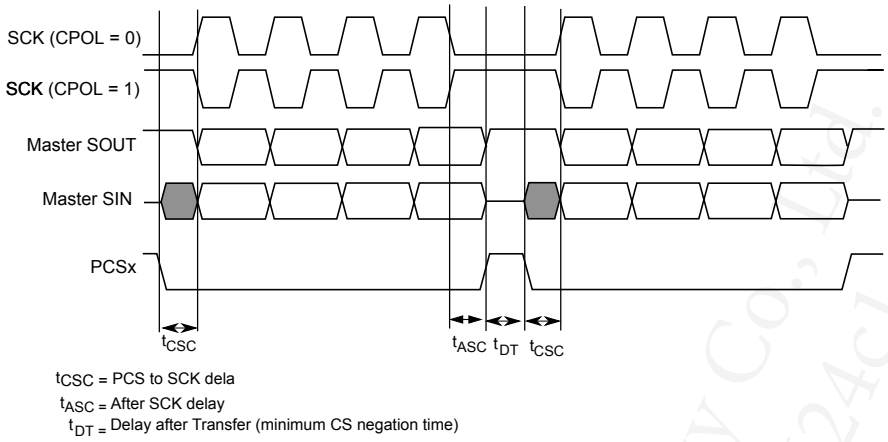


Figure 426. Example of non-continuous format (CPHA=1, CONT=0)

When CONT = 1, the PCS signal remains asserted for the duration of the two transfers. The "Delay between Transfers" (t_{DT}) is not inserted between the transfers. The following figure shows the timing diagram for two 4-bit transfers with CPHA = 1 and CONT = 1.

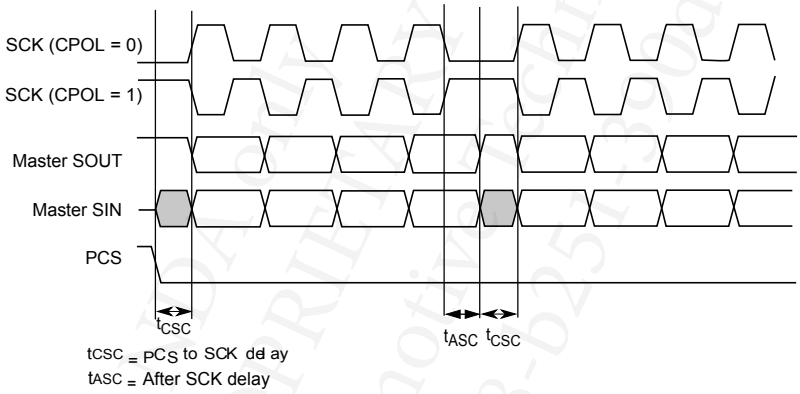


Figure 427. Example of continuous transfer (CPHA=1, CONT=1)

When using the module with continuous selection, follow these rules:

- All transmit commands must have the same PCS*n* field programming.
- The CTARs, selected by transmit commands, must be programmed with the same transfer attributes. Only the FMSZ field can be programmed differently in these CTARs.
- When transmitting multiple frames in this mode, the software must ensure that the last frame has USHR[CONT] deasserted in Master mode and the software must provide sufficient frames in the TX_FIFO to be sent out in Slave mode and the master deasserts the PCSn at end of transmission of the last frame.
- PUSHR[CONT] must be deasserted before asserting MCR[HALT] in master mode. This ensures that the PCSn signals are deasserted. Asserting MCR[HALT] during continuous transfer causes the PCSn signals to remain asserted and hence the slave device cannot transition from Running to Stopped state.

NOTE

You must fill the TX FIFO with the number of entries that are concatenated together under one PCS assertion for both master and slave before the TX FIFO is empty.

When operating in Slave mode, ensure that when the last entry in the TX FIFO is completely transmitted, that is, the corresponding TCF flag is asserted and TXFIFO is empty, the slave is deselected for any further serial communication; otherwise, an underflow error occurs.

49.5.4.6 Fast continuous selection format

The fast continuous selection format has a functioning similar to the continuous selection format except that the inter command delays, t_{ASC} and t_{CSC}, can be masked out and are not inserted by the hardware.

NOTE

The fast continuous selection format is available in the SPI configuration only when the Continuous SCK mode is disabled. Masking of delays is not allowed if the transfer is non-continuous.

The fast continuous selection format is enabled by writing a 1 to MCR[FCPCS]. When this field is asserted, the MASC and MCSC fields of the PUSHR perform the function of mask bits for the transmit frame. These fields individually mask the t_{ASC} and t_{CSC} delays as programmed by the software. A normal continuous selection format includes these two delays for each frame, which is transmitted with the CONT field asserted. To avoid these delays and to speed up the transfer process, the software can mask these delays while programming the command in the PUSHR.

While masking the delays, the software must follow these masking rules, otherwise, correct operation is not guaranteed.

- The MASC field masks the “After SCK” delay for the current frame.
- The MCSC field masks the “PCS to SCK” delay for the next frame.
- The t_{ASC} delay must not be masked when the current frame is the last frame in the continuous selection format.
- The "PCS to SCK" delay for the first frame in the continuous selection format cannot be masked.
- Masking of only t_{ASC} is not allowed. If it is masked, then t_{CSC} must be masked too.
- Masking of both t_{ASC} and t_{CSC} delays is allowed. In this case, the delay between two frames is equal to half the baud rate (baud rate is 1 SCK cycle duration) configured by the software.
- Masking of only t_{CSC} is allowed. In this case, the delay between two frames is equal to the t_{ASC} time and thus the software must ensure that the t_{ASC} time is greater than the baud rate.
- The software must not mask these delays if the continuous selection format is not used and MCR[FCPCS] is asserted.
- Rules applicable to the continuous selection format are applicable here too.

The following figure shows the timing for a fast continuous selection format transfer. Here, seven frames are transferred with both t_{ASC} and t_{CSC} delays masked except for the last frame that terminated the transfer. The last frame has t_{ASC} delay at its end.

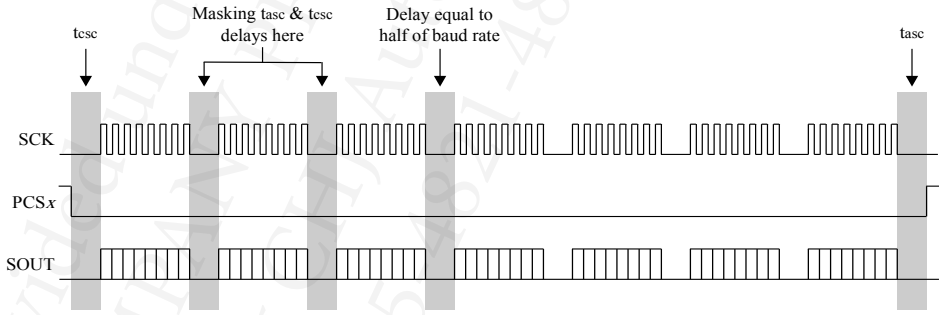


Figure 428. Example of fast continuous selection format

In case any chip select is to be changed, first the fast continuous selection format should be terminated and appropriate delays must be introduced.

49.5.5 Continuous SCK

The module provides the option of generating a Continuous SCK signal for slave peripherals that require a continuous clock.

Continuous SCK is enabled by writing a 1 to MCR[CONT_SCKE]. Enabling this field generates the Continuous SCK only if MCR[HALT] is low. Continuous SCK is valid for all configurations.

Continuous SCK is only supported for CPHA = 1. Clearing CPHA is ignored if the value of the CONT_SCKE field is 1. Continuous SCK is supported for Modified Transfer Format

Clock and transfer attributes for the Continuous SCK mode are set according to the following rules:

- When the module is in SPI configuration, CTAR0 is used initially. At the start of each SPI frame transfer, the CTAR specified by the CTAS for the frame is used.
- In all configurations, the currently selected CTAR remains in use until the start of a frame with a different CTAR specified, or the Continuous SCK mode is terminated.

It is recommended to keep the baud rate the same while using continuous SCK. Switching clock polarity between frames while using continuous SCK can cause errors in the transfer. The Continuous SCK operation is not guaranteed if the module is put into the Module Disable mode.

Enabling continuous SCK disables the PCS to SCK delay, and the Delay after Transfer (t_{DT}) is fixed to one SCK cycle. The following figure is the timing diagram for the Continuous SCK format with continuous selection disabled.

NOTE

In the Continuous SCK mode, for the SPI transfer, CTAR0 should always be used and the TX FIFO must be cleared through MCR[CLR_TXF] before initiating a transfer.

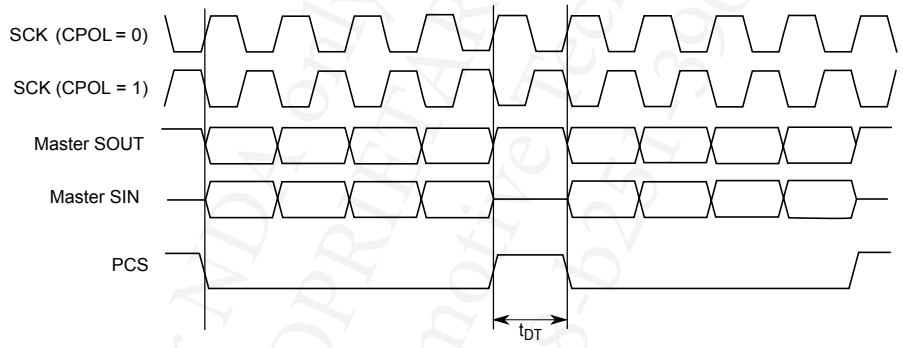


Figure 429. Continuous SCK timing diagram (CONT = 0)

If the CONT bit in the TX FIFO entry is set, PCS remains asserted between the transfers. Under certain conditions, SCK can continue with PCS asserted, but with no data being shifted out of SOUT, which means when SOUT is pulled high. This can cause the slave to receive incorrect data. Those conditions include:

- Continuous SCK with CONT bit set, but no data in the TX FIFO
- Continuous SCK with CONT bit set and entering the Stopped state (see Starting and stopping module transfers)
- Continuous SCK with CONT bit set and entering the Module Disable mode

The following figure shows the timing diagram for the Continuous SCK format with continuous selection enabled.

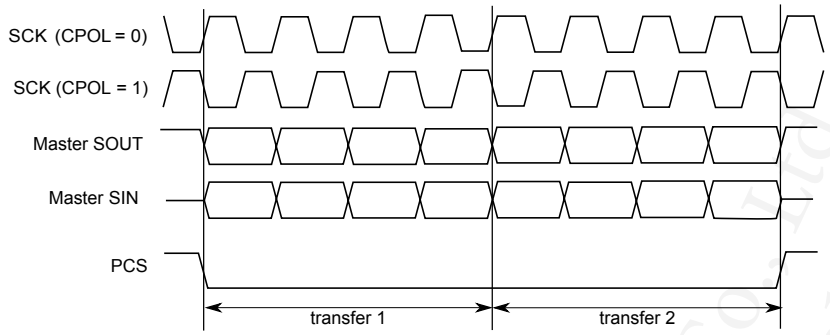


Figure 430. Continuous SCK timing diagram (CONT = 1)

49.5.6 Slave mode operation constraints

The slave mode logic shift register is buffered. This allows the data streaming operation when the module is permanently selected and data is shifted in with a constant rate.

The transmit data is transferred at the second SCK clock edge of each frame to the Shift register if the SS_b signal is asserted, and at and any time when transmit data is ready and the SS_b signal is negated.

Received data is transferred to the receive buffer at last SCK edge of each frame, defined by frame size programmed to the CTAR0/1 register. Then the data from the buffer is transferred to the RXFIFO or DDR.

If the SS_b signal negates before that last SCK edge, the data from the Shift register is lost.

49.5.7 Parity generation and check

The module can generate and check parity in the serial frame. The parity bit replaces the last transmitted bit in the frame. The parity is calculated for all transmitted data bits in frame, not including the last data bit that would be transmitted. The parity generation and control are done on a frame basis. The registers field setting frame size defines the total number of bits in the frame, including the parity bit. Thus, to transmit or receive the same number of data bits with parity check, increase the frame size by one versus the same data size frame without the parity check.

Parity can be selected as odd or even. Parity errors in the received frame set parity error flags in the Status register. The parity error interrupt requests are generated, if enabled. The module can be programmed to stop frame transmission in case of a frame reception with parity error.

49.5.7.1 Parity for SPI frames

When the module is in the master mode the parity generation is controlled by PE and PP bits of the CMD FIFO entries (PUSHR). Setting the PE bit enables parity generation for transmitted SPI frames and parity check for received frames. The PP bit defines polarity of the parity bit.

When continuous PCS selection is used to transmit SPI data, two parity generation scenarios are available:

- Generate or check parity for the whole frame
- Generate or check parity for each sub-frame separately

To generate or check parity for the whole frame, set PE bit only in the last command or TX FIFO entry, forming this frame with the PUSHR.

To generate or check parity for each sub-frame, set PE bit in each command or TX FIFO entry, forming this frame.

If the parity error occurs for received SPI frame, SR[SPEF] is set. If the value of MCR[PES] is 1, the module stops SPI frames transmission. To resume SPI operation, clear SR[SPEF] or MCR[PES].

In Slave mode, the parity is controlled by CTAR0[PE] and CTAR0[PP], as in the case of master mode parity generation without continuous PCS selection.

49.5.8 Interrupts/DMA requests

The module has several conditions that can generate only interrupt requests and two conditions that can generate interrupt or DMA requests. The following table lists these conditions.

Table 333. Interrupt and DMA request conditions

Condition	Flag	Interrupt	DMA
End of Queue (EOQ)	EOQF	Yes	-
TX FIFO Fill	TFFF	Yes	Yes
CMD FIFO Fill	CMDFFF	Yes	Yes
TX FIFO Invalid Write	TFIWF	Yes	-
Transfer Complete	TCF	Yes	-
CMD Transfer Complete	CMDTCF	Yes	-
TX FIFO Underflow	TFUF	Yes	-
RX FIFO Drain	RFDF	Yes	Yes
RX FIFO Overflow	RFOF	Yes	-
SPI Parity Error	SPEF	Yes	-

Each condition has a corresponding field in the SR and a Request Enable field in the RSER. Certain fields, as shown in the above table, generate interrupt requests or DMA requests depending on the configuration of this register.

The module also provides a global interrupt request line, which is asserted when any of individual interrupt requests lines is asserted.

49.5.8.1 EOQ interrupt request

The EOQ interrupt request indicates that the end of a transmit queue is reached. The module generates the interrupt request when the value of RSER[EOQF_RE] is 1 and the value of the EOQ field in the executing SPI command is 1.

If the value of MCR[XSPI] is 1 and the value of the EOQ field in the executing SPI command is 1, the module generates the EOQ interrupt request after the last bit of the last data frame in the command cycle is transmitted.

When the value of MCR[XSPI] is 0, the module generates the interrupt request after the last bit of the SPI frame, with the value of EOQ as 1, is transmitted.

49.5.8.2 Transmit FIFO fill interrupt or DMA request

The transmit FIFO fill request indicates that the TX FIFO is not full. This request is generated when the number of entries in the TX FIFO is less than the maximum number of possible entries, and the value of RSER[TFFF_RE] is 1.

RSER[TFFF_DIRS] selects whether a DMA request or an interrupt request is generated.

NOTE

TFFF clears automatically when DMA is used to fill TX FIFO. You must configure the DMA to fill only one FIFO location per transfer.

To clear TFFF when not using DMA, follow these steps for every PUSH performed using CPU to fill TX FIFO:

1. Wait until the value of TFFF is 1.
2. Write data to PUSHR using CPU.
3. Clear TFFF by writing a 1 to its location. If TX FIFO is not full, the flag does not clear.

49.5.8.3 CMD FIFO fill interrupt or DMA request

The CMD FIFO fill request indicates that the CMD FIFO is not full. This request is generated when the number of entries in the CMD FIFO is less than the maximum number of possible entries, and the value of RSER[CMDFFF_RE] is 1. RSER[CMDFFF_DIRS] selects whether a DMA request or an interrupt request is generated.

The request is useful when the value of MCR[XSPI] is 1 because TX FIFO and CMD FIFO can be filled independently. If the value of MCR[XSPI] is 0, the TX FIFO fill interrupt or DMA request suffices to fill both the FIFOs, which must be filled simultaneously.

NOTE

The CMDFFF flag clears automatically when DMA is used to fill CMD FIFO.

To clear CMDFFF when not using DMA, follow these steps for every PUSH performed using CPU to fill CMD FIFO:

1. Wait until the value of CMDFFF is 1.
2. Write data to PUSHR using CPU.
3. Clear CMDFFF by writing a 1 to its location. If CMD FIFO is not full, this flag does not clear.

49.5.8.4 Transmit FIFO invalid write interrupt request

The transmit FIFO invalid write request is valid only when the value of MCR[XSPI] is 1. This request indicates that data exists in the TX FIFO while the CMD FIFO is empty. Because no command fields are associated with the data present in TX FIFO, this data is considered invalid until a command entry is available. The transmit FIFO invalid write request is generated for this condition when the value of RSER[TFIWF_RE] is 1.

49.5.8.5 Transfer complete interrupt request

The transfer complete request indicates the end of the transfer of a serial frame. This request is generated at the end of each frame transfer when the value of RSER[TCF_RE] is 1.

49.5.8.6 Command transfer complete interrupt request

The command transfer complete request indicates the end of transfer of the last SPI frame in a command cycle. This request is generated when the value of RSER[CMDTCF_RE] is 1.

49.5.8.7 Transmit FIFO underflow interrupt request

The transmit FIFO underflow request indicates that an underflow condition in the TX FIFO has occurred. The transmit underflow condition is detected only for the module operating in the Slave mode and according to the SPI configuration. The value of SR[TFUF] is 1 when the TX FIFO of the module is empty, and a transfer is initiated from an external SPI master. If the value of SR[TFUF] is 1 while the value of RSER[TFUF_RE] is 1, an interrupt request is generated.

49.5.8.8 Receive FIFO drain interrupt or DMA request

The receive FIFO drain request indicates that the RX FIFO is not empty. This request is generated when the number of entries in the RX FIFO is not 0, and the value of RSER[RFDF_RE] is 1.

RSER[RFDF_DIRS] selects whether a DMA request or an interrupt request is generated. You must configure the DMA to drain only one FIFO location per transfer.

49.5.8.9 Receive FIFO overflow interrupt request

The receive FIFO overflow request indicates that an overflow condition in the RX FIFO has occurred. This request is generated when RX FIFO and the Shift register are full, and a transfer is initiated. The value of RSER[RFOF_RE] must be equal to 1 for the interrupt request to be generated.

Depending on the state of MCR[ROOE], the data from the transfer that generated the overflow is either ignored or shifted to the Shift register. If the value of the ROOE field is 1, the incoming data is shifted in to the Shift register. If the value of the ROOE field is 0, the incoming data is ignored.

49.5.8.10 SPI frame parity error interrupt request

The SPI Frame Parity Error Flag indicates that a SPI frame with parity error had been received. The SPEF_RE bit in the RSER must be set for the interrupt request to be generated.

49.5.9 Power saving features

This module supports the following power-saving strategies:

- Module Disable mode: clock gating of non-memory mapped logic

All power-saving features require a logic external to SPI.

49.5.9.1 Module Disable mode

This is a block-specific mode that the module can enter to save power. The host CPU can initiate the Module Disable mode by writing a 1 to MCR[MDIS]. The hardware can also initiate this mode.

When the value of the MDIS field is 1, the module negates the Clock Enable signal at the next frame boundary. After the Clock Enable signal is negated, it is said to have entered the Module Disable Mode. This also puts the module in the Stopped state. The value of SR[TXRXS] must be 0 to indicate the Stopped state. If implemented, the Clock Enable signal can stop the clock to the non-memory mapped logic. When Clock Enable is negated, the module is in a dormant state, but the memory mapped registers are still accessible. Certain read or write operations have a different effect when the module is in the Module Disable mode. Reading the POPR does not change the state of the RX FIFO. Similarly, writing to the PUSHR does not change the state of the TX FIFO or CMD FIFO. Clearing either of the FIFOs has no effect in the Module Disable mode. Also, changes to MCR[DIS_TXF] and MCR[DIS_RXF] have no effect in the Module Disable mode. In this mode, all status bits and register flags in the module return the correct values when read, but writing to them has no effect. Also, writing to the TCR during the Module Disable mode has no effect. Interrupt and DMA request signals cannot be cleared in the Module Disable mode.

49.6 Initialization/application information

This section describes how to initialize the module.

49.6.1 How to manage queues

Queues are not a part of this module, but the module includes features to support queue management.

Queues are primarily supported in SPI configuration, and here are the steps to manage them.

1. When the module executes the last command word from a queue, the EOQ bit in the command word indicates that it is the last entry in the queue.
2. At the end of the transfer, the value of SR[EOQF] is configured as 1.
3. If the value of SR[EOQF] is 1, the field disables serial transmission and reception of data, putting the module in the Stopped state. The value 0 is written to SR[TXRXS] indicating the Stopped state.
4. The DMA can continue to fill TX FIFO until it is full or step 5 occurs.

5. Disable DMA transfers by disabling the DMA enable request for the DMA channel assigned to TX FIFO, CMD FIFO, and RX FIFO. This is done by clearing the corresponding DMA enable request bits in the DMA controller.
6. Ensure that all the received data in RX FIFO has been transferred to the memory receive queue by reading SR[RXCNT] or by checking SR[RFDF] after each read operation of the POPR.
7. Modify the DMA descriptor of TX and RX channels for new queues.
8. Flush TX FIFO and CMD FIFO by writing a 1 to MCR[CLR_TXF] and flushing RX FIFO by writing a 1 to MCR[CLR_RXF].
9. Clear the transfer count either by setting the CTCNT bit in the command word of the first entry in the new queue or via the CPU writing directly to TCR[SPI_TCNT].
10. Enable the DMA channel through the DMA enable request. DMA channels are assigned to modules TX FIFO, CMD FIFO, and RX FIFO. For each module, a separate DMA enable request bit is available.
11. Enable serial transmission and serial reception of data by clearing the EOQF field.

49.6.2 Switching Master and Slave modes

When changing modes in the module, follow these steps to guarantee a proper operation.

1. Halt it by writing a 1 to MCR[HALT].
2. Clear the transmit and receive FIFOs by writing a 1 to MCR[CLR_TXF] and MCR[CLR_RXF].
3. Set the appropriate mode in MCR[MSTR] and enable it by writing a 0 to MCR[HALT].

49.6.3 Initializing module in Master/Slave modes

After the appropriate mode in MCR[MSTR] is configured, the module is enabled by writing a 0 to MCR[HALT]. The module's slave must be enabled before you enable its master. This ensures that the slave is ready to be communicated with, before the master initializes communication.

49.6.4 Baud rate settings

The following table shows the baud rate that is generated based on the combination of the baud rate prescaler, PBR, and the baud rate scaler, BR, in the CTARs. The values calculated assume a 100 MHz protocol frequency, and the value of the DBR field is 0..

NOTE

The clock frequency provided in this chapter is an example. See the Clocking chapter for details on the frequency used to drive this module in the chip.

Table 334. Baud rate values (bit/s)

		Baud rate divider prescaler values			
		2	3	5	7
Baud rate scaler values	2	25.0M	16.7M	10.0M	7.14M
	4	12.5M	8.33M	5.00M	3.57M
	6	8.33M	5.56M	3.33M	2.38M
	8	6.25M	4.17M	2.50M	1.79M
	16	3.12M	2.08M	1.25M	893k

Table continues on the next page...

Table 334. Baud rate values (bit/s) (continued)

		Baud rate divider prescaler values			
		2	3	5	7
	32	1.56M	1.04M	625k	446k
	64	781k	521k	312k	223k
	128	391k	260k	156k	112k
	256	195k	130k	78.1k	55.8k
	512	97.7k	65.1k	39.1k	27.9k
	1024	48.8k	32.6k	19.5k	14.0k
	2048	24.4k	16.3k	9.77k	6.98k
	4096	12.2k	8.14k	4.88k	3.49k
	8192	6.10k	4.07k	2.44k	1.74k
	16384	3.05k	2.04k	1.22k	872
	32768	1.53k	1.02k	610	436

49.6.5 Delay settings

The following table shows the values for the Delay after Transfer (t_{DT}) and CS to SCK Delay (T_{CSC}) that can be generated based on the prescaler values and the scaler values set in the CTARs. The values calculated assume a 100 MHz protocol frequency.

NOTE

The clock frequency mentioned in this section is only an example. See the clocking chapter for the frequency used to drive this module in the chip.

Table 335. Delay values

		Delay scaler values (CTAR[PDT])			
		1	3	5	7
Delay scaler values (CTAR[DT])	2	20.0 ns	60.0 ns	100.0 ns	140.0 ns
	4	40.0 ns	120.0 ns	200.0 ns	280.0 ns
	8	80.0 ns	240.0 ns	400.0 ns	560.0 ns
	16	160.0 ns	480.0 ns	800.0 ns	1.1 µs
	32	320.0 ns	960.0 ns	1.6 µs	2.2 µs
	64	640.0 ns	1.9 µs	3.2 µs	4.5 µs

Table continues on the next page...

Provided under NDA ONLY COMPANY PROPRIETARY Beijing Hyundai Automotive Technology Co., Ltd. 9d02c25-482f-4828-8251-390d3a162d1

Table 335. Delay values (continued)

		Delay scaler values (CTAR[PDT])			
		1	3	5	7
	128	1.3 µs	3.8 µs	6.4 µs	9.0 µs
	256	2.6 µs	7.7 µs	12.8 µs	17.9 µs
	512	5.1 µs	15.4 µs	25.6 µs	35.8 µs
	1024	10.2 µs	30.7 µs	51.2 µs	71.7 µs
	2048	20.5 µs	61.4 µs	102.4 µs	143.4 µs
	4096	41.0 µs	122.9 µs	204.8 µs	286.7 µs
	8192	81.9 µs	245.8 µs	409.6 µs	573.4 µs
	16384	163.8 µs	491.5 µs	819.2 µs	1.1 ms
	32768	327.7 µs	983.0 µs	1.6 ms	2.3 ms
	65536	655.4 µs	2.0 ms	3.3 ms	4.6 ms

49.6.6 Calculation of FIFO pointer addresses

Complete visibility of the FIFO contents is available through the FIFO registers, and valid entries can be identified through a memory-mapped pointer and counter for each FIFO. The pointer to the first-in entry in each FIFO is memory mapped. For the TX FIFO, the first-in pointer is Transmit Next Pointer (TXNXTPTR), and for the CMD FIFO, the first-in pointer is the Command Next Pointer (CMDNXTPTR). For the RX FIFO, the first-in pointer is the Pop Next Pointer (POPNXTPTR). The following figure illustrates the concept of first-in and last-in FIFO entries along with the FIFO counter. The TX FIFO is chosen for illustration, but the concepts carry over. See TX FIFO buffering mechanism, CMD FIFO buffering mechanism, and RX FIFO buffering mechanism for details on the FIFO operation.

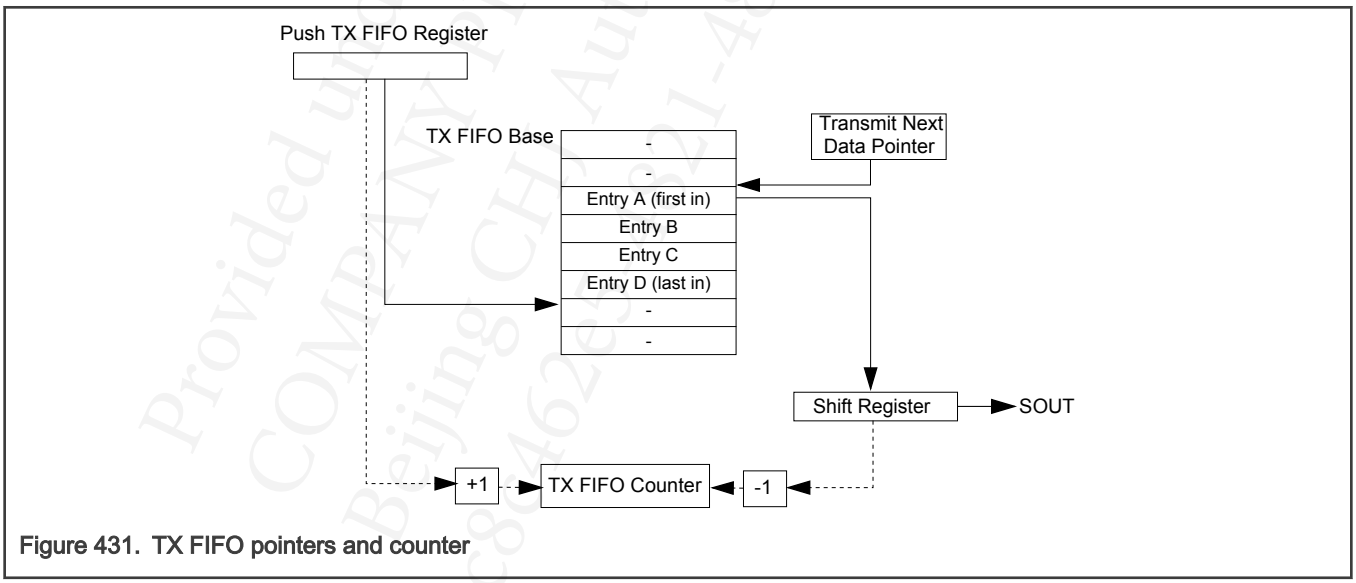


Figure 431. TX FIFO pointers and counter

49.6.6.1 Address calculation for the first-in entry and last-in entry in the TX FIFO

Use the following equation to compute the memory address of the first-in entry in the TX FIFO:

- First-in EntryAddress = TXFIFOBase+[4xTXNXTPTR]

Use the following equation to compute the memory address of the last-in entry in the TX FIFO:

- Last-in EntryAddress = TXFIFOBase+4x[TXCTR+TXNXTPTR - 1]mod[TXFIFOdepth]

Here is an explanation of the different components of these equations:

- TX FIFO base: Base address of TX FIFO
- TXCTR: TX FIFO Counter
- TXNXTPTR: Transmit Next Pointer
- TX FIFO Depth: Transmit FIFO depth, implementation specific

49.6.6.2 Address calculation for the first-in entry and last-in entry in the CMD FIFO

Use the following equation to compute the memory address of the first-in entry in the CMD FIFO:

- First-in EntryAddress = CMDFIFOBase+[4xCMDNXTPTR]

Use the following equation to compute the memory address of the last-in entry in the CMD FIFO:

- Last-in EntryAddress = CMDFIFOBase+4x[CMDCTR+CMDNXTPTR - 1]mod[CMDFIFOdepth]

Here is an explanation of the different components of these equations:

- CMD FIFO Base: Base address of CMD FIFO
- CMDCTR: CMD FIFO Counter
- CMDNXTPTR: Command Next Pointer
- CMD FIFO Depth: Command FIFO depth, implementation specific

49.6.6.3 Address calculation for the first-in entry and last-in entry in the RX FIFO

Use the following equation to compute the memory address of the first-in entry in the RX FIFO:

- First-in EntryAddress = RXFIFOBase+[4xPOPNXTPTR]

Use the following equation to compute the memory address of the last-in entry in the RX FIFO:

- Last-in EntryAddress = RXFIFOBase+4x[RXCTR+POPNXTPTR - 1]mod[RXFIFOdepth]

Here is an explanation of the different components of these equations:

- RX FIFO Base: Base address of RX FIFO
- RXCTR: TX FIFO Counter
- POPNXTPTR: Pop Next Pointer
- RX FIFO Depth: Receive FIFO depth, implementation specific

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39013b624c1

Chapter 50
CAN (FlexCAN)

50.1 Chip-specific FlexCAN information

50.1.1 Overview

This device contains four FlexCAN modules (FlexCAN_0 to FlexCAN_3) that implement the CAN 2.0B and the Flexible Data Rate (FD) CAN protocol specification.

- MCR[MDIS] field is set to 1 by default, each FlexCAN module must be enabled through the software after start-up.
- Doze mode is not supported.

50.1.2 Message buffers and data rate

Each FlexCAN instance support 128 message buffers and upto 8 Mbps programmable bit rate, subject to presence of Flexible Data Rate (FD) feature. All FlexCAN modules support individual receive buffer masks and ECC on the SRAM arrays.

The expanded message buffer memory feature allows the use of 28 additional message buffers of 64-byte payload in CAN FD mode.

50.1.3 Rx FIFO

The enhanced Rx FIFO feature extends the RxFIFO depth and support CAN FD messages.

The module interrupt requests for the Enhanced Rx FIFO feature, enabled with the Enhanced Rx FIFO Interrupt Enable Register (ERFIER), are not mapped to system-level interrupts. If an application enables these module interrupts, software can poll the fields of the Enhanced Rx FIFO Status Register (ERFSR) to determine when the module interrupt requests occur.

50.1.4 Mode configuration

FlexCAN registers always operate in supervisor mode regardless of whether REG_PROT or the system is in user mode.

50.1.5 Booting with FlexCAN

FlexCAN_0 interfaces support booting from boot ROM.

50.1.6 High-resolution timestamp

The FlexCAN instances on this chip use the Timestamp module as the timebase source for the high-resolution timestamp.

You can enable use of the high-resolution timestamp on a per-module basis via the following fields in each FlexCAN instance:

- CTRL2[MBTSBASE] (set to 01b or 10b)
- CTRL2[TSTAMPCAP] (set to 01b, 10b, or 11b)

Using the high-resolution timestamp timebase requires the Timestamp module to be properly configured and running.

NOTE

Changing the timebase source in the Timestamp module requires disabling all module instances using the Timestamp module timebase. To disable a FlexCAN instance, put the instance in Freeze mode.

Freescale Under NDA Only
CONFIDENTIAL PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b23c-90a8b6241

50.2 Introduction

The FlexCAN module is a communication controller implementing the CAN protocol according to the ISO 11898-1 standard and CAN 2.0 B protocol specifications. A general block diagram is shown in Figure 432, which describes the main subblocks implemented in the FlexCAN module, including:

- One associated memory for storing message buffers
- Receive Global Mask registers
- Receive Individual Mask registers
- Receive FIFO filters
- Receive FIFO ID filters

The functions of the submodules are described in subsequent sections.

NOTE

Legacy Rx FIFO cannot be used in FD mode.

NOTE

In CAN FD mode, the Enhanced Rx FIFO feature should be used instead of Legacy Rx FIFO.

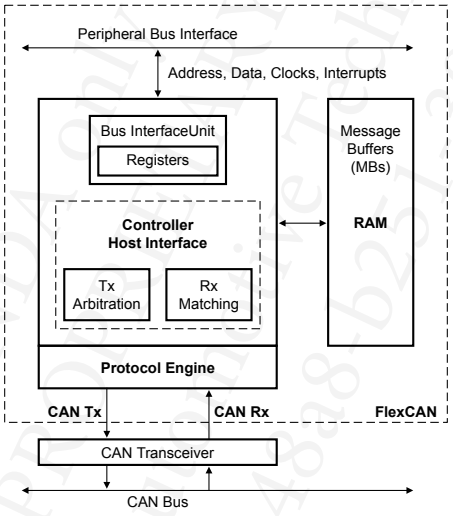


Figure 432. FlexCAN block diagram

50.2.1 Overview

The CAN protocol was primarily designed to be used as a vehicle serial data bus, meeting the specific requirements of this field:

- Real-time processing
- Reliable operation in the EMI environment of a vehicle
- Cost-effectiveness
- Required bandwidth

The FlexCAN module is a full implementation of the CAN protocol specification, the CAN with Flexible Data rate (CAN FD) protocol, and the CAN 2.0 version B protocol, which supports both standard and extended message frames and long payloads up to 64 bytes, transferred at faster rates (up to 8 Mb/s—Note that this baud rate can be reached only in the Rx and Tx digital pins, and also depends on the physical layer of the application). The message buffers are stored in an embedded RAM dedicated to the FlexCAN module. See the chip-specific FlexCAN information for the actual number of message buffers configured in the chip.

The Protocol Engine (PE) submodule manages the serial communication on the CAN bus:

- Requesting RAM access for receiving and transmitting message frames
- Validating received messages
- Performing error handling
- Detecting CAN FD messages

The Controller Host Interface (CHI) submodule manages message buffer selection for reception and transmission, taking care of arbitration and ID matching algorithms for both CAN FD and non-CAN FD message formats.

The Bus Interface Unit (BIU) submodule controls access to and from the internal interface bus, in order to establish connection to the CPU and to other blocks. Clocks, address and data buses, interrupt outputs, DMA and test signals are accessed through the BIU.

50.2.2 FlexCAN module features

The FlexCAN module includes these distinctive features:

- Full implementation of the CAN with Flexible Data Rate (CAN FD) protocol specification and CAN protocol specification, Version 2.0 B
  - Standard data frames
  - Extended data frames
  - Zero to sixty four bytes data length
  - Programmable bit rate (see the chip-specific FlexCAN information for the specific maximum rate configuration)
  - Content-related addressing
- Compliant with the ISO 11898-1 standard
- Flexible mailboxes configurable to store 0 to 8, 16, 32, or 64 bytes data length
- Each mailbox configurable as receive or transmit, all supporting standard and extended messages
- Individual Rx Mask registers per mailbox
- Full-featured legacy Rx FIFO with storage capacity for up to six frames and automatic internal pointer handling with DMA support
- Full-featured enhanced Rx FIFO with storage capacity for up to 32 CAN FD frames and automatic internal pointer handling with DMA support
- Transmission abort capability
- Flexible message buffers (MBs), totaling 128 message buffers of 8 bytes data length each, configurable as Rx or Tx
- RAM not used by reception or transmission structures can be used as general purpose RAM space
- Listen-Only mode capability
- Programmable Loop-Back mode supporting self-test operation
- Programmable transmission priority scheme: lowest ID, lowest buffer number, or highest priority
- Time stamp based on 32-bit free running timer, with an optional external time tick
- Global network time, synchronized by a specific message
- Maskable interrupts
- Independence from the transmission medium (an external transceiver is assumed)
- Short latency time due to an arbitration scheme for high-priority messages
- Low power mode
- Transceiver delay compensation feature when transmitting CAN FD messages at faster data rates

- Remote request frames may be managed automatically or by software
- CAN bit time settings and configuration bits can only be written in Freeze mode
- Tx mailbox status (Lowest priority buffer or empty buffer)
- Identifier Acceptance Filter Hit Indicator (IDHIT) register for received frames
- SYNCH bit available in Error in Status 1 register to indicate that the module is synchronous with CAN bus
- CRC status for transmitted message
- Legacy Rx FIFO Global Mask register
- Selectable priority between mailboxes and Rx FIFO during matching process
- Powerful legacy Rx FIFO ID filtering, capable of matching incoming IDs against either 128 extended, 256 standard, or 512 partial (8 bit) IDs, with up to 32 individual masking capability
- Powerful Enhanced Rx FIFO ID filtering, capable of matching incoming IDs against either 64 extended or 128 standard ID filter elements with three filtering schemes: mask + filter, range, and two filters without mask.
- 100% backward compatibility with previous FlexCAN version
- Supports detection and correction of errors in memory read accesses

  Each byte of FlexCAN memory is associated to 5 parity bits, and the error correction mechanism ensures that in this 13-bit word, errors in one bit can be corrected (correctable errors) and errors in 2 bits can be detected but not corrected (non-correctable errors).

50.2.3 Modes of operation

The FlexCAN module has these functional modes:

- Normal mode:

  In Normal mode, the module operates receiving and/or transmitting message frames, errors are managed normally, and all CAN Protocol functions are enabled.

- Freeze mode:

  Freeze mode is enabled when MCR[FRZ] is asserted. If enabled, Freeze mode is entered when MCR[HALT] is set or when is requested at chip level and MCR[FRZ_ACK] is asserted by the FlexCAN. In this mode, no transmission or reception of frames is done and synchronicity to the CAN bus is lost. See Freeze mode for more information.

- Loop-Back mode:

  The module enters this mode when CTRL1[LPB] is asserted. In this mode, FlexCAN performs an internal loop back that can be used for self-test operation. The bit stream output of the transmitter is internally fed back to the receiver input. The Rx CAN input pin is ignored and the Tx CAN output goes to the recessive state (logic '1'). FlexCAN behaves as it normally does when transmitting and treats its own transmitted message as a message received from a remote node. In this mode, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field to ensure proper reception of its own message. Both transmit and receive interrupts are generated.

- Listen-Only mode:

  The module enters this mode when CTRL1[LOM] is asserted. In this mode, transmission is disabled, all error counters are frozen, and the module operates in a CAN Error Passive mode. Only messages acknowledged by another CAN station will be received. If FlexCAN detects a message that has not been acknowledged, it will flag a BIT0 error (without changing the REC), as if it was trying to acknowledge the message.

- CAN FD Active mode:

  In this mode, FlexCAN is capable of transmitting and receiving all messages formatted according to the CAN FD Protocol and CAN 2.0 Protocol 2.0 in an interleaved fashion. The CPU can set the FlexCAN into CAN FD Active mode by configuring MCR[FDEN] in Freeze mode.

It is important to know which features are available in CAN FD active mode. This table lists the differences between FD and classical modes.

Table 336. Differences between CAN classical and CAN FD

Feature	CAN classical	CAN FD
Legacy Rx FIFO DMA	Yes	No
Legacy Rx FIFO	Yes	No
Enhanced Rx FIFO DMA	Yes	Yes
Enhanced Rx FIFO	Yes	Yes

For low-power operation, the FlexCAN module has:

- Module Disable mode:

  This low-power mode is entered when MCR[MDIS] is asserted by the CPU and MCR[LPM_ACK] is asserted by FlexCAN. When disabled, the module issues a request to disable the clocks to the CAN Protocol Engine and Controller Host Interface submodules. Exit from this mode is done by negating MCR[MDIS]. See Module Disable mode for more information.

50.3 FlexCAN signal descriptions

The FlexCAN module has two I/O signals connected to the external chip pins. These signals are summarized in the following table and described in more detail in the next subsections.

Table 337. FlexCAN signal descriptions

Signal	Description	I/O
CAN Rx	CAN receive pin	Input
CAN Tx	CAN transmit pin	Output

50.3.1 CAN Rx

This pin is the receive pin from the CAN bus transceiver. Dominant state is represented by logic level 0. Recessive state is represented by logic level 1.

50.3.2 CAN Tx

This pin is the transmit pin to the CAN bus transceiver. Dominant state is represented by logic level 0. Recessive state is represented by logic level 1.

50.4 Memory map/register definition

This section describes the registers and data structures in the FlexCAN module. The base address of the module depends on the particular memory map of the chip.

50.4.1 FlexCAN memory mapping

The address space occupied by FlexCAN has 128 bytes for registers starting at the module base address, followed by embedded RAM starting at address offset 0080h.

Each individual register is identified by its complete name and the corresponding mnemonic .

NOTE

An invalid register access will result in a bus error. This includes reading a write-only register, writing a read-only register, or accessing an invalid address.

Table 338. Register reset information

Register	Affected by hard reset	Affected by soft reset
Module Configuration Register (MCR)	Yes	Yes
Control 1 register (CTRL1)	Yes	No
Free Running Timer register (TIMER)	Yes	Yes
Rx Mailboxes Global Mask register (RXMGMASK)	No	No
Rx Buffer 14 Mask register (RX14MASK)	No	No
Rx Buffer 15 Mask register (RX15MASK)	No	No
Error Counter Register (ECR)	Yes	Yes
Error and Status 1 Register (ESR1)	Yes	Yes
Interrupt Masks 2 register (IMASK2)	Yes	Yes
Interrupt Masks 1 register (IMASK1)	Yes	Yes
Interrupt Flags 2 register (IFLAG2)	Yes	Yes
Interrupt Flags 1 register (IFLAG1)	Yes	Yes
Control 2 Register (CTRL2)	Yes	No
Error and Status 2 Register (ESR2)	Yes	Yes
CRC Register (CRCR)	Yes	Yes
Rx FIFO Global Mask register (RXFGMASK)	No	No
Rx FIFO Information Register (RXFIR)	No	No
CAN Bit Timing Register (CBT)	Yes	No
Interrupt Masks 4 register (IMASK4)	Yes	Yes
Interrupt Masks 3 register (IMASK3)	Yes	Yes
Interrupt Flags 4 register (IFLAG4)	Yes	Yes
Interrupt Flags 3 register (IFLAG3)	Yes	Yes
Message buffers	No	No
Rx Individual Mask Registers	No	No
Memory Error Control register (MECR)	Yes	Yes

Table continues on the next page...

Table 338. Register reset information (continued)

Register	Affected by hard reset	Affected by soft reset
Error Injection Address register (ERRIAR)	Yes	Yes
Error Injection Data Pattern register (ERRIDPR)	Yes	Yes
Error Injection Parity Pattern register (ERRIPPR)	Yes	Yes
Error Report Address register (RERRAR)	Yes	Yes
Error Report Data register (RERRDR)	Yes	Yes
Error Report Syndrome register (RERRSYNR)	Yes	Yes
Error Status register (ERRSR)	Yes	Yes
Enhanced CAN Bit Timing Prescalers (EPRS)	Yes	No
Enhanced Nominal CAN Bit Timing (ENCBT)	Yes	No
Enhanced Data Phase CAN bit Timing (EDCBT)	Yes	No
Enhanced Transceiver Delay Compensation (ETDC)	Yes	No
CAN FD Control register (FDCTRL)	Yes	No
CAN FD Bit Timing register (FDCBT)	Yes	No
CAN FD CRC register (FDCRC)	Yes	Yes
Enhanced Rx FIFO Control Register (ERFCR)	Yes	Yes
Enhanced Rx FIFO Interrupt Enable Register (ERFIER)	Yes	Yes
Enhanced Rx FIFO Status Register (ERFSR)	Yes	Yes
High Resolution Time Stamp (HR_TIME_STAMP)	No	No
Extra CAN FD Message Buffers	No	No
Enhanced Rx FIFO	No	No
Enhanced Rx FIFO Filter Element (ERFFEL)	No	No

The FlexCAN module can store CAN messages for transmission and reception using mailboxes and Rx FIFO structures.

50.4.2 CAN register descriptions

The table below shows the FlexCAN memory map.

The address range from offset 80h–87Fh allocates the one-hundred-twenty-eight 128-bit message buffers (MBs).

The address range from offset 1000h–17FFh allocates an extra message buffer space which can be used for CAN FD operation mode.

The memory maps for the message buffers are in FlexCAN message buffer memory map.

The address range from offset 2000h–204Ch allocates the Enhanced Rx FIFO output, and the address range from offset 2050h–29FCh allocates the rest of Enhanced RX FIFO 31 elements.

The memory map for the Enhanced Rx FIFO is in Enhanced Rx FIFO structure.

50.4.2.1 CAN memory map

CAN_0 base address: 401B_4000h

CAN_1 base address: 401B_E000h

CAN_2 base address: 402A_8000h

CAN_3 base address: 402B_2000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration register (MCR)	32	RW	D890_000Fh
4h	Control 1 register (CTRL1)	32	RW	0000_0000h
8h	Free Running Timer (TIMER)	32	RW	0000_0000h
10h	Rx Mailboxes Global Mask register (RXMGMASK)	32	RW	See description
14h	Rx 14 Mask register (RX14MASK)	32	RW	See description
18h	Rx 15 Mask register (RX15MASK)	32	RW	See description
1Ch	Error Counter (ECR)	32	RW	0000_0000h
20h	Error and Status 1 register (ESR1)	32	R2C	0000_0000h
24h	Interrupt Masks 2 register (IMASK2)	32	RW	0000_0000h
28h	Interrupt Masks 1 register (IMASK1)	32	RW	0000_0000h
2Ch	Interrupt Flags 2 register (IFLAG2)	32	W1C	0000_0000h
30h	Interrupt Flags 1 register (IFLAG1)	32	W1C	0000_0000h
34h	Control 2 register (CTRL2)	32	RW	0010_0000h
38h	Error and Status 2 register (ESR2)	32	RO	0000_0000h
44h	CRC register (CRCR)	32	RO	0000_0000h
48h	Legacy Rx FIFO Global Mask register (RXFGMASK)	32	RW	See description
4Ch	Legacy Rx FIFO Information register (RXFIR)	32	RO	See description
50h	CAN Bit Timing register (CBT)	32	RW	0000_0000h
68h	Interrupt Masks 4 register (IMASK4)	32	RW	0000_0000h
6Ch	Interrupt Masks 3 register (IMASK3)	32	RW	0000_0000h
70h	Interrupt Flags 4 register (IFLAG4)	32	W1C	0000_0000h
74h	Interrupt Flags 3 register (IFLAG3)	32	W1C	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
880h - A7Ch	Rx Individual Mask registers (RXIMR0 - RXIMR127)	32	RW	See description
AE0h	Memory Error Control register (MECR)	32	RW	800C_0080h
AE4h	Error Injection Address register (ERRIAR)	32	RW	0000_0000h
AE8h	Error Injection Data Pattern register (ERRIDPR)	32	RW	0000_0000h
AECh	Error Injection Parity Pattern register (ERRIPPR)	32	RW	0000_0000h
AF0h	Error Report Address register (RERRAR)	32	RO	0000_0000h
AF4h	Error Report Data register (RERRDR)	32	RO	0000_0000h
AF8h	Error Report Syndrome register (RERRSYNR)	32	RO	0000_0000h
AFCh	Error Status register (ERRSR)	32	W1C	0000_0000h
BF0h	Enhanced CAN Bit Timing Prescalers (EPRS)	32	RW	0000_0000h
BF4h	Enhanced Nominal CAN Bit Timing (ENCBT)	32	RW	0000_0000h
BF8h	Enhanced Data Phase CAN bit Timing (EDCBT)	32	RW	0000_0000h
BFCh	Enhanced Transceiver Delay Compensation (ETDC)	32	RW	0000_0000h
C00h	CAN FD Control register (FDCTRL)	32	RW	8000_0100h
C04h	CAN FD Bit Timing register (FDCBT)	32	RW	0000_0000h
C08h	CAN FD CRC register (FDCRC)	32	RO	0000_0000h
C0Ch	Enhanced Rx FIFO Control register (ERFCR)	32	RW	0000_0000h
C10h	Enhanced Rx FIFO Interrupt Enable register (ERFIER)	32	RW	0000_0000h
C14h	Enhanced Rx FIFO Status register (ERFSR)	32	WORZ	0000_0000h
C30h - E2Ch	High Resolution Time Stamp (HR_TIME_STAMP0 - HR_TIME_STAMP127)	32	RW	See description
3000h - 31FCh	Enhanced Rx FIFO Filter Element (ERFFEL0 - ERFFEL127)	32	RW	See description

50.4.2.2 Module Configuration register (MCR)

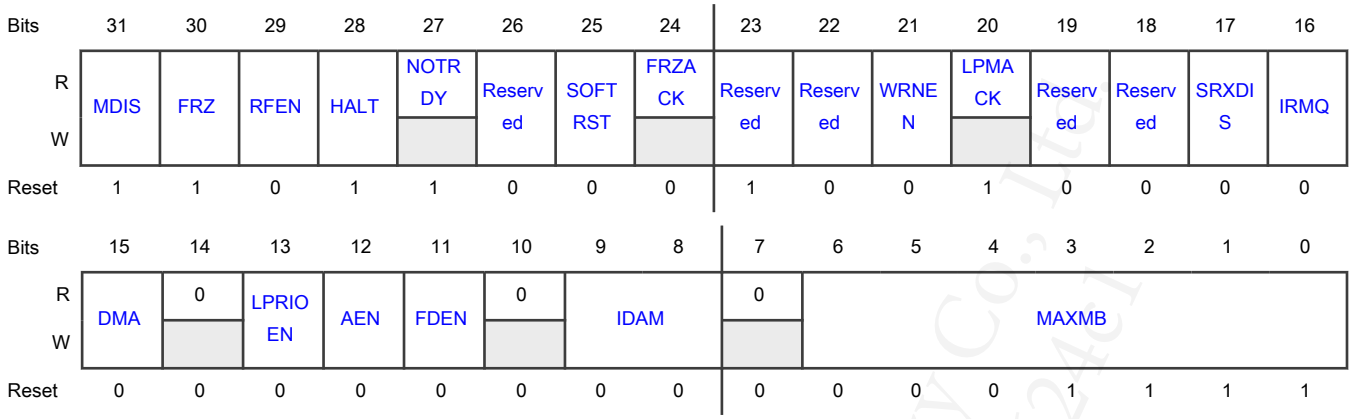
Offset

Register	Offset
MCR	0h

Function

This register defines global system configurations, such as the module operation modes and the maximum message buffer configuration.

Diagram



Fields

Field	Function
31 MDIS	Module Disable This bit controls whether FlexCAN is enabled or not. When disabled, FlexCAN disables the clocks to the CAN Protocol Engine and Controller Host Interface submodules. This bit is not affected by soft reset. 0b - Enable the FlexCAN module. 1b - Disable the FlexCAN module.
30 FRZ	Freeze Enable The FRZ bit specifies the FlexCAN behavior when MCR[HALT] is set or when Debug mode is requested at chip level. When FRZ is asserted, FlexCAN is enabled to enter Freeze mode. Negation of this bit field causes FlexCAN to exit from Freeze mode. This bit is set by hardware when a non-correctable error is detected (MECR[NCEFAFRZ] is asserted). 0b - Not enabled to enter Freeze mode. 1b - Enabled to enter Freeze mode.
29 RFEN	Legacy Rx FIFO Enable This bit controls whether the Legacy Rx FIFO feature is enabled or not. When RFEN is set, MBs 0 to 5 cannot be used for normal reception and transmission because the corresponding memory region (80h–DCh) is used by the FIFO engine as well as additional MBs (up to 32, depending on CTRL2[RFFN] setting) which are used as Legacy Rx FIFO ID filter table elements. RFEN also impacts the definition of the minimum number of peripheral clocks per CAN bit as described in Table 380. This bit can be written in Freeze mode only, because it is blocked by hardware in other modes. **NOTE** This bit cannot be set when CAN FD operation is enabled (see FDEN bit). **NOTE** RFEN bit must not be set if ERFCR[ERFEN] is set. 0b - Legacy Rx FIFO not enabled. 1b - Legacy Rx FIFO enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 HALT	Halt FlexCAN Assertion of this bit puts the FlexCAN module into Freeze mode. The CPU should clear it after initializing the message buffers and the Control registers CTRL1 and CTRL2. No reception or transmission is performed by FlexCAN before this bit is cleared. Freeze mode cannot be entered when FlexCAN is in a low power mode. The HALT bit is set by hardware when a non-correctable error is detected and MECR[NCEFAFRZ] is asserted. 0b - No Freeze mode request. 1b - Enters Freeze mode if the FRZ bit is asserted.
27 NOTRDY	FlexCAN Not Ready This read-only bit indicates that FlexCAN is either in Disable mode or Freeze mode. It is negated when FlexCAN has exited these modes. This bit is not affected by soft reset. 0b - FlexCAN module is either in Normal mode, Listen-Only mode, or Loop-Back mode. 1b - FlexCAN module is either in Disable mode or Freeze mode.
26 —	Reserved When writing to this field, always write the reset value.
25 SOFTRST	Soft Reset When SOFTRST is asserted, FlexCAN resets its internal state machines and some of the memory-mapped registers. SOFTRST can be asserted directly by the CPU when it writes to the MCR register. Because soft reset is synchronous and has to follow a request/acknowledge procedure across clock domains, it may take some time to fully propagate its effect. The SOFTRST bit remains asserted when reset is pending, and is automatically negated when reset completes. Therefore, software can poll this bit to know when the soft reset has completed. Soft reset cannot be applied when clocks are shut down in a low power mode. The module should be first removed from low power mode, and then soft reset can be applied. This bit is not affected by soft reset. 0b - No reset request. 1b - Resets the registers affected by soft reset.
24 FRZACK	Freeze Mode Acknowledge This read-only bit indicates that FlexCAN is in Freeze mode and its prescaler is stopped. The Freeze mode request cannot be granted until current transmission or reception processes have finished. Therefore the software can poll the FRZACK bit to know when FlexCAN has actually entered Freeze mode. If Freeze mode request is negated, then this bit is negated after the FlexCAN prescaler is running again. If Freeze mode is requested when FlexCAN is in a low power mode, then the FRZACK bit will be set only when the low-power mode is exited. See Freeze mode. This bit is not affected by soft reset. **NOTE** FRZACK will be asserted within 178 CAN bits from the Freeze mode request by the CPU, and negated within 2 CAN bits after the Freeze mode request removal (see Protocol timing).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - FlexCAN not in Freeze mode, prescaler running. 1b - FlexCAN in Freeze mode, prescaler stopped.
23 —	Reserved
22 —	Reserved When writing to this field, always write the reset value.
21 WRNEN	Warning Interrupt Enable When asserted, this bit enables the generation of the TWRNINT and RWRNINT flags in the Error and Status Register 1 (ESR1). If WRNEN is negated, the TWRNINT and RWRNINT flags will always be zero, independent of the values of the error counters, and no warning interrupt will ever be generated. This bit can be written in Freeze mode only, because it is blocked by hardware in other modes. 0b - TWRNINT and RWRNINT bits are zero, independent of the values in the error counters. 1b - TWRNINT and RWRNINT bits are set when the respective error counter transitions from less than 96 to greater than or equal to 96.
20 LPMACK	Low-Power Mode Acknowledge This read-only bit indicates that FlexCAN is in a low-power mode (Disable mode). A low-power mode cannot be entered until all current transmission or reception processes have finished, so the CPU can poll the LPMACK bit to know when FlexCAN has actually entered low power mode. This bit is not affected by soft reset. **NOTE** LPMACK will be asserted within 180 CAN bits from the low-power mode request by the CPU, and negated within 2 CAN bits after the low-power mode request removal (see Protocol timing). 0b - FlexCAN is not in a low-power mode. 1b - FlexCAN is in a low-power mode.
19 —	Reserved When writing to this field, always write the reset value.
18 —	Reserved When writing to this field, always write the reset value.
17 SRXDIS	Self Reception Disable This bit defines whether FlexCAN is allowed to receive frames transmitted by itself. If this bit is asserted, frames transmitted by the module will not be stored in any MB, regardless if the MB is programmed with an ID that matches the transmitted frame, and no interrupt flag or interrupt signal will be generated due

Table continues on the next page...

Table continued from the previous page...

Field	Function
	to the frame reception. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0b - Self-reception enabled. 1b - Self-reception disabled.
16 IRMQ	Individual Rx Masking And Queue Enable This bit indicates whether Rx matching process will be based either on individual masking and queue or on masking scheme with RXMGMASK, RX14MASK, RX15MASK, and RXFGMASK. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0b - Individual Rx masking and queue feature are disabled. For backward compatibility with legacy applications, the reading of C/S word locks the MB even if it is EMPTY. 1b - Individual Rx masking and queue feature are enabled.
15 DMA	DMA Enable DMA controls whether the DMA feature is enabled or not. The DMA feature can only be used in Legacy Rx FIFO or Enhanced Rx FIFO, so consequently MCR[RFEN] or ERFCR[ERFEN] must be asserted. When DMA and RFEN are set, IFLAG1[BUF5I] generates the DMA request and no RX FIFO interrupt is generated. This bit can be written in Freeze mode only as it is blocked by hardware in other modes. 0b - DMA feature for Legacy RX FIFO or Enhanced Rx FIFO disabled. 1b - DMA feature for Legacy RX FIFO or Enhanced Rx FIFO enabled.
14 —	Reserved
13 LPRIOEN	Local Priority Enable This bit is provided for backwards compatibility with legacy applications. It controls whether the local priority feature is enabled or not. It is used to expand the ID used during the arbitration process. With this expanded ID concept, the arbitration process is done based on the full 32-bit word, but the actual transmitted ID still has 11-bit for standard frames and 29-bit for extended frames. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0b - Local Priority disabled. 1b - Local Priority enabled.
12 AEN	Abort Enable When asserted, this bit enables the Tx abort mechanism. This mechanism guarantees a safe procedure for aborting a pending transmission, so that no frame is sent in the CAN bus without notification. This bit can be written only in Freeze mode because it is blocked by hardware in other modes.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE When MCR[AEN] is asserted, only the abort mechanism (see Transmission abort mechanism) must be used for updating mailboxes configured for transmission. **CAUTION** Writing the Abort code into Rx mailboxes can cause unpredictable results when MCR[AEN] is asserted. 0b - Abort disabled. 1b - Abort enabled.
11 FDEN	CAN FD operation enable This bit enables the CAN with Flexible Data rate (CAN FD) operation. This bit can be written in Freeze mode only. **NOTE** FlexCAN is able to transmit FD frame format according to ISO 11898-1. **NOTE** The Legacy Rx FIFO Enable (RFEN) bit cannot be set if FDEN is asserted. 0b - CAN FD is disabled. FlexCAN is able to receive and transmit messages in CAN 2.0 format. 1b - CAN FD is enabled. FlexCAN is able to receive and transmit messages in both CAN FD and CAN 2.0 formats.
10 —	Reserved
9-8 IDAM	ID Acceptance Mode This 2-bit field identifies the format of the Legacy Rx FIFO ID filter table elements. Note that all elements of the table are configured at the same time by this field (they are all the same format). See Legacy Rx FIFO structure. This field can be written only in Freeze mode because it is blocked by hardware in other modes. 00b - Format A: One full ID (standard and extended) per ID filter table element. 01b - Format B: Two full standard IDs or two partial 14-bit (standard and extended) IDs per ID filter table element. 10b - Format C: Four partial 8-bit standard IDs per ID filter table element. 11b - Format D: All frames rejected.
7 —	Reserved
6-0 MAXMB	Number Of The Last Message Buffer

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This 7-bit field defines the number of the last message buffers that will take part in the matching and arbitration processes. The reset value (0Fh) is equivalent to a 16 MB configuration. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Number of the last MB = MAXMB **NOTE** MAXMB must be programmed with a value smaller than or equal to the number of available message buffers, as described in FlexCAN memory partition for CAN FD. Additionally, the definition of MAXMB value must take into account the region of MBs occupied by Legacy Rx FIFO and its ID filters table space defined by CTRL2[RFFN]. MAXMB also impacts the definition of the minimum number of peripheral clocks per CAN bit as described in Table 380.

50.4.2.3 Control 1 register (CTRL1)

Offset

Register	Offset
CTRL1	4h

Function

This register is defined for specific FlexCAN control features related to the CAN bus, such as bit rate, programmable sampling point within an Rx bit, Loop Back mode, Listen-Only mode, Bus Off recovery behavior, and interrupt enabling (Bus-Off, Error, Warning). It also determines the division factor for the clock prescaler.

The CAN bit timing variables (PRESDIV, PROPSEG, PSEG1, PSEG2, and RJW) can also be configured in the CBT register, which extends the range of all these variables. If CBT[BTF] is set, PRESDIV, PROPSEG, PSEG1, PSEG2, and RJW fields of CTRL1 become read only.

If CTRL2[BTE] is set, then the PRESDIV, PROPSEG, PSEG1, PSEG2, and RJW fields in the CTRL1 register are not used by the hardware, are read as zero, and a write operation to them has no effect.

NOTE

When the CAN FD feature is enabled, do not use the PRESDIV, RJW, PSEG1, PSEG2, and PROPSEG fields of the CTRL1 register for CAN bit timing. Instead use the CBT register's EPRESDIV, ERJW, EPSEG1, EPSEG2, and EPROPSEG fields.

The contents of this register are not affected by soft reset.

NOTE

The CAN bit variables in CTRL1 and in CBT are stored in the same register.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CTI Futong Automotive Technology Co., Ltd.
c8c462e3-45f1-4188-9251-390d3a0e04d1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	PRESDIV								RJW		PSEG1			PSEG2		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BOFF MSK	ERRM SK	Reserv ed	LPB	TWRN MSK	RWRN MSK	Reserv ed	Reserv ed	SMP	BOFF REC	TSYN	LBUF	LOM	PROPSEG		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 PRESDIV	Prescaler Division Factor This 8-bit field defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency. The maximum value of this field is FFh, which gives a minimum Sclock frequency equal to the PE clock frequency divided by 256. See Protocol timing for more information. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Sclock frequency = PE clock frequency / (PRESDIV + 1).
23-22 RJW	Resync Jump Width This 2-bit field defines the maximum number of time quanta that a bit time can be changed by one resynchronization. One time quantum is equal to the Sclock period. The valid programmable values are 0–3. See Protocol timing for more information. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Resync Jump Width = RJW + 1.
21-19 PSEG1	Phase Segment 1 This 3-bit field defines the length of phase segment 1 in the bit time. The valid programmable values are 0–7. See Protocol timing for more information. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 1 = (PSEG1 + 1) × Time-Quanta.
18-16 PSEG2	Phase Segment 2 This 3-bit field defines the length of phase segment 2 in the bit time. The valid programmable values are 1–7. See Protocol timing for more information. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 2 = (PSEG2 + 1) × Time-Quanta.
15 BOFFMSK	Bus Off Interrupt Mask This bit provides a mask for the Bus Off interrupt ESR1[BOFFINT].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Bus Off interrupt disabled. 1b - Bus Off interrupt enabled.
14 ERRMSK	Error Interrupt Mask This bit provides a mask for the Error interrupt ESR1[ERRINT]. 0b - Error interrupt disabled. 1b - Error interrupt enabled.
13 —	Reserved **NOTE** This field is writeable only if the module is disabled. Otherwise the access type is read-only.
12 LPB	Loop Back Mode This bit configures FlexCAN to operate in Loop-Back mode. In this mode, FlexCAN performs an internal loop back that can be used for self-test operation. The bit stream output of the transmitter is fed back internally to the receiver input. The Rx CAN input pin is ignored and the Tx CAN output goes to the recessive state (logic 1). FlexCAN behaves as it normally does when transmitting, and treats its own transmitted message as a message received from a remote node. In this mode, FlexCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field, generating an internal acknowledge bit to ensure proper reception of its own message. Both transmit and receive interrupts are generated. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. **NOTE** In this mode, MCR[SRXDIS] cannot be asserted because this will impede the self-reception of a transmitted message. **NOTE** FDCTRL[TDCEN] and ETDC[ETDCEN] must be disabled when LPB is asserted. 0b - Loop Back disabled. 1b - Loop Back enabled.
11 TWRNMSK	Tx Warning Interrupt Mask This bit provides a mask for the Tx Warning interrupt associated with the TWRNINT flag in the Error and Status Register 1 (ESR1). This bit is read as zero when MCR[WRNEN] is negated. This bit can be written only if MCR[WRNEN] is asserted. 0b - Tx Warning interrupt disabled. 1b - Tx Warning interrupt enabled.
10 RWRNMSK	Rx Warning Interrupt Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit provides a mask for the Rx Warning interrupt associated with the RWRNINT flag in the Error and Status Register 1 (ESR1). This bit is read as zero when MCR[WRNEN] bit is negated. This bit can be written only if MCR[WRNEN] bit is asserted. 0b - Rx Warning interrupt disabled. 1b - Rx Warning interrupt enabled.
9 —	Reserved
8 —	Reserved
7 SMP	CAN Bit Sampling This bit defines the sampling mode of CAN bits at the Rx input. It can be written in Freeze mode only, because it is blocked by hardware in other modes. **NOTE** For proper operation, to assert SMP it is necessary to guarantee a minimum value of two TQs in CTRL1[PSEG1] (or CBT[EPSEG1]).This bit cannot be asserted when CAN FD is enabled (MCR[FDEN] = 1). 0b - Just one sample is used to determine the bit value. 1b - Three samples are used to determine the value of the received bit: the regular one (sample point) and two preceding samples; a majority rule is used.
6 BOFFREC	Bus Off Recovery This bit defines how FlexCAN recovers from Bus Off state. If this bit is negated, automatic recovering from Bus Off state occurs according to the CAN Specification 2.0B. If the bit is asserted, automatic recovering from Bus Off is disabled and the module remains in Bus Off state until the bit is negated by the user. If the negation occurs before 128 sequences of 11 recessive bits are detected on the CAN bus, then Bus Off recovery happens as if the BOFFREC bit had never been asserted. If the negation occurs after 128 sequences of 11 recessive bits occurred, then FlexCAN will resynchronize to the bus by waiting for 11 recessive bits before joining the bus. After negation, the BOFFREC bit can be reasserted again during Bus Off, but it will be effective only the next time the module enters Bus Off. If BOFFREC was negated when the module entered Bus Off, asserting it during Bus Off will not be effective for the current Bus Off recovery. **NOTE** See Bus off in the CAN Protocol standard (ISO 11898-1) for details. 0b - Automatic recovering from Bus Off state enabled. 1b - Automatic recovering from Bus Off state disabled.
5 TSYN	Timer Sync

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This bit enables a mechanism that resets the free running timer each time a message is received in message buffer 0. This feature provides means to synchronize multiple FlexCAN stations with a special “SYNC” message, that is, global network time. If MCR[RFEN] is set (Legacy Rx FIFO enabled), the first available mailbox, according to CTRL2[RFFN] setting, is used for timer synchronization instead of MB0. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0b - Timer sync feature disabled 1b - Timer sync feature enabled
4 LBUF	Lowest Buffer Transmitted First This bit defines the ordering mechanism for message buffer transmission. When asserted, MCR[LPRIOEN] does not affect the priority arbitration. This bit can be written in Freeze mode only, because it is blocked by hardware in other modes. 0b - Buffer with highest priority is transmitted first. 1b - Lowest number buffer is transmitted first.
3 LOM	Listen-Only Mode This bit configures FlexCAN to operate in Listen-Only mode. In this mode, transmission is disabled, all error counters described in the ECR register are frozen, and the module operates in a CAN Error Passive mode. Only messages acknowledged by another CAN station will be received. If FlexCAN detects a message that has not been acknowledged, it will flag a BIT0 error without changing the receive error counter (ECR[RXERRCNT]), as if it was trying to acknowledge the message. Listen-Only mode is acknowledged by the state of ESR1[FLTCONF] indicating Passive Error. There can be some delay between the Listen-Only mode request and acknowledge. This bit can be written in Freeze mode only because it is blocked by hardware in other modes. 0b - Listen-Only mode is deactivated. 1b - FlexCAN module operates in Listen-Only mode.
2-0 PROPSEG	Propagation Segment This 3-bit field defines the length of the propagation segment in the bit time. The valid programmable values are 0–7. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Propagation segment time = (PROPSEG + 1) × time-quanta. Time-quantum = one Sclock period.

50.4.2.4 Free Running Timer (TIMER)

Offset

Register	Offset
TIMER	8h

Function

This register represents a 16-bit free-running counter that can be read and written by the CPU. The timer starts from 0h after Reset, counts linearly to FFFFh, and wraps around.

When CTRL2[TIMER_SRC] is asserted, the timer is continuously incremented by an external time tick. The time tick must be synchronous to the peripheral clock, with a minimum pulse width of one clock cycle.

When CTRL2[TIMER_SRC] is negated, the timer is incremented by the CAN bit clock, which defines the baud rate on the CAN bus. During a message transmission/reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate. The timer is not incremented during Disable and Freeze modes.

The timer value is captured when the second bit of the identifier field of any frame is on the CAN bus. This captured value is written into the time stamp entry in a message buffer after a successful reception or transmission of a message.

If CTRL1[TSYN] is asserted, the timer is reset whenever a message is received in the first available mailbox, according to CTRL2[RFFN] setting.

The CPU can write to this register anytime. However, if the write occurs at the same time that the timer is being reset by a reception in the first mailbox, then the write value is discarded.

Reading this register affects the mailbox unlocking procedure (see Mailbox lock mechanism).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TIMER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 TIMER	Timer Value Contains the free-running counter value.

50.4.2.5 Rx Mailboxes Global Mask register (RXMGMASK)

Offset

Register	Offset
RXMGMASK	10h

Function

This register is located in RAM.

RXMGMASK is provided for legacy application support.

- When MCR[IRMQ] is negated, RXMGMASK is always in effect (the bits in the MG field will mask the mailbox filter bits).
- When MCR[IRMQ] is asserted, RXMGMASK has no effect (the bits in the MG field will not mask the mailbox filter bits).

RXMGMASK is used to mask the filter fields of all Rx MBs, excluding MBs 14-15, which have individual mask registers.

This register can only be written in Freeze mode as it is blocked by hardware in other modes.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MG															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MG															
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-0 MG	Rx Mailboxes Global Mask Bits These bits mask the mailbox filter bits. Note that the alignment with the ID word of the mailbox is not perfect as the two most significant MG bits affect the fields RTR and IDE, which are located in the Control and Status word of the mailbox. The following table shows in detail which MG bits mask each mailbox filter field.

SMB[RTR] [1]	CTRL2[RRS]	CTRL2[EACEN]	Mailbox filter fields			
			MB[RTR]	MB[IDE]	MB[ID]	Reserved
0	—	0	note [2]	note [3]	MG[28:0]	MG[31:29]
0	—	1	MG[31]	MG[30]	MG[28:0]	MG[29]
1	0	—	—	—	—	MG[31:0]
1	1	0	—	—	MG[28:0]	MG[31:29]
1	1	1	MG[31]	MG[30]	MG[28:0]	MG[29]

1. RTR bit of the incoming frame. It is saved into an auxiliary MB called Rx serial message buffer (Rx SMB).
2. If CTRL2[EACEN] is negated, the RTR bit of mailbox is never compared with the RTR bit of the incoming frame.
3. If CTRL2[EACEN] is negated, the IDE bit of mailbox is always compared with the IDE bit of the incoming frame.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHIPSAILING Automotive Technology Co., Ltd.
cac02c5482134e18b251-390d3ab624c1

Field	Function
	0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

50.4.2.6 Rx 14 Mask register (RX14MASK)

Offset

Register	Offset
RX14MASK	14h

Function

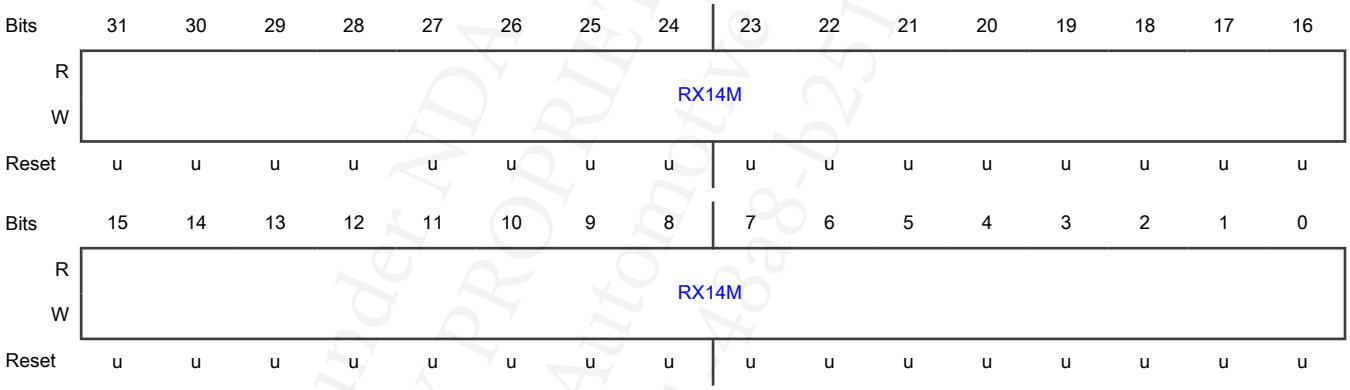
This register is located in RAM.

RX14MASK is provided for legacy application support. When MCR[IRMQ] is asserted, RX14MASK has no effect.

RX14MASK is used to mask the filter fields of message buffer 14.

This register can only be programmed when the module is in Freeze mode as it is blocked by hardware in other modes.

Diagram



Fields

Field	Function
31-0 RX14M	Rx Buffer 14 Mask Bits Each mask bit masks the corresponding mailbox 14 filter field in the same way that RXMGMASK masks other mailboxes' filters. See the description of the RXMGMASK register. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

Printed under NDA only
COMPANY PROPRIETARY
Bejing CHJ Automotive Technology Co., Ltd.
c8c0c25-4821-48a8-b251-390c3ab624c1

50.4.2.7 Rx 15 Mask register (RX15MASK)

Offset

Register	Offset
RX15MASK	18h

Function

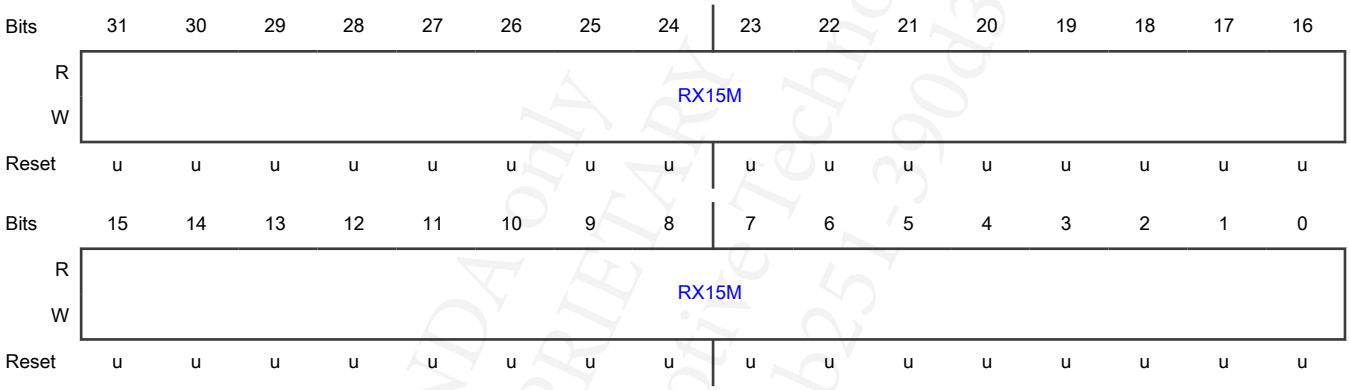
This register is located in RAM.

RX15MASK is provided for legacy application support. When MCR[IRMQ] is asserted, RX15MASK has no effect.

RX15MASK is used to mask the filter fields of message buffer 15.

This register can be programmed only when the module is in Freeze mode because it is blocked by hardware in other modes.

Diagram



Fields

Field	Function
31-0 RX15M	Rx Buffer 15 Mask Bits Each mask bit masks the corresponding mailbox 15 filter field in the same way that RXMGMASK masks other mailboxes' filters. See the description of the RXMGMASK register. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

50.4.2.8 Error Counter (ECR)

Offset

Register	Offset
ECR	1Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4c2-48a8-b251-290d3ab621c1

Function

This register has four 8-bit fields reflecting the value of the FlexCAN error counters:

- Transmit Error Counter (TXERRCNT field)
- Receive Error Counter (RXERRCNT field)
- Transmit Error Counter for errors detected in the data phase of CAN FD messages with the BRS bit set (TXERRCNT_FAST field)
- Receive Error Counter for errors detected in the data phase of CAN FD messages with the BRS bit set (RXERRCNT_FAST field)

The TXERRCNT and RXERRCNT counters take into account all errors in both CAN FD and non-FD message formats. TXERRCNT_FAST and RXERRCNT_FAST are dedicated to count only the errors that occur in the data phase of CAN FD frames with the BRS bit set.

The Fault Confinement state (ESR1[FLTCONF]) is updated based on TXERRCNT and RXERRCNT counters only. TXERRCNT and RXERRCNT counters can be written in Freeze mode only. TXERRCNT_FAST and RXERRCNT_FAST counters are read-only except in Freeze mode when the CPU can write value zero. The rules for increasing and decreasing these counters are described in the CAN protocol and are completely implemented in the FlexCAN module.

The following are the basic rules for FlexCAN bus state transitions:

- If the value of TXERRCNT or RXERRCNT becomes greater than or equal to 128, ESR1[FLTCONF] is updated to reflect Error Passive state.
- If the FlexCAN state is Error Passive, and either TXERRCNT or RXERRCNT decrements to a value less than or equal to 127 when the other already satisfies this condition, ESR1[FLTCONF] is updated to reflect Error Active state.
- If the value of TXERRCNT becomes greater than 255, ESR1[FLTCONF] is updated to reflect Bus Off state, and an interrupt may be issued. The value of TXERRCNT is then reset to zero.
- If FlexCAN is in Bus Off state, then TXERRCNT is cascaded together with another internal counter to count the 128th occurrences of 11 consecutive recessive bits on the bus. Hence, TXERRCNT is reset to zero and counts in a manner where the internal counter counts 11 such bits and then wraps around when incrementing the TXERRCNT. When TXERRCNT reaches the value of 128, ESR1[FLTCONF] is updated to be Error Active and both error counters are reset to zero. At any instance of dominant bit following a stream of less than 11 consecutive recessive bits, the internal counter resets itself to zero without affecting the TXERRCNT value. The TXERRCNT_FAST counter is frozen during Bus Off.
- If during system startup only one node is operating, then its TXERRCNT increases in each message it is trying to transmit, as a result of acknowledge errors (indicated by the ACKERR bit in the Error and Status Register). After the transition to Error Passive state, TXERRCNT does not increment anymore by acknowledge errors. Therefore the device never goes to the Bus Off state.
- If RXERRCNT increases to a value greater than 127, it is not incremented further, even if more errors are detected when being a receiver. At the next successful message reception, the counter is set to a value between 119 and 127 to return to Error Active state.
- TXERRCNT_FAST and RXERRCNT_FAST error counter values increment and decrement based on errors detected only in the data phase of CAN FD frames with the BRS bit set, following the same increment and decrement rules as TXERRCNT and RXERRCNT counters. These counters do not wrap around and get stuck at their maximum value (255). They stop counting and keep their values frozen when FlexCAN is in Bus Off state. They are reset when FlexCAN leaves Bus Off state and restart counting after FlexCAN returns to Error Active state.

NOTE

See Fault confinement in the CAN Protocol standard (ISO 11898-1) for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RXERRCNT_FAST								TXERRCNT_FAST							
W	0								0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXERRCNT								TXERRCNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 RXERRCNT_FAST	Receive Error Counter for fast bits Receive error counter for errors detected in the data phase of received CAN FD messages with the BRS bit set. The RXERRCNT_FAST counter is read-only except in Freeze mode, where the CPU can write an 8-bit zero value only.
23-16 TXERRCNT_FAST	Transmit Error Counter for fast bits Transmit error counter for errors detected in the data phase of transmitted CAN FD messages with the BRS bit set. The TXERRCNT_FAST counter is read-only except in Freeze mode, where the CPU can write an 8-bit zero value only.
15-8 RXERRCNT	Receive Error Counter Receive error counter for all errors detected in received messages. The RXERRCNT counter is read-only except in Freeze mode, where it can be written by the CPU.
7-0 TXERRCNT	Transmit Error Counter Transmit error counter for all errors detected in transmitted messages. The TXERRCNT counter is read-only except in Freeze mode, where it can be written by the CPU.

50.4.2.9 Error and Status 1 register (ESR1)

Offset

Register	Offset
ESR1	20h

Function

This register reports various error conditions detected in the reception and transmission of a CAN frame, some general status of the device, and is the source of some interrupts to the CPU. The reported error conditions are:

- Errors detected in CAN frames of any format:
  - BIT1ERR

  - BIT0ERR
  - ACKKERR
  - CRCERR
  - FRMERR
  - STFERR
- Errors detected in the data phase of CAN FD frames with the BRS bit set only:
  - BIT1ERR_FAST
  - BIT0ERR_FAST
  - CRCERR_FAST
  - FRMERR_FAST
  - STFERR_FAST

An error detected in a single CAN frame may be reported by one or more error flags. Also, error reporting is cumulative, in case more error events happen in the next frames when the CPU does not attempt to read this register.

Status bits:

- TXWRN
- RXWRN
- IDLE
- TX
- FLTCONF
- RX
- SYNCH

Interrupt bits:

- BOFFINT
- BOFFDONEINT
- ERRINT
- ERRINT_FAST
- TWRNINT
- RWRNINT

It is recommended that the CPU use the following procedure when servicing interrupt requests generated by these bits:

1. Read this register to capture all error condition and status bits. This action clears the respective bits that were set since the last read access.
2. Write 1 to clear the interrupt bit that has triggered the interrupt request.
3. Write 1 to clear the ERR_OVR bit, if it is set.

Starting from all error flags cleared, a first error event sets either the ERRINT or the ERRINT_FAST (provided the corresponding mask bit is asserted). If other error events in subsequent frames happen before the CPU serves the interrupt request, the ERR_OVR bit is set to indicate that errors from different frames have accumulated.

SYNCH	IDLE	TX	RX	FlexCAN state

Table continues on the next page...

Table continued from the previous page...

0	0	0	0	Not synchronized to CAN bus
1	1	x	x	Idle
1	0	1	0	Transmitting
1	0	0	1	Receiving

NOTE

See Fault confinement in the CAN Protocol standard (ISO 11898-1) for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BIT1ERR...	BIT0ERR...	0	CRCERR_...	FRMERR_...	STFERR_...	0				ERROVR	ERRINT_...	BOFFDON...	SYNCH	TWRNINT	RWRNINT
W											W1C	W1C	W1C		W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BIT1ERR	BIT0ERR	ACKERR	CRCERR	FRMERR	STFERR	TXWRN	RXWRN	IDLE	TX	FLTCONF		RX	BOFFINT	ERRINT	0
W														W1C	W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BIT1ERR_FAST	Bit1 Error in the Data Phase of CAN FD frames with the BRS bit set This bit indicates when an inconsistency occurs between the transmitted and the received bit in the data phase of CAN FD frames with the BRS bit set. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit sent as recessive is received as dominant.
30 BIT0ERR_FAST	Bit0 Error in the Data Phase of CAN FD frames with the BRS bit set This bit indicates when an inconsistency occurs between the transmitted and the received bit in the data phase of CAN FD frames with the BRS bit set. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit sent as dominant is received as recessive.
29	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
28 CRCERR_FAST	Cyclic Redundancy Check Error in the CRC field of CAN FD frames with the BRS bit set This bit indicates that a CRC error has been detected by the receiver node in the CRC field of CAN FD frames with the BRS bit set, that is, the calculated CRC is different from the received. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A CRC error occurred since last read of this register.
27 FRMERR_FAST	Form Error in the Data Phase of CAN FD frames with the BRS bit set This bit indicates that a form error has been detected by the receiver node in the data phase of CAN FD frames with the BRS bit set, that is, a fixed-form bit field contains at least one illegal bit. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A form error occurred since last read of this register.
26 STFERR_FAST	Stuffing Error in the Data Phase of CAN FD frames with the BRS bit set This bit indicates that a stuffing error has been detected in the data phase of CAN FD frames with the BRS bit set. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A stuffing error occurred since last read of this register.
25-22 —	Reserved
21 ERROVR	Error Overrun This bit indicates that an error condition occurred when any error flag is already set. This bit is cleared by writing it to 1. 0b - Overrun has not occurred. 1b - Overrun has occurred.
20 ERRINT_FAST	Error interrupt for errors detected in Data Phase of CAN FD frames with BRS bit set This bit indicates that at least one of the error bits detected in the data phase of CAN FD frames with the BRS bit set (BIT1ERR_FAST, BIT0ERR_FAST, CRCERR_FAST, FRMERR_FAST, or STFERR_FAST) is set. If the corresponding mask bit CTRL2[ERRMSK_FAST] is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0b - No such occurrence. 1b - Indicates setting of any error bit detected in the data phase of CAN FD frames with the BRS bit set.

Table continues on the next page...

Table continued from the previous page...

Field	Function
19 BOFFDONEINT	Bus Off Done Interrupt This bit is set when the Tx Error Counter (TXERRCNT) has finished counting 128 occurrences of 11 consecutive recessive bits on the CAN bus and is ready to leave Bus Off. If the corresponding mask bit in the Control 2 Register (BOFFDONEMSK) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0b - No such occurrence. 1b - FlexCAN module has completed Bus Off process.
18 SYNCH	CAN Synchronization Status This read-only flag indicates whether the FlexCAN is synchronized to the CAN bus and able to participate in the communication process. It is set and cleared by the FlexCAN. See the table in the overall ESR1 register description. 0b - FlexCAN is not synchronized to the CAN bus. 1b - FlexCAN is synchronized to the CAN bus.
17 TWRNINT	Tx Warning Interrupt Flag If MCR[WRNEN] is asserted, TWRNINT is set when the TXWRN flag transitions from 0 to 1, meaning that the Tx error counter reached 96. If the corresponding mask bit in the Control 1 Register (CTRL1[TWRNMSK]) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. When MCR[WRNEN] is negated, this flag is masked. CPU must clear this flag before disabling MCR[WRNEN]. Otherwise it will be set when MCR[WRNEN] is set again. Writing 0 has no effect. This flag is not generated during Bus Off state. This bit is not updated during Freeze mode. 0b - No such occurrence. 1b - The Tx error counter transitioned from less than 96 to greater than or equal to 96.
16 RWRNINT	Rx Warning Interrupt Flag If MCR[WRNEN] is asserted, RWRNINT is set when the RXWRN flag transitions from 0 to 1, meaning that the Rx error counters reached 96. If the corresponding mask bit in the Control 1 Register (CTRL1[RWRNMSK]) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. When WRNEN is negated, this flag is masked. CPU must clear this flag before disabling the bit. Otherwise it will be set when WRNEN is set again. Writing 0 has no effect. This bit is not updated during Freeze mode. 0b - No such occurrence. 1b - The Rx error counter transitioned from less than 96 to greater than or equal to 96.
15 BIT1ERR	Bit1 Error This bit indicates when an inconsistency occurs between the transmitted and the received bit in a non-CAN FD message or in the arbitration or data phase of a CAN FD message. **NOTE** This bit is not set by a transmitter in case of arbitration field or ACK slot, or in case of a node sending a passive error flag that detects dominant bits.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit sent as recessive is received as dominant.
14 BIT0ERR	Bit0 Error This bit indicates when an inconsistency occurs between the transmitted and the received bit in a non-CAN FD message or in the arbitration or data phase of a CAN FD message. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - At least one bit sent as dominant is received as recessive.
13 ACKERR	Acknowledge Error This bit indicates that an Acknowledge Error has been detected by the transmitter node, that is, a dominant bit has not been detected during the ACK SLOT. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - An ACK error occurred since last read of this register.
12 CRCERR	Cyclic Redundancy Check Error This bit indicates that a Cyclic Redundancy Check (CRC) error has been detected by the receiver node either in a non-FD message or in the arbitration or data phase of a frame in CAN FD format, that is, the calculated CRC is different from the received. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A CRC error occurred since last read of this register.
11 FRMERR	Form Error This bit indicates that a form error has been detected in a non-FD message or else in an FD message's arbitration or data phase by the receiver node, that is, a fixed-form bit field contains at least one illegal bit. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A Form Error occurred since last read of this register.
10 STFERR	Stuffing Error This bit indicates that a stuffing error has been detected in a non-FD message or else in an FD message's arbitration or data phase by the receiver node. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - A stuffing error occurred since last read of this register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 TXWRN	TX Error Warning This bit indicates when repetitive errors are occurring during message transmission and is affected by the value of TXERRCNT in ECR register only. This bit is not updated during Freeze mode. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - TXERRCNT is greater than or equal to 96.
8 RXWRN	Rx Error Warning This bit indicates when repetitive errors are occurring during message reception and is affected by the value of RXERRCNT in ECR register only. This bit is not updated during Freeze mode. After a read operation, the field's value clears to 0. 0b - No such occurrence. 1b - RXERRCNT is greater than or equal to 96.
7 IDLE	IDLE This bit indicates when CAN bus is in IDLE state. See the table in the overall ESR1 register description. 0b - No such occurrence. 1b - CAN bus is now IDLE.
6 TX	FlexCAN In Transmission This bit indicates if FlexCAN is transmitting a message. See the table in the overall ESR1 register description. 0b - FlexCAN is not transmitting a message. 1b - FlexCAN is transmitting a message.
5-4 FLTCONF	Fault Confinement State This 2-bit field indicates the confinement state of the FlexCAN module. If CTRL1[LOM] is asserted, after a delay that depends on the CAN bit timing, ESR1[FLTCONF] will indicate Error Passive. The very same delay affects the way that ESR1[FLTCONF] reflects an update to ECR register by the CPU. It may be necessary to wait up to one CAN bit time to get them coherent again. This bit field is affected by soft reset, but if the LOM bit is asserted, its reset value lasts just one CAN bit. After this time, ESR1[FLTCONF] reports Error Passive. 00b - Error Active 01b - Error Passive 1xb - Bus Off
3 RX	FlexCAN In Reception This bit indicates if FlexCAN is receiving a message. See the table in the overall ESR1 register description.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - FlexCAN is not receiving a message. 1b - FlexCAN is receiving a message.
2 BOFFINT	Bus Off Interrupt This bit is set when FlexCAN enters Bus Off state. If the corresponding mask bit in the Control Register 1 (CTRL1[BOFFMSK]) is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0b - No such occurrence. 1b - FlexCAN module entered Bus Off state.
1 ERRINT	Error Interrupt This bit indicates that at least one of the error bits (BIT1ERR, BIT0ERR, ACKERR, CRCERR, FRMERR, or STFERR) is set. If the corresponding mask bit CTRL1[ERRMSK] is set, an interrupt is generated to the CPU. This bit is cleared by writing it to 1. Writing 0 has no effect. 0b - No such occurrence. 1b - Indicates setting of any error bit in the Error and Status register.
0 —	Reserved

50.4.2.10 Interrupt Masks 2 register (IMASK2)

Offset

Register	Offset
IMASK2	24h

Function

This register allows any number of a range of the 32 message buffer interrupts to be enabled or disabled for MB63 to MB32. It contains one interrupt mask bit per buffer, enabling the CPU to determine which buffer generates an interrupt after a successful transmission or reception, that is, when the corresponding IFLAG2 bit is set.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CVT Automotive Technology Co., Ltd.
c8c462e5-4201-48a8-b251-3900a5b624d1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BUF63TO32M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BUF63TO32M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BUF63TO32M	Buffer MBi Mask Each bit enables or disables the corresponding FlexCAN message buffer interrupt for MB63 to MB32. **NOTE** Setting or clearing a bit in the IMASK2 Register can assert or negate an interrupt request, if the corresponding IFLAG2 bit is set. 0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

50.4.2.11 Interrupt Masks 1 register (IMASK1)

Offset

Register	Offset
IMASK1	28h

Function

This register allows any number of a range of the 32 message buffer interrupts to be enabled or disabled for MB31 to MB0. It contains one interrupt mask bit per buffer, enabling the CPU to determine which buffer generates an interrupt after a successful transmission or reception, that is, when the corresponding IFLAG1 bit is set.

Provided under IDA Only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co.,Ltd.
c8c462e5-821-48a8-b251-90d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BUF31TO0M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BUF31TO0M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BUF31TO0M	Buffer MBi Mask Each bit enables or disables the corresponding FlexCAN message buffer interrupt for MB31 to MB0. **NOTE** Setting or clearing a bit in the IMASK1 Register can assert or negate an interrupt request, if the corresponding IFLAG1 bit is set. 0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

50.4.2.12 Interrupt Flags 2 register (IFLAG2)

Offset

Register	Offset
IFLAG2	2Ch

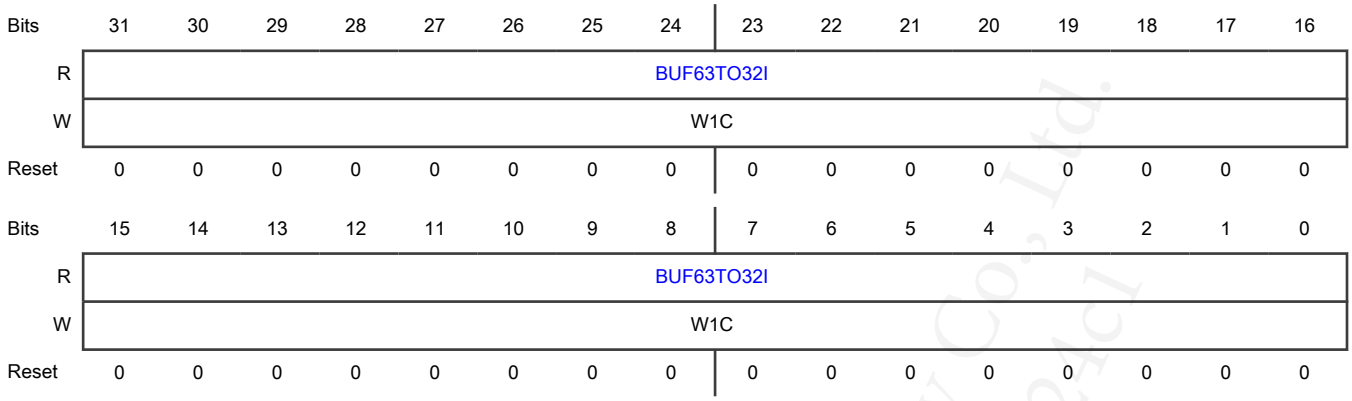
Function

This register defines the flags for the 32 message buffer interrupts for MB63 to MB32. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the respective IFLAG2 bit. If the corresponding IMASK2 bit is set, an interrupt will be generated. The interrupt flag must be cleared by writing 1 to it. Writing 0 has no effect.

Before updating MCR[MAXMB], CPU must service the IFLAG2 bits whose MB value is greater than the MAXMB to be updated; otherwise, they will remain set and be inconsistent with the number of MBs available.

Preliminary Information
COMPANY CONFIDENTIAL
Beijing eSOL Automotive Technology Co., Ltd.
c8c46250-821-48a8-b251-90d3ab624c1

Diagram



Fields

Field	Function
31-0 BUF63TO32I	Buffer MBi Interrupt Each bit flags the corresponding FlexCAN message buffer interrupt for MB63 to MB32. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.

50.4.2.13 Interrupt Flags 1 register (IFLAG1)

Offset

Register	Offset
IFLAG1	30h

Function

This register defines the flags for the 32 message buffer interrupts for MB31 to MB0. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding IFLAG1 bit. If the corresponding IMASK1 bit is set, an interrupt will be generated. The interrupt flag must be cleared by writing 1 to it. Writing 0 has no effect. There is an exception when DMA for Legacy Rx FIFO is enabled, as described below.

The BUF7I to BUF5I flags are also used to represent Legacy FIFO interrupts when the Legacy Rx FIFO is enabled. When MCR[RFEN] is set and MCR[DMA] is negated, the function of the eight least significant interrupt flags changes: BUF7I, BUF6I, and BUF5I indicate operating conditions of the Legacy FIFO , BUF0I is used to empty Legacy FIFO, and BUF4I to BUF1I bits are reserved.

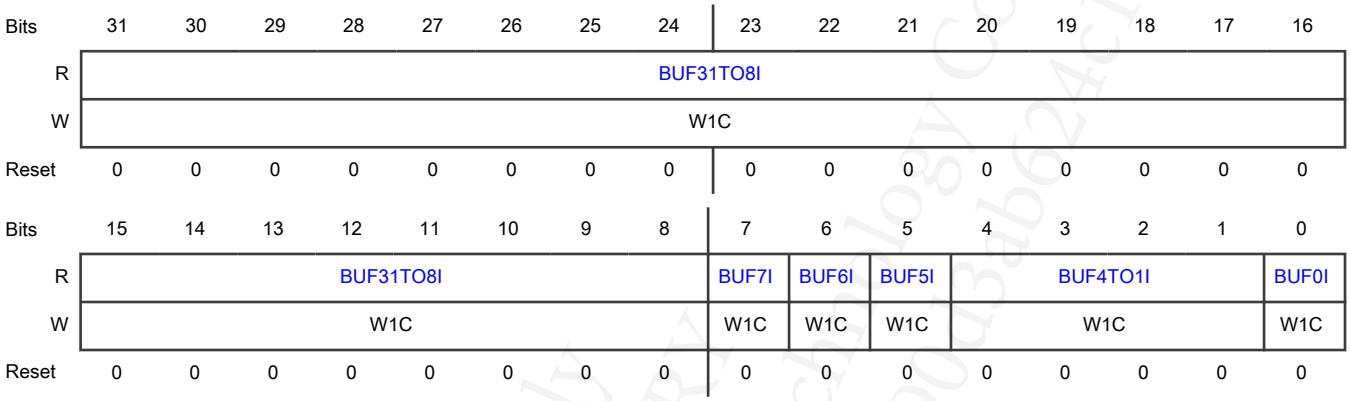
Before enabling MCR[RFEN], the CPU must service the IFLAG bits asserted in the Legacy Rx FIFO region; see Legacy Rx FIFO. Otherwise, these IFLAG bits will mistakenly show the related MBs now belonging to Legacy FIFO as having contents to be serviced. When MCR[RFEN] is negated, the Legacy FIFO flags must be cleared. The same care must be taken when a CTRL2[RFFN] value is selected, extending Legacy Rx FIFO filters beyond MB7. For example, when RFFN is 8h, the MB0–23 range is occupied by Legacy Rx FIFO filters and related IFLAG bits must be cleared.

When both the MCR[RFEN] and MCR[DMA] bits are asserted (DMA feature for Legacy Rx FIFO enabled), the function of the eight least significant interrupt flags (BUF7I–BUF0I) are changed to support the DMA operation. BUF7I and BUF6I are not

used, as well as BUF4I–BUF1I. BUF5I indicates operating condition of Legacy FIFO, and BUF0I is used to empty Legacy FIFO. Moreover, BUF5I does not generate a CPU interrupt, but generates a DMA request. IMASK1 bits in Legacy Rx FIFO region are not considered when bit MCR[DMA] is enabled. In addition the CPU must not clear the flag BUF5I when DMA is enabled. Before enabling MCR[DMA], the CPU must service the IFLAGs asserted in the Legacy Rx FIFO region. When MCR[DMA] is negated, the Legacy FIFO must be empty. Legacy FIFO must be disabled when MCR[FDEN] is enabled.

Before updating MCR[MAXMB], CPU must service the IFLAG1 bits whose MB value is greater than the MCR[MAXMB] to be updated; otherwise, they will remain set and be inconsistent with the number of MBs available.

Diagram



Fields

Field	Function
31-8 BUF31TO8I	Buffer MBi Interrupt Each bit flags the corresponding FlexCAN message buffer interrupt for MB31 to MB8. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.
7 BUF7I	Buffer MB7 Interrupt Or Legacy Rx FIFO Overflow When MCR[RFEN] is cleared (Legacy Rx FIFO disabled), this bit flags the interrupt for MB7. **NOTE** This flag is cleared by the FlexCAN whenever MCR[RFEN] is changed by CPU writes. The BUF7I flag represents Legacy Rx FIFO overflow when MCR[RFEN] is set. In this case, the flag indicates that a message was lost because the Legacy Rx FIFO is full. Note that the flag will not be asserted when the Legacy Rx FIFO is full and the message was captured by a mailbox. 0b - No occurrence of MB7 completing transmission/reception when MCR[RFEN]=0, or of Legacy Rx FIFO overflow when MCR[RFEN]=1 1b - MB7 completed transmission/reception when MCR[RFEN]=0, or Legacy Rx FIFO overflow when MCR[RFEN]=1
6 BUF6I	Buffer MB6 Interrupt Or Legacy Rx FIFO Warning

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When MCR[RFEN] is cleared (Legacy Rx FIFO disabled), this bit flags the interrupt for MB6. **NOTE** This flag is cleared by the FlexCAN whenever MCR[RFEN] is changed by CPU writes. The BUF6I flag represents Legacy Rx FIFO warning when MCR[RFEN] is set. In this case, the flag indicates when the number of unread messages within the Legacy Rx FIFO is increased to five from four due to the reception of a new one, meaning that the Legacy Rx FIFO is almost full. Note that if the flag is cleared when the number of unread messages is greater than four, it does not assert again until the number of unread messages within the Legacy Rx FIFO is decreased to be equal to or less than four. 0b - No occurrence of MB6 completing transmission/reception when MCR[RFEN]=0, or of Legacy Rx FIFO almost full when MCR[RFEN]=1 1b - MB6 completed transmission/reception when MCR[RFEN]=0, or Legacy Rx FIFO almost full when MCR[RFEN]=1
5 BUF5I	Buffer MB5 Interrupt Or Frames available in Legacy Rx FIFO When MCR[RFEN] is cleared (Legacy Rx FIFO disabled), this field flags the interrupt for MB5. **NOTE** This flag is cleared by the FlexCAN whenever MCR[RFEN] is changed by CPU writes. When MCR[RFEN] is set (Legacy Rx FIFO enabled), the BUF5I flag represents "Frames available in Legacy Rx FIFO" and indicates that at least one frame is available to be read from the Legacy Rx FIFO. When the MCR[DMA] bit is enabled, this flag generates a DMA request and the CPU must not clear this bit by writing 1 in BUF5I. 0b - No occurrence of MB5 completing transmission/reception when MCR[RFEN]=0, or of frame(s) available in the Legacy FIFO, when MCR[RFEN]=1 1b - MB5 completed transmission/reception when MCR[RFEN]=0, or frame(s) available in the Legacy Rx FIFO when MCR[RFEN]=1. It generates a DMA request in case of MCR[RFEN] and MCR[DMA] are enabled.
4-1 BUF4TO1I	Buffer MBi Interrupt Or Reserved When MCR[RFEN] is cleared (Legacy Rx FIFO disabled), these bits flag the interrupts for MB4 to MB1. **NOTE** These flags are cleared by the FlexCAN whenever MCR[RFEN] is changed by CPU writes. The BUF4TO1I flags are reserved when MCR[RFEN] is set. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception when MCR[RFEN]=0. 1b - The corresponding buffer has successfully completed transmission or reception when MCR[RFEN]=0.
0 BUF0I	Buffer MB0 Interrupt Or Clear Legacy FIFO bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
	When MCR[RFEN] is cleared (Legacy Rx FIFO disabled), this field flags the interrupt for MB0. If the Legacy Rx FIFO is enabled, this bit is used to trigger the clear Legacy FIFO operation. This operation empties Legacy FIFO contents. Before performing this operation the CPU must service all Legacy FIFO related IFLAGs. When MCR[DMA] is enabled this operation also clears the BUF5I flag and consequently aborts the DMA request. The clear Legacy FIFO operation occurs when the CPU writes 1 in BUF0I. It is only allowed in Freeze mode and is blocked by hardware in other conditions. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception when MCR[RFEN]=0. 1b - The corresponding buffer has successfully completed transmission or reception when MCR[RFEN]=0.

50.4.2.14 Control 2 register (CTRL2)

Offset

Register	Offset
CTRL2	34h

Function

This register complements Control1 register, providing control bits for memory write access in Freeze mode, for extending Legacy FIFO filter quantity, and to adjust the operation of internal FlexCAN processes like matching and arbitration.

The contents of this register are not affected by soft reset.

This table shows how the Legacy Rx FIFO filter structure is determined by the value of CTRL2[RFFN]. See the CTRL2[RFFN] field description for more information.

Table 339. Legacy Rx FIFO filter: possible structures

RFFN[3:0]	Number of Legacy Rx FIFO filter elements	Message buffers occupied by Legacy Rx FIFO and ID filter table	Remaining available mailboxes	Legacy Rx FIFO ID filter table elements affected by Rx individual masks	Legacy Rx FIFO ID filter table elements affected by Legacy Rx FIFO global mask
0h	8	MB 0-7	MB 8-127	Elements 0-7	none
1h	16	MB 0-9	MB 10-127	Elements 0-9	Elements 10-15
2h	24	MB 0-11	MB 12-127	Elements 0-11	Elements 12-23
3h	32	MB 0-13	MB 14-127	Elements 0-13	Elements 14-31
4h	40	MB 0-15	MB 16-127	Elements 0-15	Elements 16-39
5h	48	MB 0-17	MB 18-127	Elements 0-17	Elements 18-47
6h	56	MB 0-19	MB 20-127	Elements 0-19	Elements 20-55
7h	64	MB 0-21	MB 22-127	Elements 0-21	Elements 22-63

Table continues on the next page...

Table 339. Legacy Rx FIFO filter: possible structures (continued)

RFFN[3:0]	Number of Legacy Rx FIFO filter elements	Message buffers occupied by Legacy Rx FIFO and ID filter table	Remaining available mailboxes	Legacy Rx FIFO ID filter table elements affected by Rx individual masks	Legacy Rx FIFO ID filter table elements affected by Legacy Rx FIFO global mask
8h	72	MB 0-23	MB 24-127	Elements 0-23	Elements 24-71
9h	80	MB 0-25	MB 26-127	Elements 0-25	Elements 26-79
Ah	88	MB 0-27	MB 28-127	Elements 0-27	Elements 28-87
Bh	96	MB 0-29	MB 30-127	Elements 0-29	Elements 30-95
Ch	104	MB 0-31	MB 32-127	Elements 0-31	Elements 32-103
Dh	112	MB 0-33	MB 34-127	Elements 0-31	Elements 32-111
Eh	120	MB 0-35	MB 36-127	Elements 0-31	Elements 32-119
Fh	128	MB 0-37	MB 38-127	Elements 0-31	Elements 32-127

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ERRMSK_...	BOFFDON...	ECRWRE	WRMFRZ	RFFN				TASD					MRP	RRS	EACEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TIMER_S...	PRESCEN	BTE	ISOCANF...	EDFLTDIS	0	MBTSBASE		TSTAMPCAP		0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ERRMSK_FAST	Error Interrupt Mask for errors detected in the data phase of fast CAN FD frames This bit provides a mask for the ERRINT_FAST interrupt in ESR1 register. 0b - ERRINT_FAST error interrupt disabled. 1b - ERRINT_FAST error interrupt enabled.
30 BOFFDONEMSK	Bus Off Done Interrupt Mask This bit provides a mask for the bus off done interrupt in ESR1 register. 0b - Bus off done interrupt disabled. 1b - Bus off done interrupt enabled.
29	Error-correction Configuration Register Write Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
ECRWRE	This bit enables the MECR register to be updated. This bit is automatically set to 0 if the protocol described in Detection and correction of memory errors is not followed. 0b - Disable update. 1b - Enable update.
28 WRMFRZ	Write-Access To Memory In Freeze Mode Enable unrestricted write access to FlexCAN memory in Freeze mode. This bit can only be written in Freeze mode and has no effect out of Freeze mode. MCR[RFEN] must not be set during FlexCAN memory initialization. 0b - Maintain the write access restrictions. 1b - Enable unrestricted write access to FlexCAN memory.
27-24 RFFN	Number Of Legacy Rx FIFO Filters This 4-bit field defines the number of Rx Legacy FIFO filters, as shown in Table 339. The maximum selectable number of filters is determined by the chip. This field can only be written in Freeze mode as it is blocked by hardware in other modes. This field must not be programmed with values that cause the number of message buffers occupied by Legacy Rx FIFO and Legacy Rx FIFO ID Filter to exceed the number of mailboxes present, as defined by MCR[MAXMB]. **NOTE** Each group of eight filters occupies a memory space equivalent to two message buffers, which means that the more filters are implemented the fewer mailboxes will be available. Considering that the Legacy Rx FIFO occupies the memory space originally reserved for MB0-5, RFFN should be programmed with a value corresponding to a number of filters not greater than the number of available memory words, which can be calculated as follows: (SETUP_MB - 6) × 4 where SETUP_MB is the smaller of the parameter NUMBER_OF_MB and MCR[MAXMB]. The number of remaining mailboxes available will be: (SETUP_MB - 8) - (RFFN × 2) If the number of Legacy Rx FIFO filters programmed through RFFN exceeds the SETUP_MB value (memory space available), then the exceeding ones will not be functional. **NOTE** • The number of the last remaining available mailboxes is defined by the smaller of NUMBER_OF_MB minus 1 and MCR[MAXMB]. • If Rx Individual Mask registers are not enabled then all Legacy Rx FIFO filters are affected by the Legacy Rx FIFO Global Mask.
23-19 TASD	Tx Arbitration Start Delay

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This 5-bit field indicates how many CAN bits the Tx arbitration process start point can be delayed from the first bit of CRC field on CAN bus. See Tx arbitration start delay for more details. This field can be written only in Freeze mode because it is blocked by hardware in other modes.
18 MRP	Mailboxes Reception Priority If this bit is set the matching process starts from the mailboxes and if no match occurs the matching continues on the Legacy Rx FIFO or Enhanced RX FIFO . This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0b - Matching starts from Legacy Rx FIFO or Enhanced Rx FIFO and continues on mailboxes. 1b - Matching starts from mailboxes and continues on Legacy Rx FIFO or Enhanced Rx FIFO.
17 RRS	Remote Request Storing If this bit is asserted a remote request frame is submitted to a matching process and stored in the corresponding message buffer in the same fashion as a data frame. No automatic remote response frame will be generated. If this bit is negated the remote request frame is submitted to a matching process and an automatic remote response frame is generated if a message buffer with CODE = 1010b is found with the same ID. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0b - Remote response frame is generated. 1b - Remote request frame is stored.
16 EACEN	Entire Frame Arbitration Field Comparison Enable For Rx Mailboxes This bit controls the comparison of IDE and RTR bits within Rx mailbox filters with their corresponding bits in the incoming frame by the matching process. This bit does not affect matching for Legacy Rx FIFO or Enhanced Rx FIFO. This bit can be written only in Freeze mode because it is blocked by hardware in other modes. 0b - Rx mailbox filter's IDE bit is always compared and RTR is never compared despite mask bits. 1b - Enables the comparison of both Rx mailbox filter's IDE and RTR bit with their corresponding bits within the incoming frame. Mask bits do apply.
15 TIMER_SRC	Timer Source Selects the time tick source used for incrementing the free running timer counter. This bit can be written in Freeze mode only. 0b - The free running timer is clocked by the CAN bit clock, which defines the baud rate on the CAN bus. 1b - The free running timer is clocked by an external time tick. The period can be either adjusted to be equal to the baud rate on the CAN bus, or a different value as required. See the device-specific section for details about the external time tick.
14 PREXCEN	Protocol Exception Enable This bit enables the protocol exception feature.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is writable only in Freeze mode. **NOTE** See Protocol exception event in the CAN Protocol standard (ISO 11898-1) for details. 0b - Protocol exception is disabled. 1b - Protocol exception is enabled.
13 BTE	Bit Timing Expansion enable This field enables the use of the EPRS, EDCBT and ENCBT registers to configure the CAN bit timing segments, instead of using the bit timing fields of CBT, FDCBT, and CTRL1 registers. If CTRL2[BTE] is set: • PRESDIV, PROPSEG, PSEG1, PSEG2, and RJW fields in CTRL1 register are read as zero, and a write operation in them has no effect. • EPRESDIV, ERJW, EPROPSEG, EPSEG1, and EPSEG2 fields in the CBT register, and the FDCBT register, are read as zero and a write operation to them has no effect. • The ETDCOFF, ETDCEN, ETDCFAIL, and ETDCVAL fields of the ETDC register are used by FlexCAN instead of TDCOFF, TDCEN, TDCFAIL, and TDCVAL fields in FDCTRL register, which are read as zero, and a write operation in them has no effect. • ETDC[TDMDIS] can be used for disabling transceiver delay measurement. 0b - CAN Bit timing expansion is disabled. 1b - CAN bit timing expansion is enabled.
12 ISOCANFDEN	ISO CAN FD Enable This field enables the CAN FD protocol according to ISO specification (ISO 11898-1) (see CAN FD ISO compliance). This field is writable only in Freeze mode. **NOTE** FlexCAN is able to transmit FD frame format according to CAN Protocol standard (ISO 11898-1). 0b - FlexCAN operates using the non-ISO CAN FD protocol. 1b - FlexCAN operates using the ISO CAN FD protocol (ISO 11898-1).
11 EDFLTDIS	Edge Filter Disable This bit disables the edge filter used during the bus integration state. When the Edge Filter is enabled, two consecutive nominal time quanta with dominant bus state are required to detect an edge that causes synchronization. When synchronization occurs, the counting of the sequence of eleven consecutive recessive bits is restarted. The edge filter prevents the dominant pulses that are shorter than a nominal bit time (present during the data phase of an FD frame) from being mistaken for an idle condition.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is writable only in Freeze mode. **NOTE** See Bus Integration state in the CAN Protocol standard (ISO 11898-1) for details. 0b - Edge filter is enabled 1b - Edge filter is disabled
10 —	Reserved
9-8 MBTSBASE	Message Buffer Time Stamp Base This field selects which time base is used for capturing the 16-bit TIME_STAMP field of the message buffer register. There are three time base options: 1. TIMER 2. Lower 16 bits of high resolution time base 3. Upper 16 bits of high resolution time base This field can be written in Freeze Mode only. 00b - Message buffer time stamp base is TIMER 01b - Message buffer time stamp base is lower 16 bits of high resolution timer 10b - Message buffer time stamp base is upper 16 bits of high resolution timer 11b - Reserved
7-6 TSTAMPCAP	Time Stamp Capture Point This field configures the point in time when a 32-bit time base is captured during a CAN frame and stored in the high resolution time stamp register (HR_TIME_STAMP). For classical CAN frames, capture points can be the start of frame bit or the point in time a CAN frame is considered valid, which is the 7th bit of end of frame for transmission and the 6th bit of the end of frame for reception. For CAN FD frames, the high resolution time stamp can be captured at the start of frame, the point in time a CAN FD frame is considered valid, or the res bit. This field can be written in Freeze mode only. 00b - The high resolution time stamp capture is disabled 01b - The high resolution time stamp is captured in the end of the CAN frame 10b - The high resolution time stamp is captured in the start of the CAN frame 11b - The high resolution time stamp is captured in the start of frame for classical CAN frames and in res bit for CAN FD frames
5-0 —	Reserved

50.4.2.15 Error and Status 2 register (ESR2)

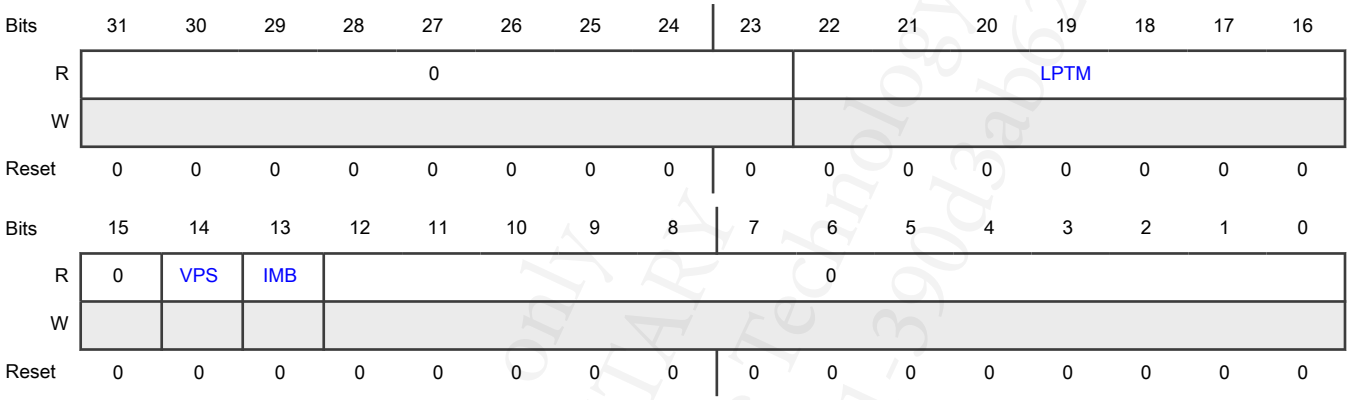
Offset

Register	Offset
ESR2	38h

Function

This register reports some general status information.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 LPTM	Lowest Priority Tx Mailbox If ESR2[VPS] is asserted, this field indicates the lowest number inactive mailbox (see the ESR2[IMB] bit description). If there is no inactive mailbox then the mailbox indicated depends on the value of CTRL1[LBUF]. If CTRL1[LBUF] is negated, then the mailbox indicated is the one that has the greatest arbitration value (see Highest-priority mailbox first). If CTRL1[LBUF] is asserted then the mailbox indicated is the highest number active Tx mailbox. If a Tx mailbox is being transmitted it is not considered in LPTM calculation. If ESR2[IMB] is not asserted and a frame is transmitted successfully, LPTM is updated with its mailbox number.
15 —	Reserved
14 VPS	Valid Priority Status This bit indicates whether ESR2[IMB] and ESR2[LPTM] contents are currently valid or not. It is asserted upon every complete Tx arbitration process unless the CPU writes to Control and Status word of a mailbox that has already been scanned, that is, it is behind Tx Arbitration Pointer, during the Tx arbitration process.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	If there is no inactive mailbox and only one Tx mailbox that is being transmitted then VPS is not asserted. This bit is negated upon the start of every Tx arbitration process or upon a write to Control and Status word of any mailbox. **NOTE** ESR2[VPS] is not affected by any CPU write into Control Status (C/S) of an MB that is blocked by abort mechanism. When MCR[AEN] is asserted, the abort code write in C/S of an MB that is being transmitted (pending abort), or any write attempt into a Tx MB with IFLAG set is blocked. 0b - Contents of IMB and LPTM are invalid. 1b - Contents of IMB and LPTM are valid.
13 IMB	Inactive Mailbox If ESR2[VPS] is asserted, this bit indicates whether there is any inactive mailbox (CODE field is either 1000b or 0b). This bit is asserted in the following cases: • During arbitration, if an ESR2[LPTM] is found and it is inactive. • If ESR2[IMB] is not asserted and a frame is transmitted successfully. This bit is always cleared in start of arbitration (see Arbitration process). **NOTE** ESR2[LPTM] mechanism has the following behavior: if an MB is successfully transmitted and ESR2[IMB]=0 (no inactive mailbox), then ESR2[VPS] and ESR2[IMB] are asserted and the index related to the MB just transmitted is loaded into ESR2[LPTM]. 0b - If ESR2[VPS] is asserted, the ESR2[LPTM] is not an inactive mailbox. 1b - If ESR2[VPS] is asserted, there is at least one inactive mailbox. LPTM content is the number of the first one.
12-0 —	Reserved

50.4.2.16 CRC register (CRCR)

Offset

Register	Offset
CRCR	44h

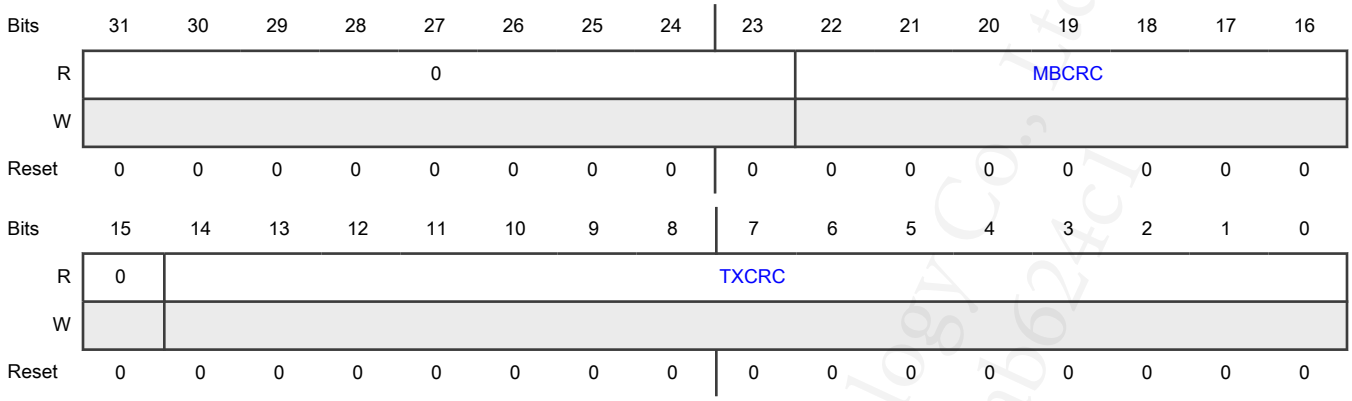
Function

This register provides information about the CRC of transmitted messages for non-FD messages. This register only reports the 15 low order bits of CRC calculations for messages in CAN FD format that require either 17 or 21 bits. For CAN FD format frames, the FDCRC register must be used. This register is updated at the same time the Tx interrupt flag is asserted.

NOTE

See CRC sequence calculation in the CAN Protocol standard (ISO 11898-1) for details.

Diagram



Fields

Field	Function
31-23 —	Reserved
22-16 MBCRC	CRC Mailbox This field indicates the number of the mailbox corresponding to the value in CRCR[TXCRC] field.
15 —	Reserved
14-0 TXCRC	Transmitted CRC value This field indicates the CRC value of the last transmitted message for non-FD frames. For FD frames, CRC value is reported in FDCRC register.

50.4.2.17 Legacy Rx FIFO Global Mask register (RXFGMASK)

Offset

Register	Offset
RXFGMASK	48h

Function

This register is located in RAM.

If Legacy Rx FIFO is enabled, RXFGMASK is used to mask the Legacy Rx FIFO ID filter table elements that do not have a corresponding RXIMR according to CTRL2[RFFN] field setting.

This register can only be written in Freeze mode as it is blocked by hardware in other modes.

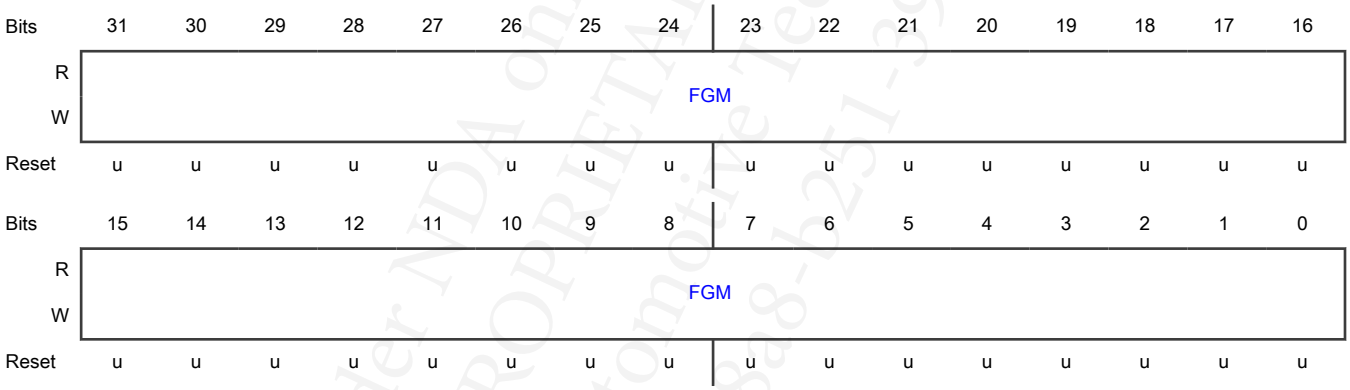
The following table shows how the FGM bits correspond to each IDAF field.

Table 340. Correspondence of Legacy Rx FIFO global mask bits to IDF fields

Legacy Rx FIFO ID filter table elements format (MCR[IDAM])	Identifier acceptance filter fields					
	RTR	IDE	RXIDA	RXIDB [1]	RXIDC [2]	Reserved
A	FGM[31]	FGM[30]	FGM[29:1]	—	—	FGM[0]
B	FGM[31], FGM[15]	FGM[30], FGM[14]	—	FGM[29:16], FGM[13:0]		—
C	—	—		—	FGM[31:24], FGM[23:16], FGM[15:8], FGM[7:0]	

1. If MCR[IDAM] is equivalent to format B, only the fourteen most significant bits of the identifier of the incoming frame are compared with the Legacy Rx FIFO filter.
2. If MCR[IDAM] is equivalent to format C, only the eight most significant bits of the identifier of the incoming frame are compared with the Legacy Rx FIFO filter.

Diagram



Fields

Field	Function
31-0 FGM	Legacy Rx FIFO Global Mask Bits These bits mask the ID filter table elements bits in a perfect alignment. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIT Automotive Technology Co., Ltd.
c8c46205-4821-4848-b251-590d8a0ec1

50.4.2.18 Legacy Rx FIFO Information register (RXFIR)

Offset

Register	Offset
RXFIR	4Ch

Function

RXFIR provides information on Legacy Rx FIFO.

This register is the port through which the CPU accesses the output of the Legacy RXFIR FIFO located in RAM. The Legacy RXFIR FIFO is written by the FlexCAN whenever a new message is moved into the Legacy Rx FIFO. Also, its output is updated whenever the output of the Legacy Rx FIFO is updated with the next message. See Legacy Rx FIFO for instructions on reading this register.

NOTE

RXFIR can be written only during memory initialization, due to the error code correction (ECC) feature. In every other case the register is read-only.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							IDHIT								
W																
Reset	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u	u

Fields

Field	Function
31-9 —	Reserved
8-0 IDHIT	Identifier Acceptance Filter Hit Indicator This field indicates which Identifier Acceptance filter was hit by the received message that is in the output of the Legacy Rx FIFO. If multiple filters match the incoming message ID then the first matching IDAF found (lowest number) by the matching process is indicated. This field is valid only when IFLAG1[BUF5I] is asserted.

Preliminary NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e5c-4821-48a8-b251-390d3b6b4c1

50.4.2.19 CAN Bit Timing register (CBT)

Offset

Register	Offset
CBT	50h

Function

This register is an alternative way to store the CAN bit timing variables described in CTRL1 register. EPRESDIV, EPROPSEG, EPSEG1, EPSEG2, and ERJW are extended versions of PRESDIV, PROPSEG, PSEG1, PSEG2, and RJW fields respectively.

The BTF bit selects the use of the timing variables defined in this register.

The contents of this register are not affected by soft reset.

NOTE

The CAN bit variables in CTRL1 and in CBT are stored in the same register.

NOTE

When the CAN FD feature is enabled (MCR[FDEN] is set), always set CBT[BTF].

NOTE

The user must ensure bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1).

NOTE

If CTRL2[BTE] is set, EPRESDIV, ERJW, EPROPSEG, EPSEG1, and EPSEG2 fields are read as zero, and a write operation into them has no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BTF	EPRESDIV										ERJW				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EPROPSEG						EPSEG1					EPSEG2				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BTF	Bit Timing Format Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Enables the use of extended CAN bit timing fields EPRESDIV, EPROPSEG, EPSEG1, EPSEG2. and ERJW replacing the CAN bit timing variables defined in CTRL1 register. This field can be written in Freeze mode only. 0b - Extended bit time definitions disabled. 1b - Extended bit time definitions enabled.
30-21 EPRESDIV	Extended Prescaler Division Factor This 10-bit field defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency when CBT[BTF] is asserted, otherwise it has no effect. It extends the CTRL1[PRESDIV] value range. The Sclock period defines the time quantum of the CAN protocol. For the reset value, the Sclock frequency is equal to the PE clock frequency (see Protocol timing). This field can be written only in Freeze mode because it is blocked by hardware in other modes. Sclock frequency = PE clock frequency / (EPRESDIV + 1)
20-16 ERJW	Extended Resync Jump Width This 5-bit field defines the maximum number of time quanta that a bit time can be changed by one re-synchronization when CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CTRL1[RJW] value range. One time quantum is equal to the Sclock period. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Resync Jump Width = ERJW + 1.
15-10 EPROPSEG	Extended Propagation Segment This 6-bit field defines the length of the propagation segment in the bit time when CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CTRL1[PROPSEG] value range. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Propagation Segment Time = (EPROPSEG + 1) × Time-Quanta. Time-Quantum = one Sclock period.
9-5 EPSEG1	Extended Phase Segment 1 This 5-bit field defines the length of phase segment 1 in the bit time when CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CTRL1[PSEG1] value range. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 1 = (EPSEG1 + 1) × Time-Quanta. Time-Quantum = one Sclock period.
4-0 EPSEG2	Extended Phase Segment 2 This 5-bit field defines the length of phase segment 2 in the bit time when CBT[BTF] bit is asserted, otherwise it has no effect. It extends the CTRL1[PSEG2] value range. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Buffer Segment 1 = (EPSEG2 + 1) × Time-Quanta. Time-Quantum = one Sclock period.

50.4.2.20 Interrupt Masks 4 register (IMASK4)

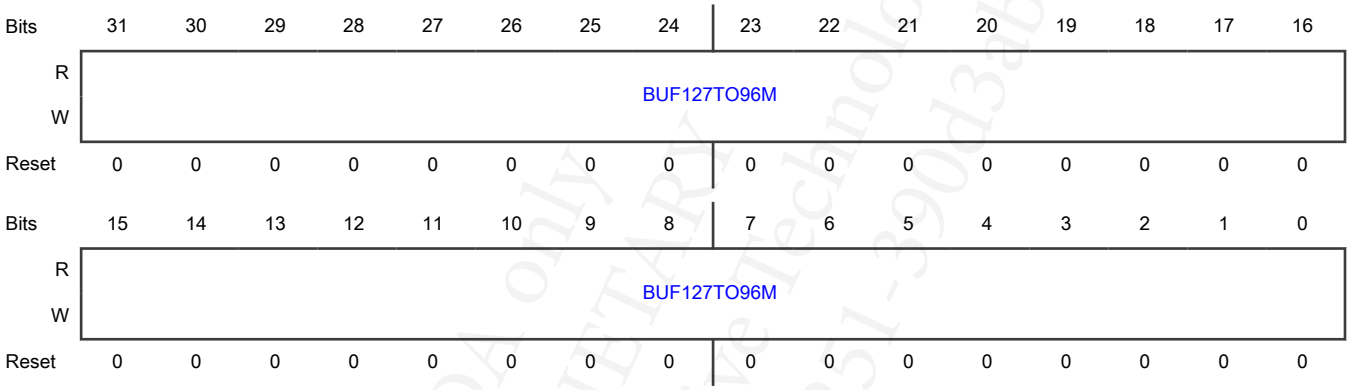
Offset

Register	Offset
IMASK4	68h

Function

This register allows any number of a range of the 32 message buffer interrupts to be enabled or disabled for MB127 to MB96. It contains one interrupt mask bit per buffer, enabling the CPU to determine which buffer generates an interrupt after a successful transmission or reception, that is, when the corresponding IFLAG4 bit is set.

Diagram



Fields

Field	Function
31-0 BUF127TO96M	Buffer MBi Mask Each bit enables or disables the corresponding FlexCAN message buffer interrupt for MB127 to MB96. When CAN FD is enabled, the MB range is defined in accordance with the MBDSRs bit fields of FDCTRL register. **NOTE** Setting or clearing a bit in the IMASK4 Register can assert or negate an interrupt request, if the corresponding IFLAG4 bit is set. 0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

50.4.2.21 Interrupt Masks 3 register (IMASK3)

Offset

Register	Offset
IMASK3	6Ch

Function

This register allows any number of a range of the 32 message buffer interrupts to be enabled or disabled for MB95 to MB64. It contains one interrupt mask bit per buffer, enabling the CPU to determine which buffer generates an interrupt after a successful transmission or reception, that is, when the corresponding IFLAG3 bit is set.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	BUF95TO64M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	BUF95TO64M															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BUF95TO64M	Buffer MBi Mask Each bit enables or disables the corresponding FlexCAN message buffer interrupt for MB95 to MB64. When CAN FD is enabled, the MB range is defined in accordance with the MBDSRs bit fields of FDCTRL register. **NOTE** Setting or clearing a bit in the IMASK3 Register can assert or negate an interrupt request, if the corresponding IFLAG3 bit is set. 0b - The corresponding buffer interrupt is disabled. 1b - The corresponding buffer interrupt is enabled.

50.4.2.22 Interrupt Flags 4 register (IFLAG4)

Offset

Register	Offset
IFLAG4	70h

Function

This register defines the flags for the 32 message buffer interrupts for MB127 to MB96. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding IFLAG4 bit. If the corresponding IMASK4 bit is set, an interrupt will be generated. The interrupt flag must be cleared by writing 1 to it. Writing 0 has no effect.

Before updating MCR[MAXMB] field, CPU must service the IFLAG4 bits whose MB value is greater than the MAXMB to be updated; otherwise, they will remain set and be inconsistent with the number of MBs available.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BUF127TO96															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BUF127TO96															
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 BUF127TO96	Buffer MBi Interrupt Each bit flags the corresponding FlexCAN message buffer interrupt for MB127 to MB96.When CAN FD is enabled, the MB range is defined in accordance to the MBDSRs bit fields of FDCTRL register. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.

50.4.2.23 Interrupt Flags 3 register (IFLAG3)

Offset

Register	Offset
IFLAG3	74h

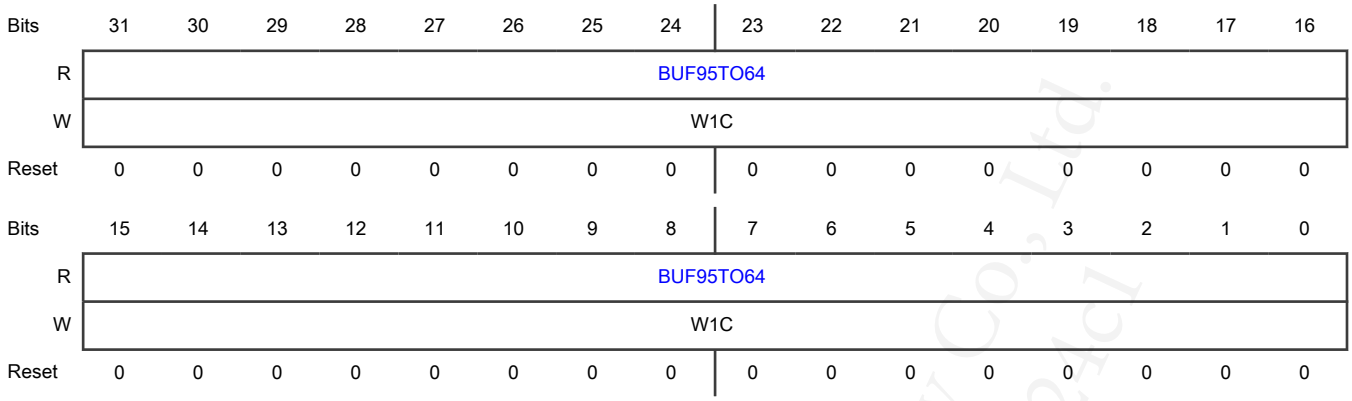
Function

This register defines the flags for the 32 message buffer interrupts for MB95 to MB64. It contains one interrupt flag bit per buffer. Each successful transmission or reception sets the corresponding IFLAG3 bit. If the corresponding IMASK3 bit is set, an interrupt will be generated. The interrupt flag must be cleared by writing 1 to it. Writing 0 has no effect.

Before updating MCR[MAXMB] field, CPU must service the IFLAG3 bits whose MB value is greater than the MAXMB to be updated; otherwise, they will remain set and be inconsistent with the number of MBs available.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CAN Automotive Technology Co., Ltd.
c8c462e3-3201-48a8-b25f-59d8ab624c1

Diagram



Fields

Field	Function
31-0 BUF95TO64	Buffer MBi Interrupt Each bit flags the corresponding FlexCAN message buffer interrupt for MB95 to MB64.When CAN FD is enabled, the MB range is defined in accordance to the MBDSRs bit fields of FDCTRL register. 0b - The corresponding buffer has no occurrence of successfully completed transmission or reception. 1b - The corresponding buffer has successfully completed transmission or reception.

50.4.2.24 Rx Individual Mask registers (RXIMR0 - RXIMR127)

Offset

For n = 0 to 127:

Register	Offset
RXIMRn	880h + (n × 4h)

Function

The RX Individual Mask Registers are used to store the acceptance masks for ID filtering in Rx MBs and the Legacy Rx FIFO.

When the Legacy Rx FIFO is disabled (MCR[RFEN] bit is negated), an individual mask is provided for each available Rx mailbox on a one-to-one correspondence. When the Legacy Rx FIFO is enabled (MCR[RFEN] bit is asserted), an individual mask is provided for each Legacy Rx FIFO ID filter table element on a one-to-one correspondence depending on the setting of CTRL2[RFFN] (see Legacy Rx FIFO).

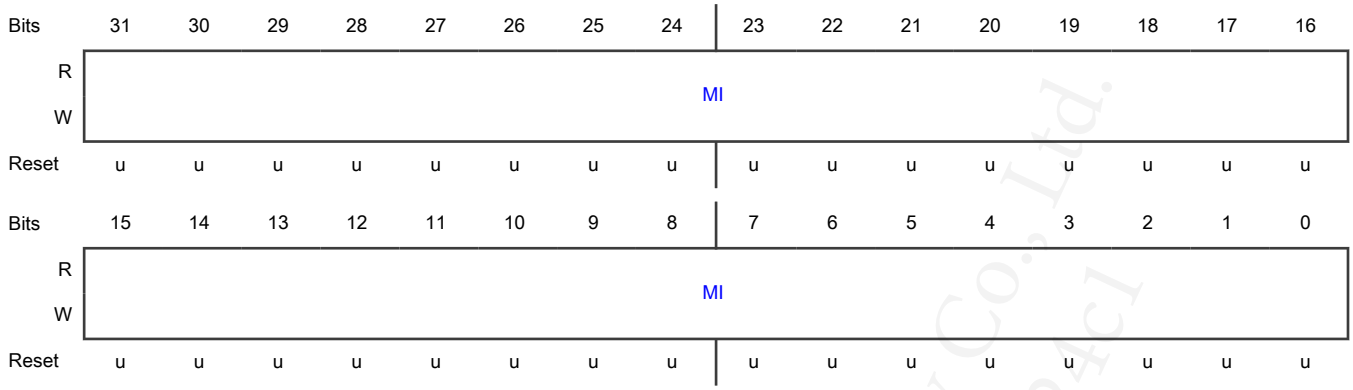
RXIMR0 stores the individual mask associated with either MB0 or ID filter table element 0, RXIMR1 stores the individual mask associated with either MB1 or ID filter table element 1, and so on.

RXIMR registers can only be accessed by the CPU when the module is in Freeze mode; otherwise, they are blocked by hardware. These registers are not affected by reset. They are located in RAM and must be explicitly initialized prior to any reception.

It is possible for the RXIMR memory region to be accessed as general purpose memory. See Bus interface for more information.

Provided under NDA of COMPANY PROPRIETARY Beijing HJJ Automotive Technology Co., Ltd. c8c0e2e1-6821-48a8-b25f-59d8ab624c1

Diagram



Fields

Field	Function
31-0 MI	Individual Mask Bits Each individual mask bit masks the corresponding bit in both the mailbox filter and Legacy Rx FIFO ID filter table element in distinct ways. For mailbox filters, see the RXMGMASK register description. For Legacy Rx FIFO ID filter table elements, see the RXFGMASK register description. 0b - The corresponding bit in the filter is "don't care." 1b - The corresponding bit in the filter is checked.

50.4.2.25 Memory Error Control register (MECR)

Offset

Register	Offset
MECR	AE0h

Function

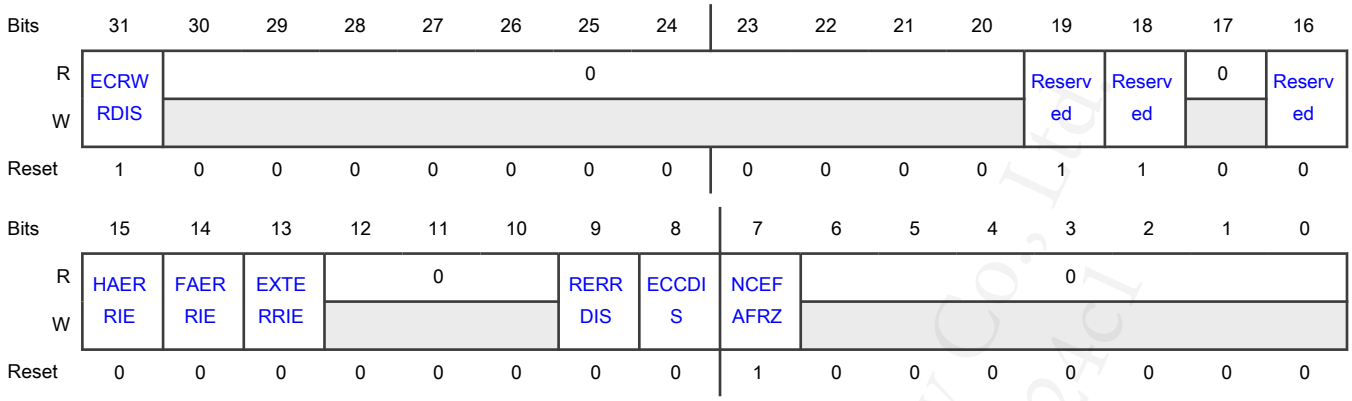
This register contains control bits for memory error detection and correction (ECC).

NOTE

When bit CTRL2[ECRWRE] is 0, writes in this register are blocked, except in the ECRWRDIS bit.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 ECRWRDIS	Error Configuration Register Write Disable Disables writes on this register. This bit is automatically set to 1 (disabled) when CTRL2[ECRWRE] is enabled. The protocol described in section Detection and correction of memory errors must be followed. 0b - Write is enabled. 1b - Write is disabled.
30-20 —	Reserved
19 —	Reserved When writing to this field, always write the reset value.
18 —	Reserved When writing to this field, always write the reset value.
17 —	Reserved
16 —	Reserved When writing to this field, always write the reset value.
15 HAERRIE	Host Access Error Injection Enable Enables the injection of errors only in memory reads issued by the host (CPU). 0b - Injection is disabled. 1b - Injection is enabled.
14	FlexCAN Access Error Injection Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
FAERRIE	Enables the injection of errors only in memory reads issued by the FlexCAN internal processes. 0b - Injection is disabled. 1b - Injection is enabled.
13 EXTERRIE	Extended Error Injection Enable Memory accesses performed by internal FlexCAN processes are 64-bit. This bit extends the error injection on 32-bit memory accesses to the complementary 32-bit word using the same 32-bit error injection data and parity words. See Error Injection Data Pattern register (ERRIDPR) and Error Injection Parity Pattern register (ERRIPPR). 0b - Error injection is applied only to the 32-bit word. 1b - Error injection is applied to the 64-bit word.
12-10 —	Reserved
9 RERRDIS	Error Report Disable Disables the update of the error report registers. The update of error-related flags and the generation of bus transfer errors are still active. **NOTE** When reading the report registers, this bit must be set to assure coherence on the consecutive register reads. 0b - Enable updates of the error report registers. 1b - Disable updates of the error report registers.
8 ECCDIS	Error Correction Disable Disables completely the memory detection and correction mechanism. Besides disabling the error report mechanism, it also stops the update of the error-related flags and generation of bus transfer errors. The parity bits continue being calculated and written into memory on write transactions. 0b - Enable memory error correction. 1b - Disable memory error correction.
7 NCEFAFRZ	Non-Correctable Errors In FlexCAN Access Put Device In Freeze Mode Determines the response when a non-correctable error is detected in a memory read performed by FlexCAN internal processes. In this case, entering Freeze mode prevents corrupted data from being treated as valid by FlexCAN internal processes. 0b - Keep normal operation. 1b - Put FlexCAN in Freeze mode (see section "Freeze mode").
6-0 —	Reserved

50.4.2.26 Error Injection Address register (ERRIAR)

Offset

Register	Offset
ERRIAR	AE4h

Function

This register holds the address where error is to be injected.

Use the following table to convert from the memory map address to the location in the physical FlexCAN RAM (where pairs of values are provided, the first is the address for MCR[FDEN] negated, the second is for MCR[FDEN] asserted):

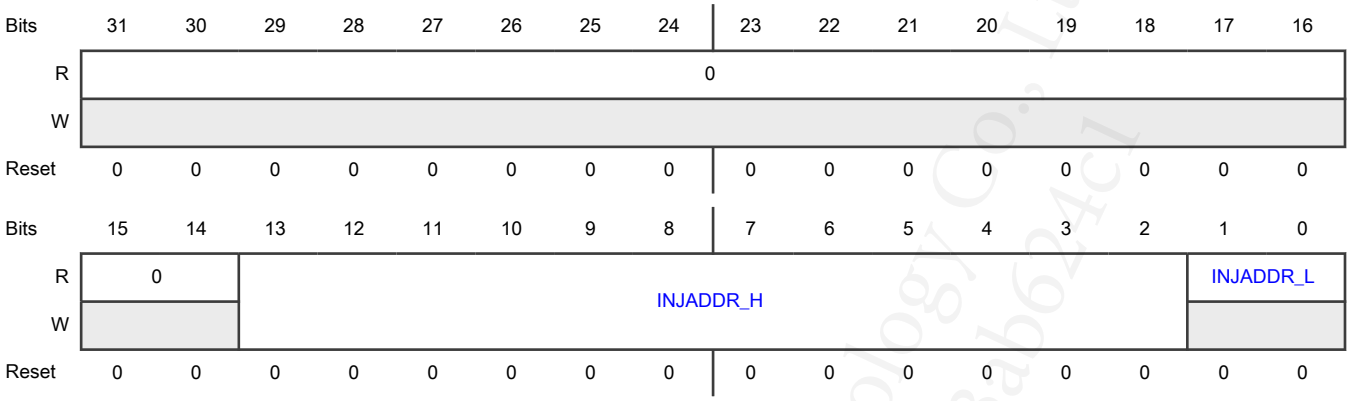
RAM contents	Injection address	Memory map
FlexCAN Registers	Not mapped	—
MBs	0000h	0080h
RXIMRs	1000h	0880h
RXFIR_0	1200h	0A80h
RXFIR_1	1204h	0A84h
RXFIR_2	1208h	0A88h
RXFIR_3	120Ch	0A8Ch
RXFIR_4	1210h	0A90h
RXFIR_5	1214h	0A94h
Reserved	—	0A98h
RXMGMASK	1220h	0AA0h
RXFGMASK	1224h	0AA4h
RX14MASK	1228h	0AA8h
RX15MASK	122Ch	0AACh
Tx_SMB	1230h	0AB0h / 0F28h
Rx_SMB0	1240h / 1278h	0AC0h / 0F70h
Rx_SMB1	1250h / 12C0h	0AD0h / 0FB8h
Rx_SMB0_TIME_STAMP	1308h	0C20h
Rx_SMB1_TIME_STAMP	130Ch	0C24h
HR_TIME_STAMP	1310h	0C30h
Enhanced MB memory	800h	1000h
Enhanced RX FIFO	1314h	2000h
ERFFEL	1D14h	3000h

Provided under NXP ONLY
COMPANY PROPRIETARY
Beijing Automotive Technology Co., Ltd.
ca162e5-4821-48a8-b251-a9b5a6d2c1

NOTE

For RXFIR, Enhanced RX FIFO, and HR_TIME_STAMP addresses, you must inject ECC errors in host access only.

Diagram



Fields

Field	Function
31-14 —	Reserved
13-2 INJADDR_H	Error Injection Address High This read-write field defines the twelve most significant bits of the physical RAM address where error is to be injected (see table above).
1-0 INJADDR_L	Error Injection Address Low This read-only field defines the two least significant bits of the physical RAM address where error is to be injected. They ensure that the address is on a thirty-two-bit boundary.

50.4.2.27 Error Injection Data Pattern register (ERRIDPR)

Offset

Register	Offset
ERRIDPR	AE8h

Function

Holds the error pattern to be injected in the data word read from memory.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c402e5-4821-4c8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DFLIP															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DFLIP															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DFLIP	Data flip pattern Bits set to 1 in the flip pattern cause the corresponding data bit in the word read from memory to invert.

50.4.2.28 Error Injection Parity Pattern register (ERRIPPR)

Offset

Register	Offset
ERRIPPR	AECh

Function

Holds the error pattern to be injected in parity bits read from memory along with data word.

Bits set to 1 in the flip pattern cause the corresponding parity bit in the word read from memory to invert.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			PFLIP3					0			PFLIP2				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			PFLIP1					0			PFLIP0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co.,Ltd.
c8462e54-4821-42a8-b251-390d3ab624c1

Fields

Field	Function
31-29 —	Reserved
28-24 PFLIP3	Parity Flip Pattern For Byte 3 (most significant)
23-21 —	Reserved
20-16 PFLIP2	Parity Flip Pattern For Byte 2
15-13 —	Reserved
12-8 PFLIP1	Parity Flip Pattern For Byte 1
7-5 —	Reserved
4-0 PFLIP0	Parity Flip Pattern For Byte 0 (Least Significant)

50.4.2.29 Error Report Address register (RERRAR)

Offset

Register	Offset
RERRAR	AF0h

Function

Reports the address used for an access in which an error (correctable or non-correctable) was detected, and also reports the identification of the source of that access.

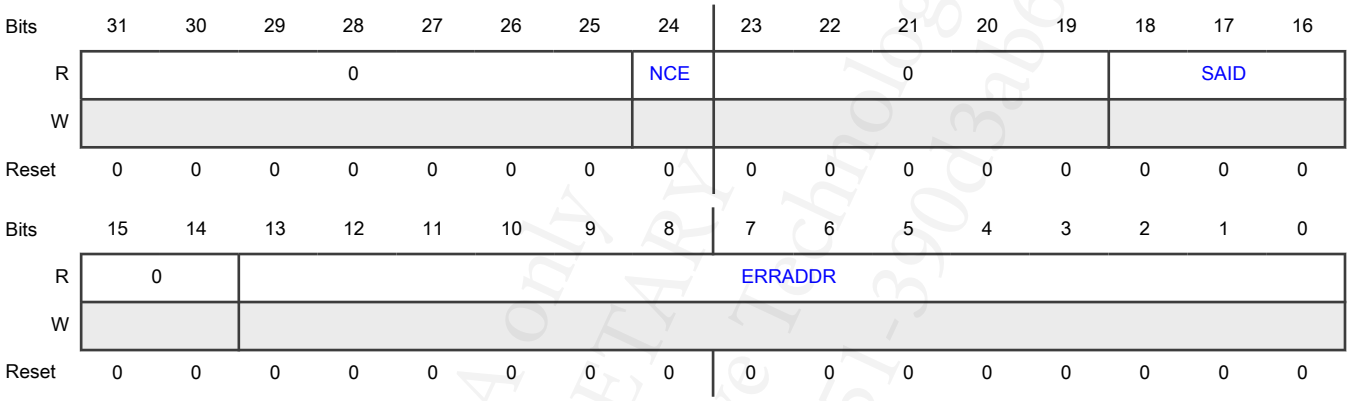
This address is always reported using a 32-bit alignment. Non-aligned accesses (ERRADDR[1:0] non-0) are reported with the address aligned and data is reported in RERRDR accordingly shifted. In case of errors detected in accesses larger than 32-bit (as performed by FlexCAN internal processes), the address of the 32-bit word which the error was detected is reported. In case of errors detected in more than one 32-bit word, only the least significant address is reported.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5e5-4821-48a8-b251-390d3ab624c1

Table 341. Source of memory access

SAID[2:0]	Error during...
0	Move-out FlexCAN access
1	Move-in
2	Tx Arbitration
3	Rx Matching
4	Move-out Host access
5-7	Reserved

Diagram



Fields

Field	Function
31-25 —	Reserved
24 NCE	Non-Correctable Error Indicates that the report is due to a non-correctable error. 0b - Reporting a correctable error 1b - Reporting a non-correctable error
23-19 —	Reserved
18-16 SAID	SAID SAID[2] — Identification of the requester of the memory read request: • 0 = Requested by FlexCAN internal processes • 1 = Requested by host (CPU) SAID[1] — Details of FlexCAN operation:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• 0 = Move • 1 = Scanning SAID[0] — Operation that requested the memory read: • 0 = Transmission • 1 = Reception For more information see Table 341.
15-14 —	Reserved
13-0 ERRADDR	Address Where Error Detected See the description of the Error Injection Address Register (ERRIAR).

50.4.2.30 Error Report Data register (RERRDR)

Offset

Register	Offset
RERRDR	AF4h

Function

Reports the raw data (unmodified by the correction performed by ECC logic) read from memory with error. The value reported does not represent the transient values of the BUSY bit (see Table 349) when reading a message buffer.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RDATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482f-48a-b251-390d3ab624c1

Fields

Field	Function
31-0 RDATA	Raw data word read from memory with error

50.4.2.31 Error Report Syndrome register (RERRSYNR)

Offset

Register	Offset
RERRSYNR	AF8h

Function

Holds the syndrome detected in a memory read with error. It also reports the bytes which were read in this 32-bit read transaction.

Each SYNDn field indicates the type of error and which bit in byte (n) is affected by the error. (SYND3 corresponds to the most significant byte in the data word read from memory; SYND0 corresponds to the least significant.)

Table 342. Syndrome definition

SYNDn (hex)	Type	Bit affected
00	—	none (no error)
01	Code	0
02	Code	1
04	Code	2
07	Data	5
08	Code	3
0E	Data	7
10	Code	4
13	Data	2
15	Data	6
16	Data	1
19	Data	3
1A	Data	4
1C	Data	0
06	—	All-zeros non-correctable error
1F	—	All-ones non-correctable error
All others	—	Non-correctable error

Provided under NDA on behalf of COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-4821-48a8-b253-39c03ab624c1

Each BEn field indicates which byte in the 32-bit word reported was effectively read. The syndrome bits are calculated for all bytes, even for the non-read ones. Errors detected in non-read bytes are indicated (see Error indication) and reported (see Error reporting).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BE3	0		SYND3					BE2	0		SYND2				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BE1	0		SYND1					BE0	0		SYND0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 BE3	Byte Enabled For Byte 3 (most significant) 0b - The byte was not read. 1b - The byte was read.
30-29 —	Reserved
28-24 SYND3	Error Syndrome For Byte 3 (most significant) See Table 342.
23 BE2	Byte Enabled For Byte 2 0b - The byte was not read. 1b - The byte was read.
22-21 —	Reserved
20-16 SYND2	Error Syndrome For Byte 2 See Table 342.
15 BE1	Byte Enabled For Byte 1 0b - The byte was not read. 1b - The byte was read.
14-13	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
12-8 SYND1	Error Syndrome for Byte 1 See Table 342.
7 BE0	Byte Enabled For Byte 0 (least significant) 0b - The byte was not read. 1b - The byte was read.
6-5 —	Reserved
4-0 SYND0	Error Syndrome For Byte 0 (least significant) See Table 342.

50.4.2.32 Error Status register (ERRSR)

Offset

Register	Offset
ERRSR	AFCh

Function

Holds the status bits of the error correction and detection operations. These flags can be cleared by writing 1 to them. Writing 0 has no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												HANCEIF	FANCEIF	0	CEIF
W													W1C	W1C		W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												HANCEIOF	FANCEIOF	0	CEIOF
W													W1C	W1C		W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-4848-b251-390d3ab624c1

Fields

Field	Function
31-20 —	Reserved
19 HANCEIF	Host Access With Non-Correctable Error Interrupt Flag Indicates that a non-correctable error was detected in a memory read initiated by host. A bus transfer error is asserted for that access. No interrupt is associated to this flag. 0b - No non-correctable errors were detected in host accesses so far. 1b - A non-correctable error was detected in a host access.
18 FANCEIF	FlexCAN Access With Non-Correctable Error Interrupt Flag Indicates that a non-correctable error was detected in a memory read initiated by FlexCAN internal processes. No interrupt is associated to this flag. 0b - No non-correctable errors were detected in FlexCAN accesses so far. 1b - A non-correctable error was detected in a FlexCAN access.
17 —	Reserved
16 CEIF	Correctable Error Interrupt Flag Indicates that a correctable error was detected in a memory read. 0b - No correctable errors were detected so far. 1b - A correctable error was detected.
15-4 —	Reserved
3 HANCEIOF	Host Access With Non-Correctable Error Interrupt Overrun Flag Indicates that a non-correctable error was detected in a memory read initiated by host when HANCEIF was set. No interrupt is associated with this flag. See Error indication. 0b - No overrun on non-correctable errors in host access 1b - Overrun on non-correctable errors in host access
2 FANCEIOF	FlexCAN Access With Non-Correctable Error Interrupt Overrun Flag Indicates that a non-correctable error was detected in a memory read initiated by FlexCAN internal processes when FANCEIF was set. No interrupt is associated to this flag. See Error indication. 0b - No overrun on non-correctable errors in FlexCAN access 1b - Overrun on non-correctable errors in FlexCAN access
1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 CEIOF	Correctable Error Interrupt Overrun Flag Indicates that a correctable error was detected in a memory read when CEIF was set. No interrupt is associated to this flag. See Error indication. 0b - No overrun on correctable errors 1b - Overrun on correctable errors

50.4.2.33 Enhanced CAN Bit Timing Prescalers (EPRS)

Offset

Register	Offset
EPRS	BF0h

Function

This register defines the CAN bit timing prescalers for the nominal phase and data phase when CTRL2[BTE] is set.

This register is used by the hardware only if CTRL2[BTE] is set; otherwise a write operation has no effect and all fields are read as zero.

This register can be written only in Freeze mode because it is blocked by hardware in other modes.

The contents of this register are not affected by soft reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								EDPRESDIV							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								ENPRESDIV							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4848-b251-390d8ab624c1

Table continued from the previous page...

Field	Function
25-16 EDPRESDIV	Extended Data Phase Prescaler Division Factor This field defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency in the data phase of a CAN FD message when CTRL2[BTE] is set. The Sclock period defines the time quantum of the CAN FD protocol for the data bit rate. Sclock frequency = PE clock frequency / (EDPRESDIV + 1). **NOTE** To minimize errors when processing FD frames, use the same value for ENPRESDIV and EDPRESDIV. For more details see the first NOTE in section CAN FD frames.
15-10 —	Reserved
9-0 ENPRESDIV	Extended Nominal Prescaler Division Factor This field defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency when CTRL2[BTE] is set; otherwise it reads as zero and a write operation has no effect. The Sclock period defines the time quantum of the CAN protocol in the nominal phase. For the reset value, the Sclock frequency is equal to the PE clock frequency (see Protocol timing). Sclock frequency = PE clock frequency / (ENPRESDIV + 1)

50.4.2.34 Enhanced Nominal CAN Bit Timing (ENCBT)

Offset

Register	Offset
ENCBT	BF4h

Function

This register provides an alternative way to store the CAN bit timing variables described in the CTRL1 and CBT registers, to get higher CAN bit timing resolution.

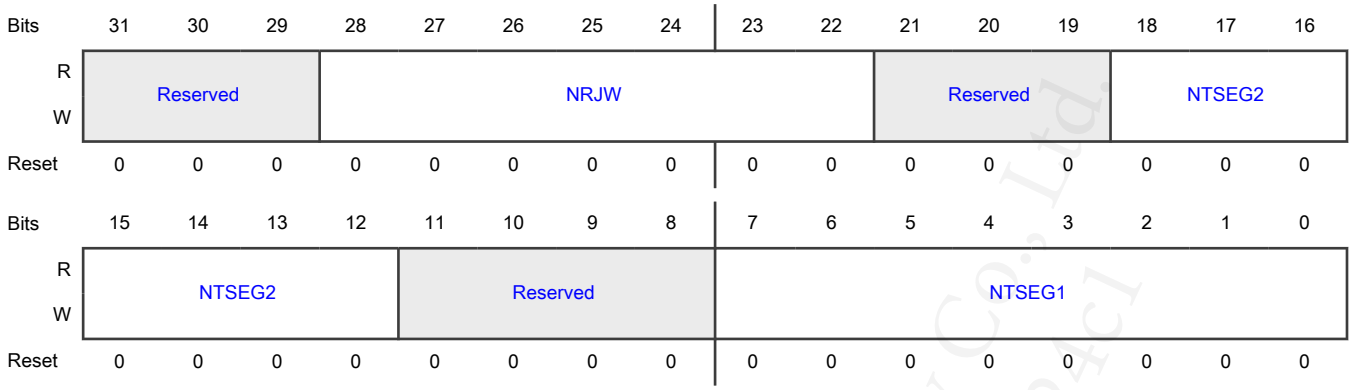
This register is used by the hardware only if CTRL2[BTE] is set; otherwise a write operation has no effect and all fields are read as zero.

The contents of this register are not affected by soft reset.

This register can be written only in Freeze mode because it is blocked by hardware in other modes.

Preliminary NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co. Ltd.
c8c462c-4821-48a8-b251-9d3ab6b4c1

Diagram



Fields

Field	Function
31-29 —	Reserved
28-22 NRJW	Nominal Resynchronization Jump Width This field defines the maximum number of time quanta that a nominal bit time can be changed by one resynchronization when CTRL2[BTE] bit is asserted; otherwise it has no effect. One time quantum is equal to the Sclock period. Nominal Resync Jump Width = NRJW + 1.
21-19 —	Reserved
18-12 NTSEG2	Nominal Time Segment 2 This field defines the length of Time Segment 2 in the nominal bit time when CTRL2[BTE] is set; otherwise it has no effect. Nominal Time Segment 2 = (NTSEG2 + 1) × Time-Quanta. Time-Quantum = one Sclock period.
11-8 —	Reserved
7-0 NTSEG1	Nominal Time Segment 1 This field defines the length of Time Segment 1 in the bit time when CTRL2[BTE] is set; otherwise it has no effect. Nominal Time Segment 1 = (NTSEG1 + 1) × Time-Quanta. Time-Quantum = one Sclock period.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c6225-4821-48a8-b251-390d3ab624c1

50.4.2.35 Enhanced Data Phase CAN bit Timing (EDCBT)

Offset

Register	Offset
EDCBT	BF8h

Function

This register provides an alternative way to store the data phase CAN bit timing variables described in the FDCBT register to achieve higher CAN bit timing resolution.

This register is used by the hardware only if CTRL2[BTE] is set; otherwise a write operation has no effect and all fields are read as zero.

The contents of this register are not affected by soft reset.

This register can be written only in Freeze mode because it is blocked by hardware in other modes.

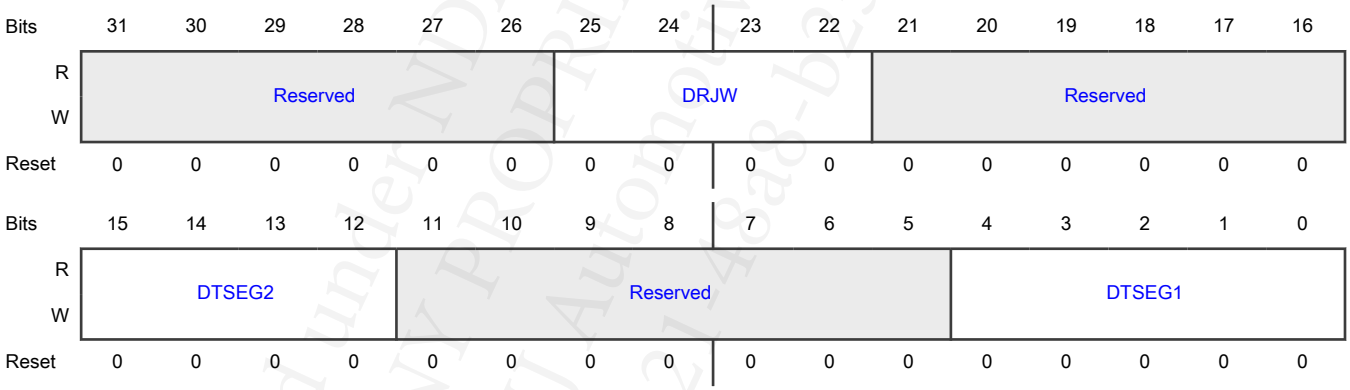
NOTE

The user must ensure bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1).

NOTE

DTSEG1 must be at least two time quanta.

Diagram



Fields

Field	Function
31-26 —	Reserved
25-22 DRJW	Data Phase Resynchronization Jump Width This field defines the maximum number of time quanta that a data phase bit time can be changed by one resynchronization when CTRL2[BTE] is set; otherwise it has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Data Phase Resync Jump Width = DRJW + 1.
21-16 —	Reserved
15-12 DTSEG2	Data Phase Time Segment 2 This field defines the length of time segment 2 in the data phase bit time when CTRL2[BTE] is set; otherwise it has no effect. Data Phase Time Segment 2 = (DTSEG2 + 1) x Time-Quanta. Time-Quantum = one Sclock period.
11-5 —	Reserved
4-0 DTSEG1	Data Phase Segment 1 This field defines the length of time segment 1 in the data phase bit time when CTRL2[BTE] is set; otherwise it has no effect. Data Phase Time Segment 1 = (NTSEG1 + 1) x Time-Quanta. Time-Quantum = one Sclock period.

50.4.2.36 Enhanced Transceiver Delay Compensation (ETDC)

Offset

Register	Offset
ETDC	BFCh

Function

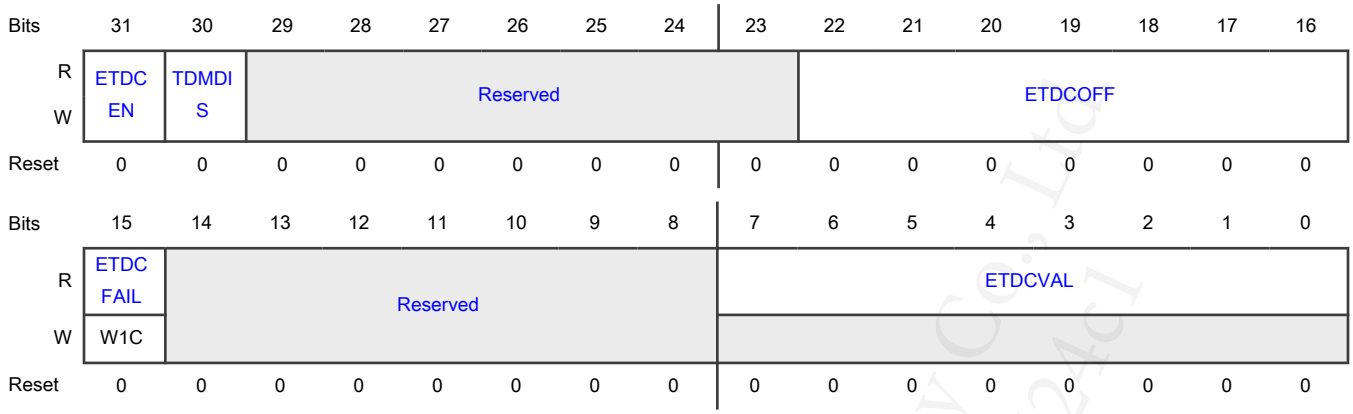
This register contains extended versions of the fields FDCTRL[TDCOFF] and FDCTRL[TDCVAL]. This register is used by the hardware only if CTRL2[BTE] is set; otherwise, a write operation has no effect and all fields are read as zero.

NOTE

See Transmitter delay compensation in the CAN Protocol standard (ISO 11898-1) for details.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4e01-48a8-b251-390d3ab624f1

Diagram



Fields

Field	Function
31 ETDCEN	Transceiver Delay Compensation Enable This bit can be used for enabling and disabling the TDC feature. It can be written in Freeze mode only. **NOTE** Refer to Transmitter delay compensation in the CAN Protocol standard (ISO 11898-1) for details. **NOTE** TDC must be disabled when the Loop Back Mode is enabled (see CTRL1[LPB] register). 0b - TDC is disabled 1b - TDC is enabled
30 TDMDIS	Transceiver Delay Measurement Disable This bit can be used for disabling the transceiver delay measurement. When the TDC measurement is disabled, the secondary sample point position is determined only by the enhanced TDC offset (ETDCOFF) field. Otherwise, if the TCD measurement is enabled, then the secondary sample point position is determined by the sum of the transceiver delay measurement plus the enhanced TDC offset. This field is not affected by soft reset. **NOTE** This bit can be enabled only if CTRL2[BTE] is set. 0b - TDC measurement is enabled 1b - TDC measurement is disabled
29-23 —	Reserved
22-16 ETDCOFF	Enhanced Transceiver Delay Compensation Offset

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field contains the offset value to be added to the measured transceiver's loop delay in order to define the position of the delayed comparison point when bit rate switching is active. See Transceiver delay compensation for more details on how the loop delay measurement is performed. ETDCOFF can be written in Freeze mode only. Its value can be defined in protocol engine (PE) clock periods (CANCLK, see Protocol timing for more details), and must be selected to be smaller than the CAN bit duration in the data bit rate for proper operation. **NOTE** ETDCOFF must not be configured as zero. **NOTE** If CTRL2[BTE] is set after a chip level hard reset, ETCDOFF is read as 1h.
15 ETDCFAIL	Transceiver Delay Compensation Fail This bit indicates when the Transceiver Delay Compensation (TDC) mechanism is out of range, unable to compensate the transceiver's loop delay and successfully compare the delayed received bits to the transmitted ones (see Transceiver delay compensation). ETDCFAIL sets in the first time FlexCAN detects the out of range condition. The CPU needs to write 1 to clear it. 0b - Measured loop delay is in range. 1b - Measured loop delay is out of range.
14-8 —	Reserved
7-0 ETDCVAL	Enhanced Transceiver Delay Compensation Value This register contains the ETDCOFF field added to the measured value of the transceiver loop delay in the latest transmitted CAN FD frame, with BRS equal to recessive. This field is only updated by the hardware if ETDC[ETDCEN] is set. This field is affected by soft reset. **NOTE** If TDMDIS is set, ETDCVAL stores ETDCOFF only.

50.4.2.37 CAN FD Control register (FDCTRL)

Offset

Register	Offset
FDCTRL	C00h

Function

This register contains control bits for the CAN FD operation. It also defines the data size of message buffers allocated in different partitions of RAM (memory blocks) as described in the table below.

When an 8-byte payload is selected:

- Block R0 allocates MB0 to MB31.
- Block R1 allocates MB32 to MB63.
- Block R2 allocates MB64 to MB95.
- Block R3 allocates MB96 to MB127.

When a payload larger than 8 bytes is selected, the maximum number of MBs in a block is limited as described below:

Table 343. Number of message buffers

Payload size	Maximum number of message buffers per RAM block
8 bytes	32
16 bytes	21
32 bytes	12
64 bytes	7

NOTE

One memory block fits exactly 32 MBs with an 8-byte payload. For other possible payload sizes, empty memory may exist between the last MB in a block and the beginning of the next block. This empty memory corresponds to less than one MB, and must not be used.

The contents of this register are not affected by soft reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FDRATE	Reserved				MBDSR3		Reserved	MBDSR2		Reserved	MBDSR1		Reserved	MBDSR0	
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TDCEN	TDCFAIL	Reserved	TDCOFF					Reserved		TDCVAL					
W		W1C														
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
31 FDRATE	Bit Rate Switch Enable This bit enables the effect of the Bit Rate Switch (BRS bit) during the data phase of Tx messages. The CPU can write this bit any time. However, its effect turns active only when the CAN bus is in Wait for Bus Idle, Bus Idle, or Bus Off state, or when the current frame under reception or transmission reaches the interframe space.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	By negating FDCTRL[FDRATE], the CPU can force all bits in CAN FD messages to be transmitted in nominal bit rate, no matter what the value is in the BRS bit of the Tx MBs. 0b - Transmit a frame in nominal rate. The BRS bit in the Tx MB has no effect. 1b - Transmit a frame with bit rate switching if the BRS bit in the Tx MB is recessive.
30-27 —	Reserved
26-25 MBDSR3	Message Buffer Data Size for Region 3 This two-bit field selects the data size (8, 16, 32, or 64 bytes) for region R3 of message buffers allocated in RAM. It can be written in Freeze mode only. 00b - Selects 8 bytes per message buffer. 01b - Selects 16 bytes per message buffer. 10b - Selects 32 bytes per message buffer. 11b - Selects 64 bytes per message buffer.
24 —	Reserved
23-22 MBDSR2	Message Buffer Data Size for Region 2 This two-bit field selects the data size (8, 16, 32, or 64 bytes) for region R2 of message buffers allocated in RAM. It can be written in Freeze mode only. 00b - Selects 8 bytes per message buffer. 01b - Selects 16 bytes per message buffer. 10b - Selects 32 bytes per message buffer. 11b - Selects 64 bytes per message buffer.
21 —	Reserved
20-19 MBDSR1	Message Buffer Data Size for Region 1 This two-bit field selects the data size (8, 16, 32, or 64 bytes) for region R1 of message buffers allocated in RAM. It can be written in Freeze mode only. 00b - Selects 8 bytes per message buffer. 01b - Selects 16 bytes per message buffer.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	10b - Selects 32 bytes per message buffer. 11b - Selects 64 bytes per message buffer.
18 —	Reserved
17-16 MBDSR0	Message Buffer Data Size for Region 0 This two-bit field selects the data size (8, 16, 32, or 64 bytes) for region R0 of message buffers allocated in RAM. It can be written in Freeze mode only. 00b - Selects 8 bytes per message buffer. 01b - Selects 16 bytes per message buffer. 10b - Selects 32 bytes per message buffer. 11b - Selects 64 bytes per message buffer.
15 TDCEN	Transceiver Delay Compensation Enable This bit can be used to enable and disable the TDC feature. It can be written in Freeze mode only. **NOTE** See Transmitter delay compensation in the CAN Protocol standard (ISO 11898-1) for details. **NOTE** TDC must be disabled when Loop Back mode is enabled (see CTRL1[LPB]). **NOTE** If CTRL2[BTE] is set, TDCEN is read as zero and a write operation has no effect. 0b - TDC is disabled 1b - TDC is enabled
14 TDCFAIL	Transceiver Delay Compensation Fail This bit indicates when the Transceiver Delay Compensation (TDC) mechanism is out of range, unable to compensate the transceiver's loop delay and successfully compare the delayed received bits to the transmitted ones (see Transceiver delay compensation). TDCFAIL sets in the first time FlexCAN detects the out of range condition. The CPU needs to write one to clear it. **NOTE** If CTRL2[BTE] is set, TDCFAIL is read as zero and a write operation has no effect. 0b - Measured loop delay is in range. 1b - Measured loop delay is out of range.
13	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
12-8 TDCOFF	Transceiver Delay Compensation Offset This bit field contains the offset value to be added to the measured transceiver's loop delay in order to define the position of the delayed comparison point when bit rate switching is active. See Transceiver delay compensation for more details on how the loop delay measurement is performed. TDCOFF can be written in Freeze mode only. Its value can be defined in Protocol Engine (PE) Clock periods (CANCLK, see Protocol timing for more details), and must be selected to be smaller than the CAN bit duration in the data bit rate for proper operation. **NOTE** If CTRL2[BTE] is set, TDCOFF is read as zero and a write operation has no effect. **NOTE** It is not recommended to have TDCOFF equal to zero.
7-6 —	Reserved
5-0 TDCVAL	Transceiver Delay Compensation Value This register contains the value of the transceiver loop delay measured from the transmitted EDL to R0 transition edge to the respective received one added to the TDCOFF value specified in the FDCTRL register. This value is an integer multiple of the Protocol Engine (PE) Clock period (CANCLK). **NOTE** If CTRL2[BTE] is set, TDCVAL is read as zero. See Protocol timing for more details on how the loop delay measurement is performed.

50.4.2.38 CAN FD Bit Timing register (FDCBT)

Offset

Register	Offset
FDCBT	C04h

Function

This register stores the CAN bit timing variables used in the data phase of CAN FD messages when the FDCTRL[FDRATE] is set, compatible with CAN FD specification. FPRESDIV, FPROPSEG, FPSEG1, FPSEG2, and FRJW are used to define the time quantum duration, the number of time quanta per CAN bit, and the sample point position for the data bit rate portion of a CAN FD message with the BRS bit set.

The contents of this register are not affected by soft reset.

NOTE

The sum of the Fast Propagation Segment (FPROPSEG) and Fast Phase Segment 1 (FPSEG1) must be at least two time quanta.

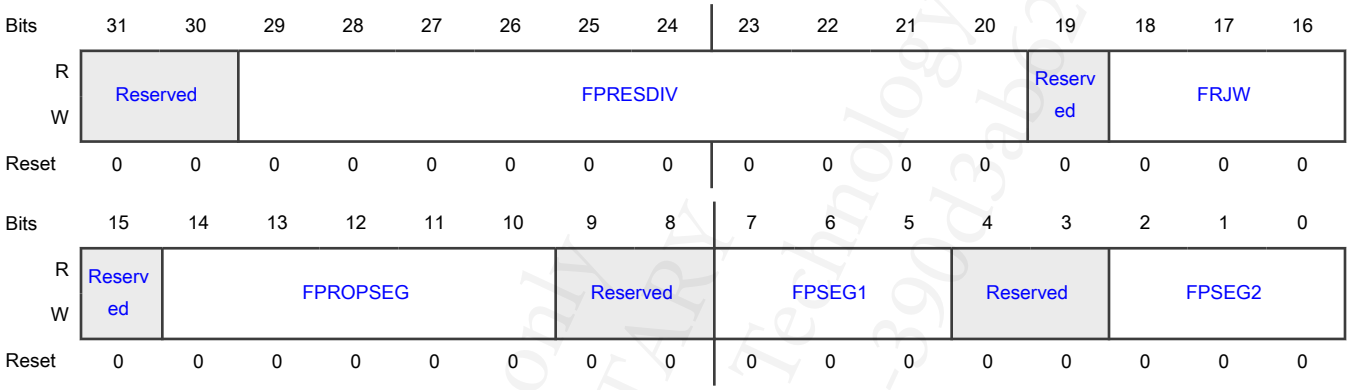
NOTE

The user must ensure bit time settings and protocol engine tolerance are in compliance with the CAN Protocol standard (ISO 11898-1).

NOTE

If CTRL2[BTE] is set, the FDCBT register is read as zero and a write operation has no effect.

Diagram



Fields

Field	Function
31-30 —	Reserved
29-20 FPRESDIV	Fast Prescaler Division Factor This 10-bit field defines the ratio between the PE clock frequency and the serial clock (Sclock) frequency in the data bit rate portion of a CAN FD message with the BRS bit set. The Sclock period defines the time quantum of the CAN FD protocol for the data bit rate. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Sclock frequency = PE clock frequency / (FPRESDIV + 1). **NOTE** To minimize errors when processing FD frames, use the same value for FPRESDIV and PRESDIV (in CBT or CTRL1). For more details see the first NOTE in section CAN FD frames.
19 —	Reserved
18-16	Fast Resync Jump Width

Table continues on the next page...

Table continued from the previous page...

Field	Function
FRJW	This 3-bit field defines the maximum number of time quanta that a bit time can be changed by one resynchronization in the data bit rate portion of a CAN FD message with the BRS bit set. One time quantum is equal to the Sclock period. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Resync Jump Width = FSJW + 1.
15 —	Reserved
14-10 FPROPSEG	Fast Propagation Segment This 5-bit field defines the length of the propagation segment in the bit time in the data bit rate portion of a CAN FD message with the BRS bit set. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Propagation Segment Time = FPROPSEG × Time-Quanta. Time-Quantum = one Sclock period.
9-8 —	Reserved
7-5 FPSEG1	Fast Phase Segment 1 This 3-bit field defines the length of phase segment 1 in the bit time in the data bit rate portion of a CAN FD message with the BRS bit set. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Segment 1 = (FPSEG1 + 1) × Time-Quanta. Time-Quantum = one Sclock period.
4-3 —	Reserved
2-0 FPSEG2	Fast Phase Segment 2 This 3-bit field defines the length of phase segment 2 in the data bit rate portion of a CAN FD message with the BRS bit set. This field can be written only in Freeze mode because it is blocked by hardware in other modes. Phase Segment 2 = (FPSEG2 + 1) × Time-Quanta. Time-Quantum = one Sclock period.

50.4.2.39 CAN FD CRC register (FDCRC)

Offset

Register	Offset
FDCRC	C08h

Function

This register provides information about the CRC of transmitted messages.

FlexCAN uses different CRC polynomials for different frame formats, as shown below.

The CRC_15 polynomial is used for all frames in CAN format. The CRC_17 polynomial is used for frames in CAN FD format with a DATA FIELD up to sixteen bytes. The CRC_21 polynomial is used for frames in CAN FD format with a DATA FIELD longer than sixteen bytes. Each polynomial shown below results in a Hamming distance of 6. This register is updated at the same time the Tx Interrupt flag is asserted.

CRC_15 = C599h: $(x^{15} + x^{14} + x^{10} + x^8 + x^7 + x^4 + x^3 + 1)$

CRC_17 = 3685Bh: $(x^{17} + x^{16} + x^{14} + x^{13} + x^{11} + x^6 + x^4 + x^3 + x^1 + 1)$

CRC_21 = 302899h: $(x^{21} + x^{20} + x^{13} + x^{11} + x^7 + x^4 + x^3 + 1)$

NOTE

See CRC sequence calculation in the CAN Protocol standard (ISO 11898-1) for details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	FD_MBCRC							Reserved			FD_TXCRC				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FD_TXCRC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-24 FD_MBCRC	CRC Mailbox Number for FD_TXCRC This field indicates the number of the mailbox corresponding to the value in the FD_TXCRC field, for both FD and non-FD frames. It reports the same information as in CRCR[MBCRC].
23-21 —	Reserved
20-0	Extended Transmitted CRC value

Table continues on the next page...

Table continued from the previous page...

Field	Function
FD_TXCRC	This 21-bit field contains the CRC value calculated over the most recent transmitted message. Different CRC polynomials are used for different frame formats. A 15-bit polynomial, CRC_15, is used for all frames in CAN format. The second 17-bit polynomial, CRC_17, is used for frames in CAN FD format with a data field up to sixteen bytes long. The third 21-bit polynomial, CRC_21, is used for frames in CAN FD format with a data field longer than sixteen bytes. For CRC_15 and CRC_17, the 6 most significant bits and the 4 most significant bits are reported as zeros, respectively. For CRC_15, this register has the same content as CRC register.

50.4.2.40 Enhanced Rx FIFO Control register (ERFCR)

Offset

Register	Offset
ERFCR	C0Ch

Function

This register defines the Enhanced Rx FIFO configuration.

This register can be written only in Freeze mode.

All the contents of this register are affected by soft reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ERFEN	DMALW					Reserved		NEXIF							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		NFE						Reserved			ERFWM				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ERFEN	Enhanced Rx FIFO enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field enables the Enhanced Rx FIFO. **NOTE** ERFEN must not be set if MCR[RFEN] is set. 0b - Enhanced Rx FIFO is disabled 1b - Enhanced Rx FIFO is enabled
30-26 DMALW	DMA Last Word This field defines the last DMA address for each Enhanced RX FIFO element. The table shows the number of elements and the last address for each Enhanced Rx FIFO element according to the value of DMALW.

Table 344. Last address and number of words in Enhanced RX FIFO

DMALW	Number of 32-bit words transferred	Last FIFO address
0	1	2000h
1	2	2004h
2	3	2008h
3	4	200Ch
4	5	2010h
5	6	2014h
6	7	2018h
7	8	201Ch
8	9	2020h
9	10	2024h
10	11	2028h
11	12	202Ch
12	13	2030h
13	14	2034h
14	15	2038h
15	16	203Ch
16	17	2040h
17	18	2044h
18	19	2048h
19	20	204Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4821-4a8-b251-390a3ab624c1

Table continued from the previous page...

Field	Function
	<table><tr><th>DMALW</th><th>Number of 32-bit words transferred</th><th>Last FIFO address</th></tr></table> **NOTE** Undefined DMALW values in the table are reserved and must not be used.
25-23 —	Reserved
22-16 NEXIF	Number of Extended ID Filter Elements This field defines the number of extended ID filter elements used during the Enhanced Rx FIFO matching process. NEXIF must satisfy this condition: NEXIF ≤ NFE + 1 The number of standard ID filter elements can be determined by this equation. Number of Standard ID Filter Elements = 2 x (NFE - NEXIF +1) The table shows the number of extended ID filters and standard ID filters available for Enhanced Rx FIFO if all the filter elements are used. **Table 345. Number of Extended ID filters and Standard ID filters for Enhanced Rx FIFO** <table><tr><th>NEXIF</th><th>NFE</th><th>Number of Extended ID filter elements</th><th>Number of Standard ID filter elements</th></tr><tr><td>0</td><td>63</td><td>0</td><td>128</td></tr><tr><td>1</td><td>63</td><td>1</td><td>126</td></tr><tr><td>2</td><td>63</td><td>2</td><td>124</td></tr><tr><td>3</td><td>63</td><td>3</td><td>122</td></tr><tr><td>4</td><td>63</td><td>4</td><td>120</td></tr><tr><td>5</td><td>63</td><td>5</td><td>118</td></tr><tr><td>6</td><td>63</td><td>6</td><td>116</td></tr><tr><td>7</td><td>63</td><td>7</td><td>114</td></tr><tr><td>8</td><td>63</td><td>8</td><td>112</td></tr><tr><td>9</td><td>63</td><td>9</td><td>110</td></tr><tr><td>10</td><td>63</td><td>10</td><td>108</td></tr><tr><td>11</td><td>63</td><td>11</td><td>106</td></tr></table>

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c6c462e5482148ab2b1590d3ab624d1

Table continued from the previous page...

Field	Function

Table 345. Number of Extended ID filters and Standard ID filters for Enhanced Rx FIFO (continued)

NEXIF	NFE	Number of Extended ID filter elements	Number of Standard ID filter elements
12	63	12	104
13	63	13	102
14	63	14	100
15	63	15	98
16	63	16	96
17	63	17	94
18	63	18	92
19	63	19	90
20	63	20	88
21	63	21	86
22	63	22	84
23	63	23	82
24	63	24	80
25	63	25	78
26	63	26	76
27	63	27	74
28	63	28	72
29	63	29	70
30	63	30	68
31	63	31	66
32	63	32	64
33	63	33	62
34	63	34	60
35	63	35	58
36	63	36	56
37	63	37	54
38	63	38	52
39	63	39	50
40	63	40	48

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Table 345. Number of Extended ID filters and Standard ID filters for Enhanced Rx FIFO (continued)

NEXIF	NFE	Number of Extended ID filter elements	Number of Standard ID filter elements
41	63	41	46
42	63	42	44
43	63	43	42
44	63	44	40
45	63	45	38
46	63	46	36
47	63	47	34
48	63	48	32
49	63	49	30
50	63	50	28
51	63	51	26
52	63	52	24
53	63	53	22
54	63	54	20
55	63	55	18
56	63	56	16
57	63	57	14
58	63	58	12
59	63	59	10
60	63	60	8
61	63	61	6
62	63	62	4
63	63	63	2
64	63	64	0

15-14 —	Reserved
13-8 NFE	Number of Enhanced Rx FIFO Filter Elements

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field defines the total number of filter elements used during the enhanced RX FIFO matching process according to the table.

Table 346. Total number of Enhanced Rx FIFO filters

NFE	Maximum number of Extended ID filter elements (NEXIF = NFE + 1)	Maximum number of standard ID filter elements (NEXIF = 0)
0	1	2
1	2	4
2	3	6
3	4	8
4	5	10
5	6	12
6	7	14
7	8	16
8	9	18
9	10	20
10	11	22
11	12	24
12	13	26
13	14	28
14	15	30
15	16	32
16	17	34
17	18	36
18	19	38
19	20	40
20	21	42
21	22	44
22	23	46
23	24	48
24	25	50
25	26	52
26	27	54

Table continues on the next page...

Table continued from the previous page...

Field	Function

Table 346. Total number of Enhanced Rx FIFO filters (continued)

NFE	Maximum number of Extended ID filter elements (NEXIF = NFE + 1)	Maximum number of standard ID filter elements (NEXIF = 0)
27	28	56
28	29	58
29	30	60
30	31	62
31	32	64
32	33	66
33	34	68
34	35	70
35	36	72
36	37	74
37	38	76
38	39	78
39	40	80
40	41	82
41	42	84
42	43	86
43	44	88
44	45	90
45	46	92
46	47	94
47	48	96
48	49	98
49	50	100
50	51	102
51	52	104
52	53	106
53	54	108
54	55	110
55	56	112

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Table 346. Total number of Enhanced Rx FIFO filters (continued)

NFE	Maximum number of Extended ID filter elements (NEXIF = NFE + 1)	Maximum number of standard ID filter elements (NEXIF = 0)
56	57	114
57	58	116
58	59	118
59	60	120
60	61	122
61	62	124
62	63	126
63	64	128

7-5 —	Reserved
4-0 ERFWM	Enhanced Rx FIFO Watermark This field defines the minimum number of CAN messages stored in the Enhanced RX FIFO. When that number is reached, then ERFSR[ERFWMI] is set. Minimum number of CAN messages = ERFWM + 1.

50.4.2.41 Enhanced Rx FIFO Interrupt Enable register (ERFIER)

Offset

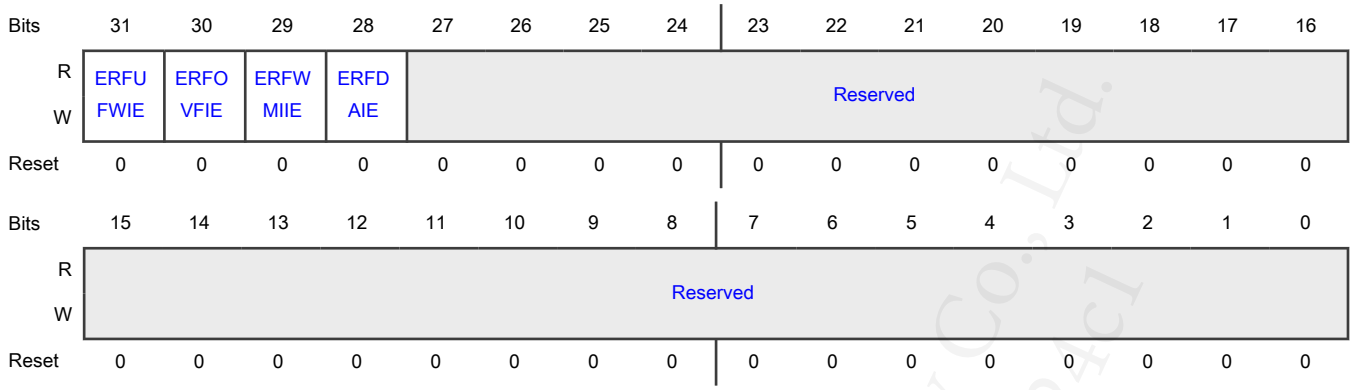
Register	Offset
ERFIER	C10h

Function

This register contains the interrupt enables for the Enhanced Rx FIFO.

This register is affected by soft reset.

Diagram



Fields

Field	Function
31 ERFUFWIE	Enhanced Rx FIFO Underflow Interrupt Enable This field enables interrupt for ERFSR[ERFUFW]. 0b - Enhanced Rx FIFO Underflow interrupt is disabled 1b - Enhanced Rx FIFO Underflow interrupt is enabled
30 ERFOVFIE	Enhanced Rx FIFO Overflow Interrupt Enable This field enables interrupt for ERFSR[ERFOVF]. 0b - Enhanced Rx FIFO Overflow is disabled 1b - Enhanced Rx FIFO Overflow is enabled
29 ERFWMIIE	Enhanced Rx FIFO Watermark Indication Interrupt Enable This field enables interrupt for ERFSR[ERFWMI]. 0b - Enhanced Rx FIFO Watermark interrupt is disabled 1b - Enhanced Rx FIFO Watermark interrupt is enabled
28 ERFDAIE	Enhanced Rx FIFO Data Available Interrupt Enable This field enables interrupt for ERFSR[ERFDA]. 0b - Enhanced Rx FIFO Data Available interrupt is disabled 1b - Enhanced Rx FIFO Data Available interrupt is enabled
27-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHANGHONG Electronic Technology Co., Ltd.
c8c462e5-4381-48a8-b251-390d3ab624c1

50.4.2.42 Enhanced Rx FIFO Status register (ERFSR)

Offset

Register	Offset
ERFSR	C14h

Function

This register contains the status bits of the Enhanced RX FIFO including error indications and a clear FIFO bit.

This register is affected by soft reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ERFUFW	ERFOVF	ERFWMI	ERFDA	0	Reserved									ERFE	ERFF
W	W1C	W1C	W1C	W1C	ERFCLR											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										ERFEL					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ERFUFW	Enhanced Rx FIFO Underflow This field indicates that an underflow condition occurred in the enhanced Rx FIFO. This field generates an interrupt if ERFIER[ERFUFWIE] is set. 0b - No such occurrence 1b - Enhanced Rx FIFO underflow
30 ERFOVF	Enhanced Rx FIFO Overflow This field indicates that an overflow condition occurred in the Enhanced Rx FIFO. This field generates an interrupt if ERFIER[ERFOVFIE] is set. 0b - No such occurrence 1b - Enhanced Rx FIFO overflow
29 ERFWMI	Enhanced Rx FIFO Watermark Indication

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is set by the hardware if the number of messages available in the Enhanced Rx FIFO is greater than the watermark defined in ERFCR[ERFWM]. This field generates an interrupt if ERFIER[ERFWMIIE] is set. 0b - No such occurrence 1b - The number of messages in FIFO is greater than the watermark
28 ERFDA	Enhanced Rx FIFO Data Available This field is set by the hardware when there is at least one message stored in the ERX FIFO. This field generates an interrupt if ERFIER[ERFDAIE] is set. 0b - No such occurrence 1b - There is at least one message stored in Enhanced Rx FIFO
27 ERFCLR	Enhanced Rx FIFO Clear Writing one to this field during Freeze mode clears Enhanced Rx FIFO content. Writing out of Freeze mode or writing zero to this field has no effect. 0b - No effect 1b - Clear Enhanced Rx FIFO content
26-18 —	Reserved
17 ERFE	Enhanced Rx FIFO empty This field indicates that Enhanced Rx FIFO is empty. 0b - Enhanced Rx FIFO is not empty 1b - Enhanced Rx FIFO is empty
16 ERFF	Enhanced Rx FIFO full This field indicates that Enhanced Rx FIFO is full. 0b - Enhanced Rx FIFO is not full 1b - Enhanced Rx FIFO is full
15-6 —	Reserved
5-0 ERFEL	Enhanced Rx FIFO Elements This field indicates the number of CAN messages stored in the Enhanced Rx FIFO.

Provided under NDA only
COMPANY PROPRIETARY
Beijing C-I Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39d8ab624c1

50.4.2.43 High Resolution Time Stamp (HR_TIME_STAMP0 - HR_TIME_STAMP127)

Offset

For n = 0 to 127:

Register	Offset
HR_TIME_STAMPn	C30h + (n × 4h)

Function

The High Resolution Time Stamp registers are used for storing a copy of a 32-bit timer at the start or end of a CAN frame.

HRST0 stores the 32-bit time stamp associated with MB0, HRSTS1 stores the the 32-bit time stamp associated with MB1, and so on.

These registers are not affected by reset.

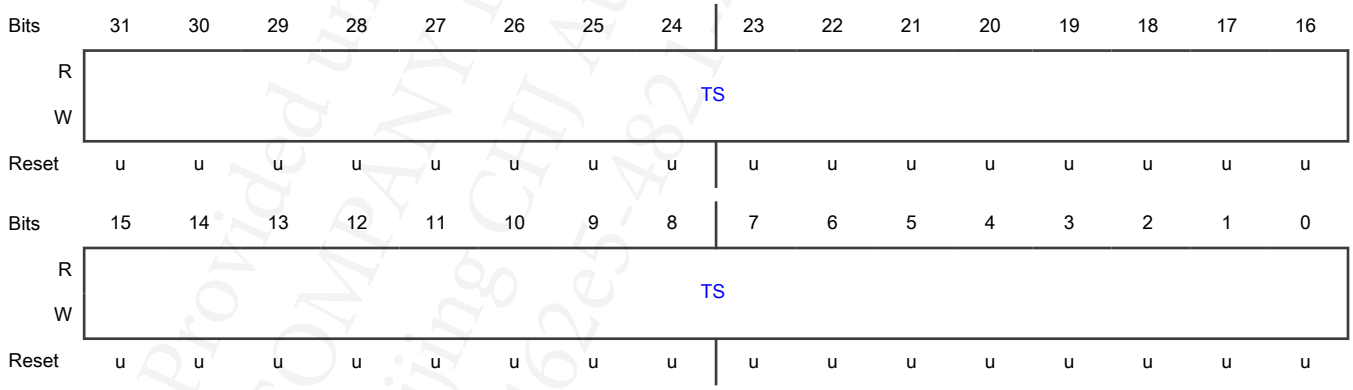
NOTE

These registers must not be written out of Freeze mode.

Table 347. High Resolution Time Stamp register operation

TSTAMPCAP	Captured time base	Capture point
b00	None	None
b01	32 bits of high-resolution on-chip timer	Seventh bit of the end of frame field for transmission and sixth bit of the end of frame field for reception.
b10	32 bits of high-resolution on-chip timer	Start of frame.
b11	32 bits of high-resolution on-chip timer	Start of frame for classical CAN frame format and res bit for CAN FD frame format.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e54821148a81025f1d0d3ab6f4c1

Fields

Field	Function
31-0 TS	High Resolution Time Stamp HR_TIME_STAMP register which captures the copy of timestamp of corresponding MB always captures 32-bit timer value.

50.4.2.44 Enhanced Rx FIFO Filter Element (ERFFEL0 - ERFFEL127)

Offset

For n = 0 to 127:

Register	Offset
ERFFELn	3000h + (n × 4h)

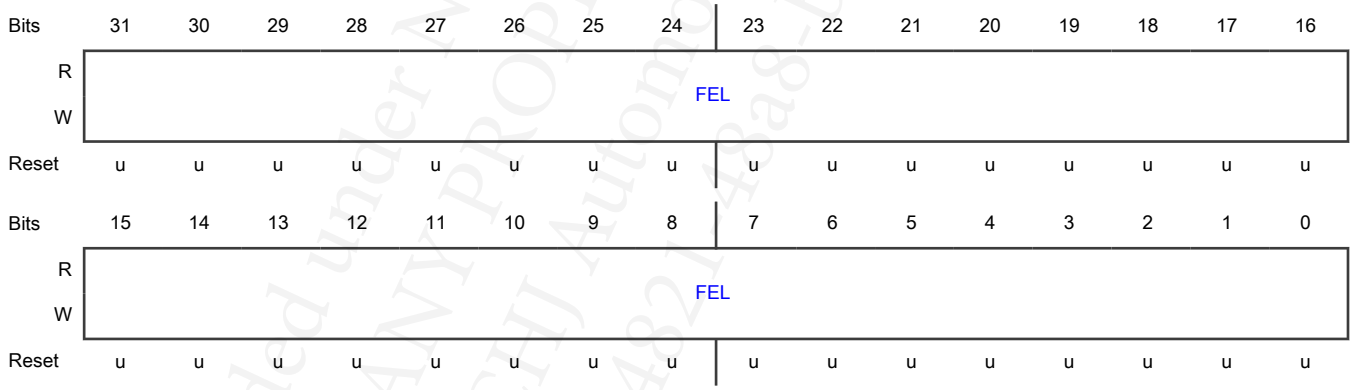
Function

These registers are used for storing the filter elements of the Enhanced Rx FIFO.

For standard ID filtering, each ERFFEL register stores one filter element. For extended ID filtering, each pair of ERFFEL registers stores one filter element.

ERFFEL registers can be written only in Freeze mode; otherwise, they are blocked by hardware. These registers are not affected by reset. They are located in RAM and must be explicitly initialized prior to any reception.

Diagram



Fields

Field	Function
31-0 FEL	Filter Element Bits Each filter element is used during the match process and, if the matching criteria are met, a message is stored in the Enhanced Rx FIFO.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-8a8-b23-590d3ab624c1

50.4.3 Message buffer structure

The message buffer structure used by the FlexCAN module is represented in the following figure. Both extended (29-bit identifier) and standard (11-bit identifier) frames used in the CAN specification (Version 2.0 Part B) are represented. Each individual MB is formed by 16, 24, 40, or 72 bytes, depending on the quantity of data bytes allocated for the message payload: 8, 16, 32, or 64 data bytes, respectively.

The memory area 80h–87Fh is used by the mailboxes. When CAN FD is enabled, the exact address for each MB depends on the size of its payload. See FlexCAN memory partition for CAN FD for more detailed information.

When CAN FD is enabled, the memory area 1000h–17FFh can be used for implementing up to 28 64-byte additional MBs. This additional MB region (enhanced MB memory) can be used partially or completely only if the total number of MBs configured by MCR[MAXMB] is less than 128. This means that the maximum number of MBs in the area 80h–87FH and area 1000h–17FFH is 128. See FlexCAN memory partition for CAN FD for more detailed information.

Table 348. Message buffer structure — example with 64-byte payload

	31	30	29	28	27	24	23	22	21	20	19	18	17	16	15	8	7	0
0h	EDL	BRS	ESI		CODE			SRR	IDE	RTR	DLC				TIME STAMP			
4h	PRIO			ID (standard/extended)											ID (extended)			
8h	Data byte 0						Data byte 1								Data byte 2		Data byte 3	
Ch	Data byte 4						Data byte 5								Data byte 6		Data byte 7	
10h	Data byte 8						Data byte 9								Data byte 10		Data byte 11	
14h	Data byte 12						Data byte 13								Data byte 14		Data byte 15	
18h	Data byte 16						Data byte 17								Data byte 18		Data byte 19	
1Ch	Data byte 20						Data byte 21								Data byte 22		Data byte 23	
20h	Data byte 24						Data byte 25								Data byte 26		Data byte 27	
24h	Data byte 28						Data byte 29								Data byte 30		Data byte 31	
28h	Data byte 32						Data byte 33								Data byte 34		Data byte 35	
2Ch	Data byte 36						Data byte 37								Data byte 38		Data byte 39	
30h	Data byte 40						Data byte 41								Data byte 42		Data byte 43	
34h	Data byte 44						Data byte 45								Data byte 46		Data byte 47	
38h	Data byte 48						Data byte 49								Data byte 50		Data byte 51	
3Ch	Data byte 52						Data byte 53								Data byte 54		Data byte 55	
40h	Data byte 56						Data byte 57								Data byte 58		Data byte 59	
44h	Data byte 60						Data byte 61								Data byte 62		Data byte 63	

= Unimplemented or reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
Serial#62e5-4821-4a36-b251-390d3ab6240e

EDL — Extended Data Length

This bit distinguishes between CAN format and CAN FD format frames. The EDL bit must not be set for message buffers configured to RANSWER with code field 1010b (see table below).

BRS — Bit Rate Switch

This bit defines whether the bit rate is switched inside a CAN FD format frame.

ESI — Error State Indicator

This bit indicates if the transmitting node is error active or error passive.

CODE — Message Buffer Code

This 4-bit field can be accessed (read or write) by the CPU and by the FlexCAN module itself, as part of the message buffer matching and arbitration process. The encoding is shown in Table 349 and Table 350. See Functional description for additional information.

Table 349. Message buffer code for Rx buffers

CODE description	Rx code BEFORE receive new frame	SRV[1]	Rx code AFTER successful reception[2]	RRS[3]	Comment
0000b: INACTIVE — MB is not active.	INACTIVE	—	—	—	MB does not participate in the matching process.
0100b: EMPTY — MB is active and empty.	EMPTY	—	FULL	—	When a frame is received successfully (after the Move-in process), the CODE field is automatically updated to FULL.
0010b: FULL — MB is full.	FULL	Yes	FULL	—	The act of reading the C/S word followed by unlocking the MB (SRV) does not make the code return to EMPTY. It remains FULL. If a new frame is moved to the MB after the MB was serviced, the code still remains FULL. See Matching process for matching details related to FULL code.
		No	OVERRUN	—	If the MB is FULL and a new frame

Table continues on the next page...

Provided under NDA only COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-4821-48a2-b251-390d3a6262c1

Table 349. Message buffer code for Rx buffers (continued)

CODE description	Rx code BEFORE receive new frame	SRV[1]	Rx code AFTER successful reception[2]	RRS[3]	Comment
					is moved to this MB before the CPU services it, the CODE field is automatically updated to OVERRUN. See Matching process for details about overrun behavior.
0110b: OVERRUN — MB is being overwritten into a full buffer.	OVERRUN	Yes	FULL	—	If the CODE field indicates OVERRUN and CPU has serviced the MB, when a new frame is moved to the MB then the code returns to FULL.
		No	OVERRUN	—	If the CODE field already indicates OVERRUN, and another new frame must be moved, the MB will be overwritten again, and the code will remain OVERRUN. See Matching process for details about overrun behavior.
1010b: RANSWER[4] — A frame was configured to recognize a Remote Request frame and transmit a Response frame in return.[5]	RANSWER	—	TANSWER(1110b)	0	A Remote Answer was configured to recognize a remote request frame received. After that an MB is set to transmit a response frame. The code is automatically changed to TANSWER (1110b). See Matching process

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table 349. Message buffer code for Rx buffers (continued)

CODE description	Rx code BEFORE receive new frame	SRV[1]	Rx code AFTER successful reception[2]	RRS[3]	Comment
					for details. If CTRL2[RRS] is negated, transmit a response frame whenever a remote request frame with the same ID is received.
		—	—	1	This code is ignored during matching and arbitration process. See Matching process for details.
CODE[0]=1: BUSY — FlexCAN is updating the contents of the MB. The CPU must not access the MB.	BUSY[6]	—	FULL	—	Indicates that the MB is being updated. It will be negated automatically and does not interfere with the next CODE.
		—	OVERRUN	—	

1. SRV: Serviced MB. MB was read and unlocked by reading TIMER or other MB.
2. A frame is considered a successful reception after the frame to be moved to MB (move-in process). See Move-in for details.
3. Remote Request Stored bit, see "Control 2 register (CTRL2)" for details.
4. Code 1010b is not considered Tx and an MB with this code should not be aborted.
5. Code 1010b must be used in message buffers configured in CAN FD format, having the EDL bit set.
6. Note that for Tx MBs, the BUSY bit should be ignored upon read, except when AEN bit is set in the MCR register. If this bit is asserted, the corresponding MB does not participate in the matching process.

Table 350. Message buffer code for Tx buffers

CODE Description	Tx Code BEFORE tx frame	MB RTR	Tx Code AFTER successful transmission	Comment
1000b: INACTIVE — MB is not active	INACTIVE	—	—	MB does not participate in arbitration process.
1001b: ABORT — MB is aborted	ABORT	—	—	MB does not participate in arbitration process.
1100b: DATA — MB is a Tx data frame (MB RTR must be 0)	DATA	0	INACTIVE	Transmit data frame unconditionally once. After transmission,

Table continues on the next page...

Table 350. Message buffer code for Tx buffers (continued)

CODE Description	Tx Code BEFORE tx frame	MB RTR	Tx Code AFTER successful transmission	Comment
				the MB automatically returns to the INACTIVE state.
1100b: REMOTE — MB is a Tx Remote Request frame (MB RTR must be 1)	REMOTE	1	EMPTY	Transmit remote request frame unconditionally once. After transmission, the MB automatically becomes an Rx Empty MB with the same ID.
1110b: TANSWER — MB is a Tx Response frame from an incoming Remote Request frame	TANSWER	—	RANSWER	This is an intermediate code that is automatically written to the MB by the CHI as a result of a match to a remote request frame. The remote response frame will be transmitted unconditionally once, and then the code will automatically return to RANSWER (1010b). The CPU can also write this code with the same effect. The remote response frame can be either a data frame or another remote request frame depending on the RTR bit value. See Matching process and Arbitration process for details.

SRR — Substitute Remote Request

Fixed recessive bit, used only in extended format. It must be set to one by the user for transmission (Tx Buffers) and will be stored with the value received on the CAN bus for Rx receiving buffers. It can be received as either recessive or dominant. If FlexCAN receives this bit as dominant, then it is interpreted as an arbitration loss.

1 = Recessive value is compulsory for transmission in extended format frames

0 = Dominant is not a valid value for transmission in extended format frames

IDE — ID Extended Bit

This field identifies whether the frame format is standard or extended.

1 = Frame format is extended

0 = Frame format is standard

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cac4520c-4821-48a8-8251-390d3a8624c1

RTR — Remote Transmission Request

This bit affects the behavior of remote frames and is part of the reception filter. See Table 349, Table 350, and the description of the RRS field in Control 2 register (CTRL2) for additional details.

If FlexCAN transmits this bit as '1' (recessive) and receives it as '0' (dominant), it is interpreted as an arbitration loss. If this bit is transmitted as '0' (dominant), then if it is received as '1' (recessive), the FlexCAN module treats it as a bit error. If the value received matches the value transmitted, it is considered a successful bit transmission.

1 = Indicates the current MB may have a remote request frame to be transmitted if MB is Tx. If the MB is Rx then incoming remote request frames may be stored.

0 = Indicates the current MB has a data frame to be transmitted. In Rx MB it may be considered in matching processes.

NOTE

When configuring CAN FD frames, the RTR bit must be negated.

DLC — Length of Data in Bytes

This 4-bit field is the length (in bytes) of the Rx or Tx data, which is located in offset 8h–Fh of the MB space (see Table 348). In reception, this field is written by the FlexCAN module, copied from the DLC (Data Length Code) field of the received frame. In transmission, this field is written by the CPU and corresponds to the DLC field value of the frame to be transmitted. When RTR = 1, the frame to be transmitted is a remote frame and does not include the data field, regardless of the DLC field (see Table 352).

TIME STAMP — Free-Running Counter Time Stamp

This 16-bit field is a copy of the Free-Running Timer, captured for Tx and Rx frames at the time when the beginning of the Identifier field appears on the CAN bus.

Table 351. Time stamp operation

TSTAMPCAP	MBTSBASE	TIMER_SOURCE	Captured time base	Capture point
b00	bxx	0	CAN_TIMER incremented by CAN bit clock	Second bit of identifier field
b00	bxx	1	CAN_TIMER incremented by on-chip timer clock	Second bit of identifier field
bxx	b00	0	CAN_TIMER incremented by CAN bit clock	Second bit of identifier field
bxx	b00	1	CAN_TIMER incremented by on-chip timer clock	Second bit of identifier field
b01	b01	x	Lower 16 bits of high-resolution on-chip timer	Seventh bit of the end of frame field for transmission and sixth bit of the end of frame field for reception
b01	b10	x	Upper 16 bits of high-resolution on-chip timer	Seventh bit of the end of frame field for transmission and sixth bit of the end of frame field for reception

Table continues on the next page...

Table 351. Time stamp operation (continued)

TSTAMPCAP	MBTSBASE	TIMER_SOURCE	Captured time base	Capture point
b10	b01	x	Lower 16 bits of high-resolution on-chip timer	Start of frame
b10	b10	x	Upper 16 bits of high-resolution on-chip timer	Start of frame
b11	b01	x	Lower 16 bits of high-resolution on-chip timer	Start of frame for classical CAN frame format and res bit for CAN FD frame format
b11	b10	x	Upper 16 bits of high-resolution on-chip timer	Start of frame for classical CAN frame format and res bit for CAN FD frame format

PRIO — Local priority

This 3-bit field is used only when MCR[LPRIO_EN] is set, and it only makes sense for Tx mailboxes. These bits are not transmitted. They are appended to the regular ID to define the transmission priority. See Arbitration process.

ID — Frame Identifier

In standard frame format, only the 11 most significant bits (28 to 18) are used for frame identification in both receive and transmit cases. The 18 least significant bits are ignored. In extended frame format, all bits are used for frame identification in both receive and transmit cases.

DATA BYTE 0 to 63 — Data Field

Up to sixty four bytes can be used for a data frame, depending on the size of payload selected for the message buffers.

For Rx frames, the data is stored as it is received from the CAN bus. DATA BYTE (*n*) is valid only if *n* is less than DLC as shown in the table below.

Table 352. DATA BYTEs validity

DLC	Valid DATA BYTEs
0	none
1	DATA BYTE 0
2	DATA BYTE 0 to 1
3	DATA BYTE 0 to 2
4	DATA BYTE 0 to 3
5	DATA BYTE 0 to 4
6	DATA BYTE 0 to 5
7	DATA BYTE 0 to 6
8	DATA BYTE 0 to 7
9	DATA BYTE 0 to 11
10	DATA BYTE 0 to 15

Table continues on the next page...

Table 352. DATA BYTEs validity (continued)

DLC	Valid DATA BYTEs
11	DATA BYTE 0 to 19
12	DATA BYTE 0 to 23
13	DATA BYTE 0 to 31
14	DATA BYTE 0 to 47
15	DATA BYTE 0 to 63

50.4.4 FlexCAN memory partition for CAN FD

When CAN FD is enabled, the FlexCAN RAM can be partitioned in blocks of 512 bytes. Each block can accommodate a number of message buffers which depends on the configuration provided by FDCTRL[MBDSRn] bit fields as shown in table below.

Table 353. RAM partition

RAM block	Number of MBs with 8 bytes (default range)	Size control bit field in FDCTRL register	Number of MBs of different sizes, per block
0	0 to 31	MBDSR0	MBDSR0=00, 32 MBs with 8 bytes payload MBDSR0=01, 21 MBs with 16 bytes payload MBDSR0=10, 12 MBs with 32 bytes payload MBDSR0=11, 7 MBs with 64 bytes payload
1	32 to 63	MBDSR1	MBDSR1=00, 32 MBs with 8 bytes payload MBDSR1=01, 21 MBs with 16 bytes payload MBDSR1=10, 12 MBs with 32 bytes payload MBDSR1=11, 7 MBs with 64 bytes payload
2	64 to 95	MBDSR2	MBDSR2=00, 32 MBs with 8 bytes payload MBDSR2=01, 21 MBs with 16 bytes payload MBDSR2=10, 12 MBs with 32 bytes payload MBDSR2=11, 7 MBs with 64 bytes payload
3	96 to 127	MBDSR3	MBDSR3=00, 32 MBs with 8 bytes payload

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CSU Automotive Technology Co., Ltd.
c1c462e5-4821-48a8-b251-390d3ab2404d

Table 353. RAM partition (continued)

RAM block	Number of MBs with 8 bytes (default range)	Size control bit field in FDCTRL register	Number of MBs of different sizes, per block
			MBDSR3=01, 21 MBs with 16 bytes payload MBDSR3=10, 12 MBs with 32 bytes payload MBDSR3=11, 7 MBs with 64 bytes payload
4	—	—	7 MBs with 64 bytes (total number of MBs in blocks 0, 1, 2, 3 ≤ 121)
5	—	—	7 MBs with 64 bytes (total number of MBs in blocks 0, 1, 2, 3 ≤ 114)
6	—	—	7 MBs with 64 bytes (total number of MBs in blocks 0, 1, 2, 3 ≤ 107)
7	—	—	7 MBs with 64 bytes (total number of MBs in blocks 0, 1, 2, 3 ≤ 100)

When payload sizes of 16, 32, or 64 bytes are configured in some or all RAM blocks 0, 1, 2, and 3, the total number of MBs and its respective number order may differ from the default configuration of 8 bytes. For example, suppose Block0 is configured to 8 bytes payload, Block1 to 16 bytes, Block2 to 32 bytes and Block3 to 64 bytes, suppose also MCR[MAXMB] is configured to 95. Then the following table indicates how the message buffers will be arranged in RAM.

Table 354. RAM partition example

RAM block	Payload size	Number of MBs in the RAM block	Message buffer range
0	FDCTRL[MBDSR0]=00, 8 bytes payload	32	0 to 31
1	FDCTRL[MBDSR1]=01, 16 bytes payload	21	32 to 52
2	FDCTRL[MBDSR2]=10, 32 bytes payload	12	53 to 64
3	FDCTRL[MBDSR3]=11, 64 bytes payload	7	65 to 71
4	64 bytes payload	7	72 to 78
5	64 bytes payload	7	79 to 85
6	64 bytes payload	7	85 to 92
7	64 bytes payload	3	93 to 95

50.4.5 FlexCAN message buffer memory map

The FlexCAN memory buffers are allocated in memory according to the tables below.

Table 355. 8-byte message buffers

Address offset (hex)	MBDSR=b00 8-byte payload
0080	MB0
0090	MB1
00A0	MB2
00B0	MB3
00C0	MB4
00D0	MB5
00E0	MB6
00F0	MB7
0100	MB8
0110	MB9
0120	MB10
0130	MB11
0140	MB12
0150	MB13
0160	MB14
0170	MB15
0180	MB16
0190	MB17
01A0	MB18
01B0	MB19
01C0	MB20
01D0	MB21
01E0	MB22
01F0	MB23
0200	MB24
0210	MB25
0220	MB26
0230	MB27
0240	MB28

Table continues on the next page...

Table 355. 8-byte message buffers (continued)

Address offset (hex)	MBDSR=b00 8-byte payload
0250	MB29
0260	MB30
0270	MB31
0280	MB32
0290	MB33
02A0	MB34
02B0	MB35
02C0	MB36
02D0	MB37
02E0	MB38
02F0	MB39
0300	MB40
0310	MB41
0320	MB42
0330	MB43
0340	MB44
0350	MB45
0360	MB46
0370	MB47
0380	MB48
0390	MB49
03A0	MB50
03B0	MB51
03C0	MB52
03D0	MB53
03E0	MB54
03F0	MB55
0400	MB56
0410	MB57
0420	MB58
0430	MB59

Table continues on the next page...

Table 355. 8-byte message buffers (continued)

Address offset (hex)	MBDSR=b00 8-byte payload
0440	MB60
0450	MB61
0460	MB62
0470	MB63
0480	MB64
0490	MB65
04A0	MB66
04B0	MB67
04C0	MB68
04D0	MB69
04E0	MB70
04F0	MB71
0500	MB72
0510	MB73
0520	MB74
0530	MB75
0540	MB76
0550	MB77
0560	MB78
0570	MB79
0580	MB80
0590	MB81
05A0	MB82
05B0	MB83
05C0	MB84
05D0	MB85
05E0	MB86
05F0	MB87
0600	MB88
0610	MB89
0620	MB90

Table continues on the next page...

Table 355. 8-byte message buffers (continued)

Address offset (hex)	MBDSR=b00 8-byte payload
0630	MB91
0640	MB92
0650	MB93
0660	MB94
0670	MB95
0680	MB96
0690	MB97
06A0	MB98
06B0	MB99
06C0	MB100
06D0	MB101
06E0	MB102
06F0	MB103
0700	MB104
0710	MB105
0720	MB106
0730	MB107
0740	MB108
0750	MB109
0760	MB110
0770	MB111
0780	MB112
0790	MB113
07A0	MB114
07B0	MB115
07C0	MB116
07D0	MB117
07E0	MB118
07F0	MB119
0800	MB120
0810	MB121

Table continues on the next page...

Table 355. 8-byte message buffers (continued)

Address offset (hex)	MBDSR=b00 8-byte payload
0820	MB122
0830	MB123
0840	MB124
0850	MB125
0860	MB126
0870	MB127

Table 356. 16-byte message buffers

Address offset (hex)	MBDSR=b01 16-byte payload
0080	MB0
0098	MB1
00B0	MB2
00C8	MB3
00E0	MB4
00F8	MB5
0110	MB6
0128	MB7
0140	MB8
0158	MB9
0170	MB10
0188	MB11
01A0	MB12
01B8	MB13
01D0	MB14
01E8	MB15
0200	MB16
0218	MB17
0230	MB18
0248	MB19
0260	MB20

Table continues on the next page...

Table 356. 16-byte message buffers (continued)

Address offset (hex)	MBDSR=b01 16-byte payload
0280	MB21
0298	MB22
02B0	MB23
02C8	MB24
02E0	MB25
02F8	MB26
0310	MB27
0328	MB28
0340	MB29
0358	MB30
0370	MB31
0388	MB32
03A0	MB33
03B8	MB34
03D0	MB35
03E8	MB36
0400	MB37
0418	MB38
0430	MB39
0448	MB40
0460	MB41
0480	MB42
0498	MB43
04B0	MB44
04C8	MB45
04E0	MB46
04F8	MB47
0510	MB48
0528	MB49
0540	MB50
0558	MB51

Table continues on the next page...

Table 356. 16-byte message buffers (continued)

Address offset (hex)	MBDSR=b01 16-byte payload
0570	MB52
0588	MB53
05A0	MB54
05B8	MB55
05D0	MB56
05E8	MB57
0600	MB58
0618	MB59
0630	MB60
0648	MB61
0660	MB62
0680	MB63
0698	MB64
06B0	MB65
06C8	MB66
06E0	MB67
06F8	MB68
0710	MB69
0728	MB70
0740	MB71
0758	MB72
0770	MB73
0788	MB74
07A0	MB75
07B8	MB76
07D0	MB77
07E8	MB78
0800	MB79
0818	MB80
0830	MB81
0848	MB82

Table continues on the next page...

Table 356. 16-byte message buffers (continued)

Address offset (hex)	MBDSR=b01 16-byte payload
0860	MB83

Table 357. 32-byte message buffers

Address offset (hex)	MBDSR=b10 32-byte payload
0080	MB0
00A8	MB1
00D0	MB2
00F8	MB3
0120	MB4
0148	MB5
0170	MB6
0198	MB7
01C0	MB8
01E8	MB9
0210	MB10
0238	MB11
0280	MB12
02A8	MB13
02D0	MB14
02F8	MB15
0320	MB16
0348	MB17
0370	MB18
0398	MB19
03C0	MB20
03E8	MB21
0410	MB22
0438	MB23
0480	MB24
04A8	MB25

Table continues on the next page...

Table 357. 32-byte message buffers (continued)

Address offset (hex)	MBDSR=b10 32-byte payload
04D0	MB26
04F8	MB27
0520	MB28
0548	MB29
0570	MB30
0598	MB31
05C0	MB32
05E8	MB33
0610	MB34
0638	MB35
0680	MB36
06A8	MB37
06D0	MB38
06F8	MB39
0720	MB40
0748	MB41
0770	MB42
0798	MB43
07C0	MB44
07E8	MB45
0810	MB46
0838	MB47

Table 358. 64-byte message buffers

Address offset (hex)	MBDSR=b11 64-byte payload
0080	MB0
00C8	MB1
0110	MB2
0158	MB3
01A0	MB4

Table continues on the next page...

Table 358. 64-byte message buffers (continued)

Address offset (hex)	MBDSR=b11 64-byte payload
01E8	MB5
0230	MB6
0280	MB7
02C8	MB8
0310	MB9
0358	MB10
03A0	MB11
03E8	MB12
0430	MB13
0480	MB14
04C8	MB15
0510	MB16
0558	MB17
05A0	MB18
05E8	MB19
0630	MB20
0680	MB21
06C8	MB22
0710	MB23
0758	MB24
07A0	MB25
07E8	MB26
0830	MB27

Table 359. 64-Byte message buffers in RAM blocks 4, 5, 6, and 7

Address offset (hex)	64-byte payload
1000	MB28
1048	MB29
1090	MB30
10D8	MB31
1120	MB32

Table continues on the next page...

Table 359. 64-Byte message buffers in RAM blocks 4, 5, 6, and 7 (continued)

Address offset (hex)	64-byte payload
1168	MB33
11B0	MB34
1200	MB35
1248	MB36
1290	MB37
12D8	MB38
1320	MB39
1368	MB40
13B0	MB41
1400	MB42
1448	MB43
1490	MB44
14D8	MB45
1520	MB46
1568	MB47
15B0	MB48
1600	MB49
1648	MB50
1690	MB51
16D8	MB52
1720	MB53
1768	MB54
17B0	MB55

50.4.6 Legacy Rx FIFO structure

When MCR[RFEN] is set, the memory area 80h–DCh (which is normally occupied by MBs 0–5) is used by the reception Legacy FIFO engine.

The region 80h–8Ch contains the output of the Legacy Rx FIFO which must be read by the CPU as a message buffer. This output contains the oldest message that has been received but not yet read. The region 90h–DCh is reserved for internal use of the Legacy Rx FIFO engine.

An additional memory area, which starts at E0h and may extend up to 2DCh (normally occupied by MBs 6–37) depending on the CTRL2[RFFN] field setting, contains the ID filter table (configurable from 8 to 128 table elements) that specifies filtering criteria for accepting frames into the Legacy Rx FIFO.

Out of reset, the ID filter table flexible memory area defaults to E0h and extends only to FCh, which corresponds to MBs 6 to 7 for RFFN = 0, for backward compatibility with previous versions of FlexCAN.

The following shows the Legacy Rx FIFO data structure.

Table 360. Legacy Rx FIFO structure

	31 28	24 23	22	21	20	19 18 17 16	15 8	7 0
80h	IDHIT		SRR	IDE	RTR	DLC	TIME STAMP	
84h		ID standard				ID extended		
88h	Data byte 0		Data byte 1				Data byte 2	Data byte 3
8Ch	Data byte 4		Data byte 5				Data byte 6	Data byte 7
90h–DCh	Reserved							
E0h	ID filter table element 0							
E4h	ID filter table element 1							
E8h–2D4h	ID filter table elements 2 to 125							
2D8h	ID filter table element 126							
2DCh	ID filter table element 127							
		= Unimplemented or reserved						

Each ID filter table element occupies an entire 32-bit word and can be compounded by one, two, or four Identifier Acceptance Filters (IDAF) depending on the MCR[IDAM] field setting. The following figures show the IDAF indexation.

The following table shows the three different formats of the ID table elements. Note that all elements of the table must have the same format. See Legacy Rx FIFO for more information.

Table 361. ID table structure

Format	31	30	29 24	23 16	15	14	13 8	7 1	0
A	RTR	IDE	RXIDA (standard = 29–19, extended = 29–1)						
B	RTR	IDE	RXIDB_0 (standard = 29–19, extended = 29–16)		RTR	IDE	RXIDB_1 (standard = 13–3, extended = 13–0)		
C	RXIDC_0 (std/ext = 31–24)		RXIDC_1 (std/ext = 23–16)		RXIDC_2 (std/ext = 15–8)		RXIDC_3 (std/ext = 7–0)		
	= Unimplemented or Reserved								

RTR — Remote frame

This bit specifies if remote frames are accepted into the Legacy FIFO if they match the target ID.

1 = Remote frames can be accepted and data frames are rejected

0 = Remote frames are rejected and data frames can be accepted

IDE — Extended frame

Specifies whether extended or standard frames are accepted into the Legacy FIFO if they match the target ID.

1 = Extended frames can be accepted and standard frames are rejected

0 = Extended frames are rejected and standard frames can be accepted

RXIDA — Rx frame identifier (Format A)

Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In the standard frame format, only the 11 most significant bits (29 to 19) are used for frame identification. In the extended frame format, all bits are used.

RXIDB_0, RXIDB_1 — Rx frame identifier (Format B)

Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In the standard frame format, the 11 most significant bits (a full standard ID) (29 to 19 and 13 to 3) are used for frame identification. In the extended frame format, all 14 bits of the field are compared to the 14 most significant bits of the received ID.

RXIDC_0, RXIDC_1, RXIDC_2, RXIDC_3 — Rx frame identifier (Format C)

Specifies an ID to be used as acceptance criteria for the Legacy FIFO. In both standard and extended frame formats, all 8 bits of the field are compared to the 8 most significant bits of the received ID.

IDHIT — Identifier acceptance filter hit indicator

This 9-bit field indicates which identifier acceptance filter was hit by the received message that is in the output of the Legacy Rx FIFO. See Legacy Rx FIFO for more information.

50.4.7 Enhanced Rx FIFO structure

When ERFCR[ERFEN] is set, the Enhanced Rx FIFO is enabled. The region 2000h–204Ch contains the output of the Enhanced Rx FIFO which must be read by the CPU as a message buffer. This output contains the oldest message that has been received but not yet read. The following table shows the Enhanced Rx FIFO data structure.

Table 362. Enhanced RX FIFO Structure

	31	30	29	28 ... 24	23	22	21	20	19	18	17	16	15 ... 8	7	6 ... 0
2000h	EDL	BRS	ESI	RESERVED		SRR	IDE	RTR	DLC				TIME STAMP LEGACY		
2004h	RESERVED			ID (STANDARD/EXTENDED)									ID (EXTENDED)		
2008h	Data byte 0					Data byte 1							Data byte 2		Data byte 3
200Ch	Data byte 4					Data byte 5							Data byte 6		Data byte 7
2010h	Data byte 8					Data byte 9							Data byte 10		Data byte 11
2014h	Data byte 12					Data byte 13							Data byte 14		Data byte 15
2018h	Data byte 16					Data byte 17							Data byte 18		Data byte 19
201Ch	Data byte 20					Data byte 21							Data byte 22		Data byte 23
2020h	Data byte 24					Data byte 25							Data byte 26		Data byte 27
2024h	Data byte 28					Data byte 29							Data byte 30		Data byte 31
2028h	Data byte 32					Data byte 33							Data byte 34		Data byte 35
202Ch	Data byte 36					Data byte 37							Data byte 38		Data byte 39
2030h	Data byte 40					Data byte 41							Data byte 42		Data byte 43

Table continues on the next page...

Table 362. Enhanced RX FIFO Structure (continued)

2034h	Data byte 44	Data byte 45	Data byte 46	Data byte 47
2038h	Data byte 48	Data byte 49	Data byte 50	Data byte 51
203Ch	Data byte 52	Data byte 53	Data byte 54	Data byte 55
2040h	Data byte 56	Data byte 57	Data byte 58	Data byte 59
2044h	Data byte 60	Data byte 61	Data byte 62	Data byte 63
IH_OFF	RESERVED			ID HIT
TS_OFF	HR TIME STAMP			
2050h	31 Enhanced FIFO Elements (RESERVED)			
...				
29FCh				

NOTE

ID HIT offset and high resolution time stamp offset are changed dynamically according to data length code (DLC) as shown in Table 363.

Table 363. ID HIT offset and high resolution time stamp offset

Data Length Code (DLC)	ID HIT offset (IH_OFF)	High resolution time stamp offset (TS_OFF)
0	2008h	200Ch
1–4	200Ch	2010h
5–8	2010h	2014h
9	2014h	2018h
10	2018h	201Ch
11	201Ch	2020h
12	2020h	2024h
13	2028h	202Ch
14	2038h	203Ch
15	2048h	204Ch

EDL — Extended Data Length

This bit distinguishes between classical CAN format and CAN FD format frames.

0 – Classical CAN frame format

1 – CAN FD frame format

BRS — Bit Rate Switch

This bit defines whether the bit rate is switched inside a CAN FD format frame.

0 – Bit rate is not switched in a CAN FD frame

1 – Bit rate is switched in a CAN FD frame

ESI — Error State Indicator

This bit indicates if the transmitting node is error active or error passive. This bit is meaningful only if EDL = 1.

0 – Transmitting node is error active

1 – Transmitting node is error passive

SRR — Substitute Remote Request

Fixed recessive bit, used only in extended format. Transmitting nodes always send it as recessive and receiving nodes can receive it as either recessive or dominant. If FlexCAN receives this bit as dominant, then it is interpreted as an arbitration loss.

IDE — ID Extended Bit

This field identifies whether the frame format is standard or extended.

0 – Frame format is standard

1 – Frame format is extended

RTR — Remote Frame

0 = Indicates the current frame is a data frame

1 = Indicates the current frame is remote request

DLC — Data Length Code

This 4-bit field defines the number of bytes in the data field of a CAN frame (Data byte 0 to Data byte 63). When RTR = 1, the frame is a remote request and does not include the data field, regardless of the DLC field. Refer to Table 352 for more details.

LEGACY TIME STAMP — 16-bit Time Stamp

This 16-bit field is a copy of the Free-Running Timer, captured during the CAN frame. See Table 351 to find more details about legacy time stamp operation.

ID — Frame Identifier

In base frame format, only the 11 most significant bits are used for frame identification. The 18 least significant bits are ignored. In extended frame format, all bits are used for frame identification.

DATA BYTE 0 to 63 — Data Field

Up to sixty four bytes can be stored in the data field.

IDHIT — Identifier Acceptance Filter Hit Indicator

This field indicates which Enhanced Rx FIFO Filter Element (ERFFELn) was hit by the received message that is in the output of the Enhanced Rx FIFO. For each filter region, standard-ID filter space and extended-ID filter space, there is an independent index starting from zero. Table 364 shows how IDHIT is written by FlexCAN according to each filter element.

Table 364. IDHIT for Enhanced Rx FIFO

Enhanced Rx FIFO filter element — ERFFEL	ID HIT value	Filter element type
ERFFEL0	0	Extended-ID
ERFFEL1	1	Extended-ID
. . .	. . .	Extended-ID
ERFFEL(m–1)	m–1	Extended-ID
ERFFEL(m)	0	Standard-ID

Table continues on the next page...

Table 364. IDHIT for Enhanced Rx FIFO (continued)

Enhanced Rx FIFO filter element — ERFFEL	ID HIT value	Filter element type
ERFFEL(m+1)	1	Standard-ID
. . .	. . .	Standard-ID
ERFFEL(2n–m+1)	2x(n–m)+1	Standard-ID

NOTE

Where m = NEXIF and n = NFE. If NEXIF = 0, only standard-ID filter elements exist, and if NEXIF > NFE, then only extended-ID filter elements exist.

HR TIME STAMP — High-resolution Time Stamp

32-bit time base captured during the CAN frame. When CTRL2[TSTAMPCAP] is different from zero, a 32-bit time base is captured from a dedicated on-chip timer which operates in free running mode. See TSTAMPCAP field description in Control 2 register (CTRL2) to find more details about capture point configuration of the high-resolution time stamp.

50.5 Functional description

The FlexCAN module is a CAN protocol engine with a very flexible mailbox system for transmitting and receiving CAN frames. The mailbox system is composed of a set of message buffers (MB) that store configuration and control data, time stamp, message ID, and data (see Message buffer structure).

For Classical CAN frames, simultaneous reception through Legacy FIFO and mailbox is supported. For CAN FD frames, reception is supported through mailboxes and Enhanced Rx FIFO. For mailbox reception, a matching algorithm makes it possible to store received frames only into MBs that have the same ID programmed in the ID field. A masking scheme makes it possible to match the ID programmed on the MB with a range of IDs on received CAN frames. For transmission, an arbitration algorithm decides the prioritization of MBs to be transmitted based on the message ID (optionally augmented by 3 local priority bits) or the MB ordering.

Before proceeding with the functional description, an important concept must be explained. A message buffer is said to be active at a given time if it can participate in both the matching and arbitration processes. An Rx MB with a 0b code is inactive (see Table 349). Similarly, a Tx MB with a 1000b or 1001b code is also inactive (see Table 350).

The FlexCAN module is also able to receive and transmit messages in CAN FD format. The message buffers are sized to adequately store the quantity of data bytes selected by the FDCTRL[MBDSRn] fields. The quantity of FD MBs available for a given quantity of data bytes is described in the FDCTRL register. See also FlexCAN memory partition for CAN FD.

50.5.1 Transmit process

To transmit a CAN frame, the CPU must prepare a message buffer for transmission by executing the following procedure:

1. Check whether the respective interrupt bit is set and clear it.
2. If the MB is active (transmission pending), write the ABORT code (1001b) to the CODE field of the Control and Status word to request an abort of the transmission.
3. Wait for the corresponding IFLAG bit to be asserted by polling the IFLAG register, or by the interrupt request if enabled by the respective IMASK bit.
4. Read back the CODE field to check if the transmission was aborted or transmitted (see Transmission abort mechanism).
5. Clear the corresponding interrupt flag.
6. Write the ID register (containing the local priority if enabled via MCR[LPRIO_EN]).

7. Write payload data bytes.
8. Configure the Control and Status word with the desired configuration.
    a. Set ID type via MB_CS[IDE].
    b. Set Remote Transmission Request (if needed) via MB_CS[RTR].
    c. If MCR[FDEN] is enabled, also configure the MB_CS[EDL] and MB_CS[BRS] fields. For details about the relationship between the written value and transmitted value of the MB_CS[ESI] field, see Table 374.[4]
    d. Set Data Length Code in bytes via MB_CS[DLC]. See Table 352 for detailed information.
    e. Activate the message buffer to transmit the CAN frame by setting MB_CS[CODE] to Ch.

NOTE

It is strongly recommended that all the fields in MB_CS word be configured in only one 32-bit write operation to maximize software performance. If the fields are configured in separate writes, the MB_CS[CODE] must be the last write in the C/S word.

When the MB is activated, it participates in the arbitration process and is eventually transmitted according to its priority. When the DLC value stored in the MB selected for transmission is larger than the respective MB payload size, FlexCAN adds the necessary number of bytes with constant CCh pattern to complete the expected DLC.

At the end of the successful transmission:

- The value of the free running timer is written into the Time Stamp field.
- The CODE field in the Control and Status word is updated.
- Both CRC and FDCRC registers are updated.
- A status flag is set in the Interrupt Flag register.
- An interrupt is generated if allowed by the corresponding Interrupt Mask Register bit.

The new CODE field after transmission depends on the code that was used to activate the MB (see Table 349 and Table 350 in Message buffer structure).

When the Abort feature is enabled (MCR[AEN] is asserted), after the Interrupt flag is asserted for a mailbox configured as transmit buffer, the mailbox is blocked. Therefore the CPU is not able to update it until the Interrupt Flag is negated by the CPU. This means that the CPU must clear the corresponding IFLAG bit before starting to prepare this MB for a new transmission or reception.

NOTE

If backwards compatibility is desired (MCR[AEN] is negated), write the INACTIVE code (1000b) to the CODE field to inactivate the MB. However, in this case the pending frame may be transmitted without notification (see Mailbox inactivation).

50.5.2 Arbitration process

The arbitration process scans the mailboxes, searching for the Tx mailbox that holds the message to be sent in the next opportunity. This mailbox is called the arbitration winner.

The scan starts from the lowest number mailbox and runs toward the higher ones.

The arbitration process is triggered in the following events:

- From the CRC field of the CAN frame. The start point depends on the value of CTRL2[TASD].
- During the Error Delimiter field of a CAN frame.
- During the Overload Delimiter field of a CAN frame.

[4] The ESI field does not need to be written, and will automatically be transmitted dominant by error active nodes and recessive by error passive nodes. Note that there is an exception if the FlexCAN is operating as a network gateway: In that case, the CPU writes the MB_CS[ESI] bit according to the error status of the node which sent the message.

- When the winner is inactivated and the CAN bus has still not reached the first bit of the Intermission field.
- When there is a CPU write to the C/S word of a winner MB and the CAN bus has still not reached the first bit of the Intermission field.
- When CHI is in Idle state and the CPU writes to the C/S word of any MB.
- When FlexCAN exits Bus Off state.
- Upon leaving Freeze mode or Low Power mode.

If the arbitration process does not manage to evaluate all mailboxes before the CAN bus has reached the first bit of the Intermission field, the temporary arbitration winner is invalidated and the FlexCAN will not compete for the CAN bus in the next opportunity.

The arbitration process selects the winner among the active Tx mailboxes at the end of the scan according to the settings of both CTRL1[LBUF] and MCR[LPRIOEN].

50.5.2.1 Lowest-number mailbox first

If CTRL1[LBUF] is asserted, the first (lowest number) active Tx mailbox found is the arbitration winner. MCR[LPRIOEN] has no effect when CTRL1[LBUF] is asserted.

50.5.2.2 Highest-priority mailbox first

If CTRL1[LBUF] is negated, then the arbitration process searches the active Tx mailbox with the highest priority, which means that this mailbox's frame would have a higher probability to win the arbitration on CAN bus when multiple external nodes compete for the bus at the same time.

The sequence of bits considered for this arbitration is called the *arbitration value* of the mailbox. The highest-priority Tx mailbox is the one that has the lowest arbitration value among all Tx mailboxes.

If two or more mailboxes have equivalent arbitration values, the mailbox with the lowest number is the arbitration winner.

The composition of the arbitration value depends on the MCR[LPRIOEN] setting.

50.5.2.2.1 Local priority disabled

If MCR[LPRIOEN] is negated the arbitration value is built in the exact sequence of bits as they would be transmitted in a CAN frame (see the following table) in such a way that the local priority is disabled.

Table 365. Composition of the arbitration value when local priority is disabled

Format	Mailbox arbitration value (32 bits)				
Standard (IDE = 0)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	- (18 bits)	- (1 bit)
Extended (IDE = 1)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

50.5.2.2.2 Local priority enabled

To enable local priority, MCR[LPRIOEN] must be asserted. In this case the mailbox PRIO field is included at the very left of the arbitration value (see the following table).

Table 366. Composition of the arbitration value when local priority is enabled

Format	Mailbox arbitration value (35 bits)					
Standard (IDE = 0)	PRIO (3 bits)	Standard ID (11 bits)	RTR (1 bit)	IDE (1 bit)	- (18 bits)	- (1 bit)

Table continues on the next page...

Table 366. Composition of the arbitration value when local priority is enabled (continued)

Format	Mailbox arbitration value (35 bits)					
Extended (IDE = 1)	PRIO (3 bits)	Extended ID[28:18] (11 bits)	SRR (1 bit)	IDE (1 bit)	Extended ID[17:0] (18 bits)	RTR (1 bit)

Because the PRIO field is the most significant part of the arbitration value, mailboxes with low PRIO values have higher priority than mailboxes with high PRIO values regardless of the rest of their arbitration values.

Note that the PRIO field is not part of the frame on the CAN bus. Its purpose is only to affect the internal arbitration process.

50.5.2.3 Arbitration process (continued)

After the arbitration winner is found, its content is copied to a hidden auxiliary MB called Tx Serial Message Buffer (Tx SMB), which has the same structure as a normal MB but is not user accessible. This operation is called move-out and after it is done, write access to the C/S word of the corresponding MB is blocked (if MCR[AEN] is asserted). Write access is restored in the following events:

- After the MB is transmitted and the corresponding IFLAG bit is cleared by the CPU.
- FlexCAN enters Freeze mode or Bus Off.
- FlexCAN loses the bus arbitration or there is an error during the transmission.

At the first opportunity window on the CAN bus, the message on the Tx SMB is transmitted according to the CAN protocol rules.

Arbitration process can be triggered in the following situations:

- During Rx and Tx frames from CAN CRC field to end of frame. CTRL2[TASD] value may be changed to optimize the arbitration start point.
- During CAN Bus Off state from TX_ERR_CNT=124 to 128. CTRL2[TASD] value may be changed to optimize the arbitration start point.
- During C/S write by CPU in Bus Idle. First C/S write starts arbitration process, and a second C/S write during this same arbitration restarts the process. If other C/S writes are performed, Tx arbitration process is pending. If there is no arbitration winner after the arbitration process has finished, then the TX arbitration machine begins a new arbitration process. If there is a pending arbitration and Bus Idle state starts, then an arbitration process is triggered. In this case the first and second C/S write in Bus Idle will not restart the arbitration process. It is possible that there is not enough time to finish arbitration in Wait For Bus Idle state and the next state is Idle. In this case the scan is not interrupted, and it is completed during Bus Idle state. During this arbitration C/S write does not cause arbitration restart.
- Arbitration winner deactivation during a valid arbitration window.
- Upon exiting Freeze mode (first bit of the Wait For Bus Idle state). If there is a re-synchronization during Wait For Bus Idle, the arbitration process is restarted.

Arbitration process stops in the following situations:

- All mailboxes were scanned.
- A Tx active mailbox is found if lowest buffer feature is enabled.
- Arbitration winner inactivation or abort during any arbitration process.
- There was not enough time to finish Tx arbitration process (for instance, when a deactivation was performed near the end of frame). In this case arbitration process is pending.
- Error or overload flag in the bus.
- Low Power or Freeze mode request in Idle state.

Arbitration is considered pending as described below:

- It was not possible to finish arbitration process in time.
- C/S write during arbitration if write is performed in a MB whose number is lower than the Tx arbitration pointer.
- Any C/S write if there is no Tx arbitration process in progress.
- Rx Match has just updated a Rx code to Tx code.
- Entering Bus Off state.

C/S write during arbitration has the following effect:

- If C/S write is performed in the arbitration winner, a new process is restarted immediately.
- If C/S write is performed in a MB whose number is higher than the Tx arbitration pointer, the ongoing arbitration process will scan this MB as normal.

50.5.3 Receive process

To be able to receive CAN frames into a mailbox, the CPU must prepare it for reception by executing the following steps:

1. If the mailbox is active (either Tx or Rx) inactivate the mailbox (see Mailbox inactivation), preferably with a safe inactivation (see Transmission abort mechanism).
2. Write the ID word into the mailbox.
3. Write the EMPTY code (0100b) to the CODE field of the Control and Status word to activate the mailbox. No setup is required for EDL, BRS, and ESI bits—they are overwritten by the respective bit fields in the received message.

After the MB is activated, it will be able to receive frames that match the programmed filter. At the end of a successful reception, the mailbox is updated by the move-in process (see Move-in) as follows:

1. The received data field (8 bytes at most for Classical CAN message format and up to 64 bytes for CAN FD message format) is stored.
2. The received Identifier field is stored.
3. The value of the Free Running Timer at the time of the second bit of frame's Identifier field is written into the mailbox's Time Stamp field.
4. The received SRR, IDE, RTR, EDL, BRS, ESI, and DLC fields are stored.
5. The CODE field in the Control and Status word is updated (see Table 349 and Table 350 in Section Message buffer structure).
6. A status flag is set in the Interrupt Flag Register and an interrupt is generated if allowed by the corresponding Interrupt Mask Register bit.

The recommended way for the CPU to service (read) the frame received in a mailbox is by the following procedure:

1. Read the Control and Status word of that mailbox.
2. Check if the BUSY bit is deasserted, indicating that the mailbox is locked. Repeat step 1) while it is asserted. See Mailbox lock mechanism.
3. Read the contents of the mailbox. After the mailbox is locked, its contents won't be modified by FlexCAN move-in processes. See Move-in.
4. Acknowledge the proper flag at IFLAG registers.
5. Read the free running timer to unlock the mailbox.

The CPU should poll for frame reception by the status flag bit for the specific mailbox in one of the IFLAG registers and not by the CODE field of that mailbox. Polling the CODE field does not work because after a frame is received and the CPU services the mailbox (by reading the C/S word followed by unlocking the mailbox), the CODE field will not return to EMPTY. It will remain FULL, as explained in Table 349. If the CPU tries to work around this behavior by writing to the C/S word to force an EMPTY code after reading the mailbox without a prior safe inactivation, a newly received frame matching the filter of that mailbox may be lost.

CAUTION

In summary: never do polling by reading directly the C/S word of the mailboxes. Instead, read the IFLAG registers.

Note that the received frame's Identifier field is always stored in the matching mailbox, thus the contents of the ID field in a mailbox may change if the match was due to masking. When MCR[SRXDIS] is asserted, FlexCAN will not store frames transmitted by itself in any MB, even if it contains a matching Rx mailbox, and no interrupt flag or interrupt signal will be generated. Otherwise, when MCR[SRXDIS] is deasserted, FlexCAN can receive frames transmitted by itself if a matching Rx mailbox exists.

When MCR[DMA] is asserted, upon receiving a frame in Legacy FIFO, IFLAG1[BUF5I] generates a DMA request and does not generate a CPU interrupt (see Legacy Rx FIFO under DMA operation). The IMASK1 bits in Legacy Rx FIFO region are not used.

The DMA controller must service the received frame using the following procedure:

1. Read the Control and Status word (read 80h address, optional).
2. Read the ID field (read 84h address, optional).
3. Read all data bytes (start read at 88h address, optional).

50.5.4 Matching process

The matching process scans the MB memory looking for Rx MBs programmed with the same ID as the one received from the CAN bus. . The matching starts from the lowest number message buffer toward the higher ones.

For enhanced Rx FIFO see Enhanced Rx FIFO.

For legacy Rx FIFO see Legacy Rx FIFO.

As the frame is being received, it is stored in a hidden auxiliary MB called Rx Serial Message Buffer (Rx SMB).

The matching process start point depends on the following conditions:

- If the received frame is a remote frame, the start point is the CRC field of the frame.
- If the received frame is a data frame with DLC field equal to zero, the start point is the CRC field of the frame.
- If the received frame is a data frame with DLC field different than zero, the start point is the DATA field of the frame.

If a matching ID is found in one of the mailboxes, the contents of the Rx SMB are transferred to to the matched mailbox by the move-in process. If any CAN protocol error is detected then no match results are transferred to the matched mailbox at the end of reception.

The matching process scans all matching elements of the active Rx mailboxes (CODE is EMPTY, FULL, OVERRUN, or RANSWER) in search of a successful comparison with the matching elements of the Rx SMB that is receiving the frame on the CAN bus. The Rx SMB has the same structure as a mailbox. The reception structures (mailboxes) associated with the matching elements that had a successful comparison are the *matched structures*. The *matching winner* is selected at the end of the scan among those matched structures and depends on conditions described ahead. See the following table.

Table 367. Matching architecture

Structure	SMB[RTR]	CTRL2[RRS]	CTRL2[EACEN]	MB[IDE]	MB[RTR]	MB[ID[1]]	MB[CODE]
Mailbox	0	—	0	cmp[2]	no_cmp[3]	cmp_msk[4]	EMPTY or FULL or OVERRUN
Mailbox	0	—	1	cmp_msk	cmp_msk	cmp_msk	EMPTY or FULL or OVERRUN
Mailbox	1	0	—	cmp	no_cmp	cmp	RANSWER

Table continues on the next page...

Table 367. Matching architecture (continued)

Structure	SMB[RTR]	CTRL2[RRS]	CTRL2[EACEN]	MB[IDE]	MB[RTR]	MB[ID[1]]	MB[CODE]
Mailbox	1	1	0	cmp	no_cmp	cmp_msk	EMPTY or FULL or OVERRUN
Mailbox	1	1	1	cmp_msk	cmp_msk	cmp_msk	EMPTY or FULL or OVERRUN

1. For mailbox structure, If SMB[IDE] is asserted, the ID is 29 bits (ID Standard + ID Extended). If SMB[IDE] is negated, the ID is only 11 bits (ID Standard). For Legacy Rx FIFO structure, the ID depends on IDAM.
2. cmp: Compares the Rx SMB contents with the MB contents regardless the masks.
3. no_cmp: The Rx SMB contents are not compared with the MB contents.
4. cmp_msk: Compares the Rx SMB contents with MB contents taking into account the masks.

NOTE

For Enhanced Rx FIFO, see Enhanced Rx FIFO matching process.

A reception structure is free-to-receive when any of the following conditions is satisfied:

- The CODE field of the mailbox is EMPTY.
- The CODE field of the mailbox is either FULL or OVERRUN and it has already been serviced (the C/S word was read by the CPU and unlocked as described in Mailbox lock mechanism).
- The CODE field of the mailbox is either FULL or OVERRUN and an inactivation (see Mailbox inactivation) is performed.

The scan order for mailboxes is from the matching element with lowest number to the higher ones.

The matching winner search for mailboxes is affected by MCR[IRMQ]. If it is negated, the matching winner is the first matched mailbox regardless if it is free-to-receive or not. If it is asserted, the matching winner is selected according to the priority below:

1. the first free-to-receive matched mailbox;
2. the last non free-to-receive matched mailbox.

If a non-safe mailbox inactivation (see Mailbox inactivation) occurs during matching process and the mailbox inactivated is the temporary matching winner, then the temporary matching winner is invalidated. The matching elements scan is not stopped nor restarted—it continues normally. The consequence is that the current matching process works as if the matching elements compared before the inactivation did not exist, therefore a message may be lost.

Suppose, for example, that IRMQ is enabled, there are two MBs with the same ID, and FlexCAN starts receiving messages with that ID. Let us say that these MBs are the second and the fifth in the array. When the first message arrives, the matching algorithm finds the first match in MB number 2. The code of this MB is EMPTY, so the message is stored there. When the second message arrives, the matching algorithm finds MB number 2 again, but it is not "free-to-receive", so it keeps looking, finds MB number 5 and stores the message there. If yet another message with the same ID arrives, the matching algorithm finds out that there are no matching MBs that are "free-to-receive", so it decides to overwrite the last matched MB, which is number 5. In doing so, it sets the CODE field of the MB to indicate OVERRUN.

The ability to match the same ID in more than one MB can be exploited to implement a reception queue to allow more time for the CPU to service the MBs. By programming more than one MB with the same ID, received messages are queued into the MBs. The CPU can examine the Time Stamp field of the MBs to determine the order in which the messages arrived.

Matching to a range of IDs is possible by using ID acceptance masks. FlexCAN supports individual masking per MB (see the description of Rx Individual Mask registers (RXIMR0 - RXIMR127)). During the matching algorithm, if a mask bit is asserted, then the corresponding ID bit is compared. If the mask bit is negated, the corresponding ID bit is a "don't care". Note that the Individual Mask Registers are implemented in RAM, so they are not initialized out of reset. Also, they can only be programmed when the module is in Freeze mode; otherwise, they are blocked by hardware.

FlexCAN also supports an alternate masking scheme with only four mask registers (RXMGMASK, RX14MASK, and RX15MASK) for backwards compatibility with legacy applications. This alternate masking scheme is enabled when the IRMQ bit in the MCR register is negated.

50.5.5 Move process

There are two types of move process: move-in and move-out.

50.5.5.1 Move-in

The move-in process is the copy of a message received by an Rx SMB to an Rx mailbox that has matched it. Each Rx SMB has its own move-in process, but only one is performed at a given time as described ahead. The move-in starts only when the message held by the Rx SMB has a corresponding matching winner (see Matching process) and all of the following conditions are true:

- The CAN bus has either reached or already gone past:
  - The second bit of Intermission field next to the frame that carried the message that is in the Rx SMB.
  - The first bit of an overload frame next to the frame that carried the message that is in the Rx SMB.
- There is no ongoing matching process.
- The destination mailbox is not locked by the CPU.
- There is no ongoing move-in process from another Rx SMB. If more than one move-in process is to be started at the same time, both are performed and the newest substitutes for the oldest.

The term pending move-in is used throughout the documentation and stands for a move-to-be that still does not satisfy all of the aforementioned conditions.

The move-in is cancelled and the Rx SMB is able to receive another message if any of the following conditions is satisfied:

- The destination mailbox is inactivated after the CAN bus has reached the first bit of Intermission field next to the frame that carried the message and its matching process has finished.
- There is a previous pending move-in to the same destination mailbox.
- The Rx SMB is receiving a frame transmitted by the FlexCAN itself and self-reception is disabled (MCR[SRXDIS] is asserted).
- Any CAN protocol error is detected.

Note that the pending move-in is not cancelled if the module enters Freeze or Low-Power mode. It stays on hold, only waiting for Freeze and Low-Power mode to be exited and the module to be unlocked. If an MB is unlocked during Freeze mode, the move-in happens immediately.

The move-in process is the execution by the FlexCAN of the following steps:

1. Read all data words from the Rx SMB in accordance with the selected payload size for the Rx storage element.
2. Write all data words to the Rx mailbox in accordance with the selected payload size for the Rx storage element. If the data size of the storage element is smaller than the original payload size described in the message's DLC field, the payload is truncated and the high order bytes that do not fit the destination size are lost.
3. Read the Control/Status and ID words from the Rx SMB.
4. Write Control/Status and ID words to the Rx mailbox, and update the CODE field.

The move-in process is not atomic, in such a way that it is immediately cancelled by the inactivation of the destination mailbox (see Mailbox inactivation). In this case the mailbox may remain partially updated, thus incoherent.

The BUSY Bit (least significant bit of the CODE field) of the destination message buffer is asserted while the move-in is being performed to alert the CPU that the message buffer content is temporarily incoherent.

50.5.5.2 Move-out

The move-out process is the copy of the content from a Tx Mailbox to the Tx SMB when a message for transmission is available (see Arbitration process). The move-out occurs in the following conditions:

- The first bit of Intermission field
- During Bus Off state when TX Error Counter is in the 124 to 128 range
- During Bus Idle state
- During Wait For Bus Idle state

The move-out process is not atomic. Only the CPU has priority to access the memory concurrently out of Bus Idle state. In Bus Idle, the move-out has the lowest priority to the concurrent memory accesses.

50.5.6 Data coherence

In order to maintain data coherency and FlexCAN proper operation, the CPU must obey the rules described in Transmit process and Receive process.

50.5.6.1 Transmission abort mechanism

The abort mechanism provides a safe way to request the abort of a pending transmission. A feedback mechanism is provided to inform the CPU if the transmission was aborted or if the frame could not be aborted and was transmitted instead.

Two primary conditions must be fulfilled in order to abort a transmission:

- MCR[AEN] must be asserted.
- The first CPU action must be the writing of abort code (1001b) into the CODE field of the Control and Status word.

Active MBs configured for transmission must be aborted first before they can be updated. If the abort code is written to a Mailbox that is currently being transmitted or to a Mailbox that was already loaded into the Tx SMB for transmission, the write operation is blocked and the transmission is not disturbed. However, the abort request is captured and kept pending until one of the following conditions is satisfied:

- The module loses the bus arbitration.
- There is an error during the transmission.
- The module is put into Freeze mode.
- The module enters Bus Off state.
- There is an overload frame.

If none of the conditions above are reached:

- The MB is transmitted correctly.
- The interrupt flag is set in the IFLAG register.
- An interrupt to the CPU is generated (if enabled).

The abort request is automatically cleared when the interrupt flag is set. On the other hand, if only one of the above conditions is reached, the frame is not transmitted; therefore:

- The abort code is written into the CODE field.
- The interrupt flag is set in the IFLAG.
- An interrupt is (optionally) generated to the CPU.

If the CPU writes the abort code before the transmission begins internally, then the write operation is not blocked; therefore, the MB is updated and the interrupt flag is set. In this way the CPU only needs to read the abort code to make sure the active MB was safely inactivated. Although the AEN bit is asserted and the CPU wrote the abort code, in this case the MB is inactivated and not aborted, because the transmission did not start yet. One Mailbox is aborted only when the abort request is captured and kept pending until one of the previous conditions is satisfied.

50.5.6.2 Mailbox inactivation

Inactivation is a mechanism provided to protect the mailbox against updates by the FlexCAN internal processes, thus allowing the CPU to rely on mailbox data coherence after having updated it, even in Normal mode.

Inactivation of transmission mailboxes must be performed just when MCR[AEN] is deasserted.

If a mailbox is inactivated, it participates in neither the arbitration process nor the matching process until it is reactivated. See Transmit process and Receive process for more detailed instructions on how to inactivate and reactivate a mailbox.

To inactivate a mailbox, the CPU must update its CODE field to INACTIVE (either 0b or 1000b).

Because you will not be able to synchronize the CODE field update with the FlexCAN internal processes, an inactivation can have the following consequences:

- A frame in the bus that matches the filtering of the inactivated Rx mailbox may be lost without notice, even if there are other mailboxes with the same filter.
- A frame containing the message within the inactivated Tx mailbox may be transmitted without setting the respective IFLAG.

In order to perform a safe inactivation and avoid the above consequences for Tx mailboxes, the CPU must use the transmission abort mechanism (see Transmission abort mechanism).

The inactivation automatically unlocks the mailbox (see Mailbox lock mechanism).

50.5.6.3 Mailbox lock mechanism

Other than mailbox inactivation, FlexCAN has another data coherence mechanism for the receive process. When the CPU reads the Control and Status word of an Rx MB with codes FULL or OVERRUN, FlexCAN assumes that the CPU wants to read the whole MB in an atomic operation, and therefore it sets an internal lock flag for that MB. The lock is released when the CPU reads the free running timer (global unlock operation), or when it reads the Control and Status word of another MB regardless of its code. A CPU write into the C/S word also unlocks the MB, but this procedure is not recommended for normal unlock use because it cancels a pending move and potentially may lose a received message. The MB locking prevents a new frame from being written into the MB while the CPU is reading it.

NOTE

The locking mechanism applies only to Rx MBs that have a code different than INACTIVE (0b) or EMPTY[1] (0100b). Also, Tx MBs cannot be locked.

Suppose, for example, that the second and the fifth MBs of the array are programmed with the same ID, and FlexCAN has already received and stored messages into these two MBs. Suppose now that the CPU decides to read MB number 5 and at the same time another message with the same ID is arriving. When the CPU reads the Control and Status word of MB number 5, this MB is locked. The new message arrives and the matching algorithm finds out that there are no free-to-receive MBs, so it decides to override MB number 5. However, this MB is locked, so the new message cannot be written there. It will remain in the Rx SMB waiting for the MB to be unlocked, and only then will be written to the MB.

If the MB is not unlocked in time and yet another new message with the same ID arrives, then the new message overwrites the one on the Rx SMB and there will be no indication of lost messages either in the CODE field of the MB or in the Error and Status Register.

While the message is being moved in from the Rx SMB to the MB, the BUSY bit on the CODE field is asserted. If the CPU reads the Control and Status word and finds out that the BUSY bit is set, it should defer accessing the MB until the BUSY bit is negated.

NOTE

If the BUSY bit is asserted or if the MB is empty, then reading the Control and Status word does not lock the MB.

Inactivation takes precedence over locking. If the CPU inactivates a locked Rx MB, then its lock status is negated and the MB is marked as invalid for the current matching round. Any pending message on the Rx SMB will not be transferred anymore to the MB.

[1] In previous FlexCAN versions, reading the C/S word locked the MB even if it was EMPTY. This behavior is maintained when the IRMQ bit is negated.

An MB is unlocked when the CPU reads the Free Running Timer Register (see Free Running Timer (TIMER)), or the C/S word of another MB.

Lock and unlock mechanisms have the same functionality in both Normal and Freeze modes.

An unlock during Normal or Freeze mode results in the move-in of the pending message. However, the move-in is postponed if an unlock occurs during a low power mode (see Modes of operation), and it takes place only when the module returns to Normal or Freeze modes.

50.5.7 Enhanced Rx FIFO

FlexCAN supports an enhanced Rx FIFO engine which can store up to 32 CAN FD messages. The region 2000h–204Fh contains the output of the FIFO which should be read by the CPU. The enhanced Rx FIFO is enabled by setting EFRCR[ERFEN]. Although FlexCAN has two FIFO options, Legacy Rx FIFO and Enhanced Rx FIFO, they cannot be enabled at the same time. Refer to Legacy Rx FIFO to find detailed information about Legacy Rx FIFO.

The enhanced Rx FIFO has a watermark which is configured by setting EFRCR[ERFWM]. If EFRCR[ERFWM] is set, the CPU is notified only if a minimum number of messages is stored in the FIFO. When the number of stored messages is greater than the value in EFRCR[ERFWM], ERFSR[ERFWMI] is set by the hardware. Optionally, an interrupt or a DMA transfer can be triggered if MCR[DMA] or ERFIER[ERFWMIIE] are set, respectively.

For the Enhanced Rx FIFO to receive, the CPU must execute the configuration procedure below. The same procedure must be done if the CPU needs to change any of the configuration of the Enhanced RX FIFO.

1. Enter Freeze mode.
2. Enable Enhanced Rx FIFO by setting ERFCR[ERFEN], if it is not already enabled.[5]
3. Write one to ERFSR[ERFCLR] to reset Enhanced Rx FIFO engine.
4. Clear EFRSR[ERFUFW], EFRSR[ERFOVF], EFRSR[ERFWMI], and EFRSR[ERFDA], if they are set.
5. Write EFRCR[NFE] to configure the total number of enhanced Rx FIFO filter elements to be used in Enhanced Rx FIFO reception.
6. Write ERFCR[NEXIF] to configure the number of extended ID and standard ID filter elements to be used in Enhanced Rx FIFO reception.[6]
7. Configure the Enhanced Rx FIFO watermark by writing ERFCR[ERFWM].
8. If interrupts will be used, set the interrupt enables in the ERIER register.
9. If DMA will be used, set MCR[DMA] to enable DMA operation and write ERFCR[DMALW] to configure the number of words to transfer for each Enhanced Rx FIFO data element.
10. Configure the filter elements by writing in the ERFFEL*n* registers.[7]
11. Leave Freeze mode.

There are two types of Enhanced Rx FIFO filter elements that can be stored in ERFFEL*n* registers: extended-ID filter element and standard-ID filter element. Each extended-ID filter element is stored in two ERFFEL*n* registers while each standard-ID filter element is stored in one ERFFEL*n* register. The total number of Enhanced RX FIFO filter elements is defined by ERFCR[NFE].

In addition, the filter memory space can be split into two regions: one for extended-ID filter elements and another for standard-ID filter elements, according to ERFCR[NEXIF]. The figure below shows how the Enhanced Rx filter elements are defined. Detailed information about the Enhanced Rx FIFO matching process and filter element formats can be found in Enhanced Rx FIFO matching process.

[5] MCR[RFEN] must be cleared so that Enhanced Rx FIFO is enabled.
[6] NEXIF ≤ NFE + 1.
[7] ERFFEL*n* registers are implemented in RAM; thus they must be explicitly initialized prior to any reception.

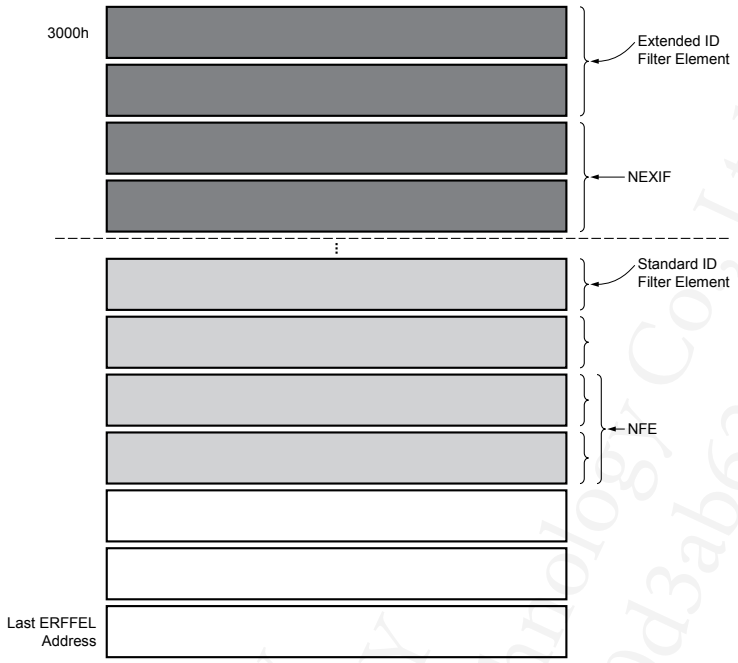


Figure 433. Number of enhanced Rx FIFO filter elemets

50.5.7.1 Enhanced Rx FIFO matching process

When ERFCR[ERFEN] is enabled, FlexCAN scans the ERFFEL*n* memory region. If at least one filter element satisfies the matching criteria, then the CAN message content is transfered to the enhanced RX FIFO memory.

NOTE

If multiple filters match the incoming message ID, then the first matching filter found by the matching process must be indicated in IDHIT.

Each ERFFEL*n* register can store one standard filter element. The matching criteria are defined by ERFFEL*n*[FSCH] in this way:

- If FSCH = b00, the filter scheme is based on mask and filter. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base frame format (IDE = 0)
  2. (ID[*n*] = STD ID filter [*n*]) or (STD ID Mask[*n*] =0) for each bit *n* from 0 to 10
  3. (RTR = RTR Filter) or (RTR MASK = 0)

In this explanation, RTR and ID are the Remote Transmit Request field and the ID from a CAN message, respectively.

If FSCH = b00, the filters and masks are defined as shown in this table.

Table 368. Standard ID filter element with filter + mask scheme (FSCH = b00)

31	30	29	28	27	26										16	15			12	11	10										0
FSCH=b00		Reserved		RTR filter	STD ID filter											Reserved				RTR mask	STD ID mask										

- If FSCH = b01, the filter scheme is based on range. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base frame format (IDE = 0)

2. ID ≥ STD ID Filter1
3. ID ≤ STD ID Filter2
4. (RTR = RTR filter) or (RTR MASK = 0)

RTR and ID are the Remote Transmit Request bit and ID from a CAN message, respectively. If FSCH = b01, the filters and mask are defined as shown in this table.

Table 369. Standard ID filter element with range scheme (FSCH = b01)

31	30	29	28	27	26	16	15	12	11	10	0
FSCH=b01		Reserved		RTRfilter	STD ID Filter2		Reserved		RTR mask	STD ID Filter1	

- If FSCH = b10, the filter scheme is based on two filters without masks. A CAN message matches a standard ID filter element only if these criteria are reached:
  1. CAN message is base frame format (IDE = 0)
  2. (ID[*n*] = STD ID Filter1[*n*]) or (ID[*n*] = STD ID Filter2[*n*]) for each bit *n* from 0 to 10
  3. (RTR = RTR Filter1) or (RTR = RTR Filter2)

RTR and ID are the Remote Transmit Request bit and ID from a CAN message, respectively. If FSCH = b10, the filters are defined as shown in this table.

Table 370. Standard ID filter element with 2-filter scheme (FSCH = b10)

31	30	29	28	27	26	16	15	12	11	10	0
FSCH=b10		Reserved		RTR Filter 2	STD ID Filter2		Reserved		RTR Filter 1	STD ID Filter1	

Each pair of ERFFEL*n* registers can store one extended filter element. The matching criteria are defined by ERFFEL*n*[FSCH] in this way:

- If FSCH = b00, the filter scheme is based on mask and filter. A CAN message matches an extended ID filter element only if these criteria are reached:
  1. CAN message is extended frame format (IDE = 1)
  2. (ID[*n*] = EXT ID filter [*n*]) or (EXT ID Mask[*n*] =0) for each bit *n* from 0 to 28
  3. (RTR = RTR Filter) or (RTR MASK = 0)

If FSCH = b00, the filters and masks are defined as shown in this table.

Table 371. Extended ID filter element with filter + mask scheme (FSCH = b00)

31	30	29	28	0
FSCH		RTR filter	EXT ID filter	
Reserved		RTR mask	EXT ID mask	

- If FSCH = b01, the filter scheme is based on range. A CAN message matches an extended ID filter element only if the following criteria are reached:
  1. CAN message is extended frame format (IDE = 1)
  2. ID ≥ EXT ID Filter1

3. ID ≤ EXT ID Filter2
4. (RTR = RTR Filter) or (RTR MASK = 0)

If FSCH = b01, the filters and masks are defined as shown in this table.

Table 372. Extended ID filter element with range scheme (FSCH = b01)

31	30	29	28	0
FSCH		RTR filter	EXT ID Filter2	
Reserved		RTR mask	EXT ID filter 1	

- If FSCH = b10, the filter scheme is based on two filters without masks. A CAN message matches an extended ID filter element only if these criteria are reached:
  1. CAN message is extended frame format (IDE = 1)
  2. (ID[*n*] = EXT ID Filter1[*n*]) or (ID[*n*] = EXT ID Filter2[*n*]) for each bit *n* from 0 to 28
  3. (RTR = RTR Filter1) or (RTR = RTR Filter2)

If FSCH = b10, the filters and masks are defined as shown in this table.

Table 373. Extended ID filter element with 2-filter scheme (FSCH = b10)

31	30	29	28	0
FSCH		RTR Filter2	EXT ID Filter2	
Reserved		RTR Filter1	EXT ID filter 1	

50.5.7.2 Enhanced Rx FIFO under DMA operation

The DMA feature is enabled by asserting both ERF[ERFEN] and MCR[DMA]. The DMA controller can read the received message by reading a message buffer structure at the Enhanced FIFO output port at the address range defined in Enhanced Rx FIFO structure.

The CPU should not access the Enhanced FIFO output port address range during DMA operation, so that the FIFO engine operates properly. Before enabling MCR[DMA], the CPU must service Enhanced Rx FIFO status bits. Otherwise, these bits may show that the FIFO has data to be serviced, and mistakenly generate a DMA request. Before disabling MCR[DMA], the CPU must first clear the ERFUFW, ERFOVF, ERMWMI, and ERFDA fields in the ERFSR register, and then clear the Enhanced RX FIFO engine by writing one to ERFSR[ERFCLR].

ERFSR[ERFDA] is set by the hardware when there is at least one frame available to be read from the Enhanced Rx FIFO, and ERFSR[ERFWMI] is set by the hardware when the number of frames stored in the FIFO is greater than the watermark defined by ERFCR[ERFWM]. A DMA request is generated when ERFSR[ERFWMI] is set. Upon receiving the request, the DMA controller can read the message in the Enhanced Rx FIFO output. The DMA controller must perform the number of read loops in the FIFO output defined by ERFCR[ERFWM]. Also, each message reading process must end by the address defined in ERFCR[DMALW].

The following rules must be obeyed for Enhanced Rx FIFO DMA operation:

- Because a DMA transfer cannot be dynamically changed, ERFCR[DMALW] should be programmed so that the Enhanced Rx FIFO element can store the largest CAN message present on the CAN bus.
- Data bytes are valid according to the DLC field. See Table 352.

Each time one message is read from FIFO by the DMA controller, ERFSR[ERFDA] is cleared by FlexCAN and set again if there is at least one message stored in the FIFO. Moreover, ERFSR[ERFWMI] is cleared by the hardware when the messages are read in the FIFO output, and set again if the number of messages in the FIFO is greater than the value defined by ERFCR[ERFWM].

For example, suppose that the maximum number of bytes in the data field of a CAN frame for a certain application is eight, and high-resolution time stamp is enabled. In that case the last Enhanced RX FIFO address offset can be found in Table 362 and Table 363. Using this address offset, ERFCR[DMALW] can be determined like this:

- Maximum number of data bytes = 8
- HR TIME STAMP enabled
- Last address offset = TS_OFF = 2014h
- DMALW = 5

50.5.7.3 Enhanced Rx FIFO clear operation

When ERFCR[ERFEN] is set, the CPU can clear the Enhanced Rx FIFO by writing one into ERFSR[ERFCLR]. The clear operation resets the internal FIFO pointers although the FIFO content stored in RAM is not changed. This operation can only be performed in Freeze mode and is blocked by hardware in other modes. This operation does not clear the ERFUFW, ERFOVF, ERFDA, and ERFWMI fields in EFRSR. Consequently the CPU must service all these fields before executing the clear FIFO task.

50.5.8 Legacy Rx FIFO

The Legacy Rx FIFO is receive-only and is enabled by asserting MCR[RFEN]. The reset value of this bit is zero to maintain software backward compatibility with previous versions of the module that did not have the Legacy FIFO feature.

CAUTION

Legacy Rx FIFO must not be enabled when CAN FD feature is enabled.

The Legacy FIFO is 6-message deep. The memory region occupied by the Legacy FIFO structure (both message buffers and Legacy FIFO engine) is described in Legacy Rx FIFO structure. The CPU can read the received messages sequentially, in the order they were received, by repeatedly reading a message buffer structure at the output of the Legacy FIFO.

IFLAG1[BUF5I] (Frames available in Legacy Rx FIFO) is asserted when there is at least one frame available to be read from the Legacy FIFO. An interrupt is generated if it is enabled by the corresponding mask bit. Upon receiving the interrupt, the CPU can read the message (accessing the output of the Legacy FIFO as a message buffer) and the RXFIR register, and then clear the interrupt. If there are more messages in the Legacy FIFO the act of clearing the interrupt updates the output of the Legacy FIFO with the next message and updates RXFIR with the attributes of that message, reissuing the interrupt to the CPU. Otherwise, the flag remains negated. The output of the Legacy FIFO is valid only while IFLAG1[BUF5I] is asserted.

IFLAG1[BUF6I] (Legacy Rx FIFO Warning) is asserted when the number of unread messages within the Legacy Rx FIFO is increased to five from four due to the reception of a new one, meaning that the Legacy Rx FIFO is almost full. The flag remains asserted until the CPU clears it.

IFLAG1[BUF7I] (Legacy Rx FIFO Overflow) is asserted when an incoming message was lost because the Legacy Rx FIFO is full. Note that the flag will not be asserted when the Legacy Rx FIFO is full and the message was captured by a Mailbox. The flag remains asserted until the CPU clears it.

Clearing one of those three flags does not affect the state of the other two.

An interrupt is generated if an IFLAG bit is asserted and the corresponding mask bit is asserted too.

A powerful filtering scheme is provided to accept only frames intended for the target application, reducing the interrupt servicing workload. The filtering criteria is specified by programming a table of up to 128 32-bit registers, according to CTRL2[RFFN] setting, that can be configured to one of the following formats (see also Legacy Rx FIFO structure):

- Format A: 128 IDAFs (extended or standard IDs including IDE and RTR)
- Format B: 256 IDAFs (standard IDs or extended 14-bit ID slices including IDE and RTR)
- Format C: 512 IDAFs (standard or extended 8-bit ID slices)

NOTE

A chosen format is applied to all entries of the filter table. It is not possible to mix formats within the table.

Every frame available in the Legacy Rx FIFO has a corresponding IDHIT (Identifier Acceptance Filter Hit Indicator) that can be read in the IDHIT field from C/S word, as shown in the Legacy Rx FIFO Structure description. Another way the CPU can obtain this information is by accessing the RXFIR register. RXFIR[IDHIT] refers to the message at the output of the Legacy FIFO and is valid while the IFLAG1[BUF5I] flag is asserted. The RXFIR register must be read only before clearing the flag, which guarantees that the information refers to the correct frame within the Legacy FIFO.

Up to 32 elements of the filter table are individually affected by the Individual Mask Registers (RXIMRx), according to the setting of CTRL2[RFFN], allowing very powerful filtering criteria to be defined. If MCR[IRMQ] is negated, then the Legacy Rx FIFO filter table is affected by RXFGMASK.

NOTE

For more information about the difference between FD and non-FD regarding this feature, see Table 336.

50.5.8.1 Legacy Rx FIFO under DMA operation

The receive-only Legacy FIFO can support DMA. This feature is enabled by asserting both MCR[RFEN] and MCR[DMA]. The reset value of MCR[DMA] is zero to maintain backward compatibility with previous versions of the module that did not have the DMA feature.

The DMA controller can read the received message by reading a message buffer structure at the Legacy FIFO output port at the 80h–8Ch address range.

When MCR[DMA] is asserted the CPU must not access the Legacy FIFO output port address range. Before enabling MCR[DMA], the CPU must service the IFLAGs asserted in the Legacy Rx FIFO region. Otherwise, these IFLAGs may show that the FIFO has data to be serviced, and mistakenly generate a DMA request. Before disabling MCR[DMA], the CPU must perform a clear Legacy FIFO operation.

IFLAG1[BUF5I] (Frames available in Legacy Rx FIFO) is asserted when there is at least one frame available to be read from the FIFO. Consequently a DMA request is generated simultaneously. Upon receiving the request, the DMA controller can read the message (accessing the output of the Legacy FIFO as a message buffer). The DMA reading process must end by reading address 8Ch, which clears IFLAG1[BUF5I] and updates both the FIFO output with the next message (if FIFO is not empty) and the RXFIR register with the attributes of the new message. If there are more messages stored in the FIFO, IFLAG1[BUF5I] will be re-asserted and another DMA request is issued. Otherwise, the flag remains negated.

IFLAG1[BUF6I] and IFLAG1[BUF7I] are not used when the DMA feature is enabled.

When FlexCAN is working with DMA, the CPU does not receive any Legacy Rx FIFO interruption and must not clear the related IFLAGs. In addition, the related IMASKs are not used to mask the generation of DMA requests.

NOTE

For more information about the difference between FD and non-FD regarding this feature, see Table 336.

50.5.8.2 Clear Legacy FIFO operation

When MCR[RFEN] is asserted, the clear Legacy FIFO operation is a feature used to empty Legacy FIFO contents. With MCR[RFEN] asserted the Clear FIFO occurs when the CPU writes one in IFLAG1[BUF0I]. This operation can only be performed in Freeze mode and is blocked by hardware in other modes. This operation does not clear the FIFO IFLAGs; consequently the CPU must service all FIFO IFLAGs before executing the clear FIFO task.

When Legacy Rx FIFO is working with DMA, the clear FIFO operation clears IFLAG1[BUF5I] and the DMA request is canceled.

CAUTION

Clear Legacy FIFO operation does not clear IFLAGs, except when MCR[DMA] is asserted; in this case only IFLAG1[BUF5I] is cleared.

50.5.9 CAN protocol related features

This section describes the CAN protocol related features.

50.5.9.1 CAN FD ISO compliance

The CAN FD protocol has been improved to increase the failure detection capability that was in the original CAN FD protocol, which is also called non-ISO CAN FD, by CAN in Automation (CiA). A three-bit stuff counter and a parity bit have been introduced in the improved CAN FD protocol, now called ISO CAN FD. The CRC calculation has also been modified. All these improvements make the ISO CAN FD protocol incompatible with the non-FD CAN FD protocol. The non-ISO CAN FD is still supported by FlexCAN so that it can be used mainly during an intermediate phase, for evaluation and development purposes.

Therefore, it is strongly recommended to configure FlexCAN to the ISO CAN FD protocol by setting the ISOCANFDEN field in the CTRL2 register.

50.5.9.2 CAN FD frames

The ISO 11898-1 standard specifies the Classical Frame format compliant to ISO 11898-1 (2003) and introduces the CAN Flexible Data Rate Frame format. The Classical Frame format allows bit rates up to 1 Mbit/s and payloads up to 8 bytes per frame. The Flexible Data Rate Frame format allows bit rates higher than 1 Mbit/s and payloads longer than 8 bytes per frame. FlexCAN can receive and transmit CAN FD messages interleaved with Classical CAN messages.

There are three additional control bits in the CAN FD frame. The Extended Data Length (EDL) bit enables a longer data payload with different data length coding. The Bit Rate Switch (BRS) bit decides whether the bit rate is switched inside a CAN FD format frame. The Error State Indicator (ESI) flag is transmitted dominant by error active nodes, and recessive by error passive nodes. There are no Remote Frames (see Remote frames) in the CAN FD format. A message configured to transmit a Remote Frame is always sent out in the Classical CAN format. When an FD frame is received and matches a mailbox, the RTR bit in the receiving message buffer is negated. The RTR bit must be considered in classical frames only.

CAN FD messages may be formatted as long frames where the data field exceeds 8 bytes, and may range from 12 up to 64 bytes. They can also be configured to support bit rate switching, where the control field, the data field, and the CRC field of a CAN frame are transmitted with a higher bit rate than the beginning and the end of the frame. Messages in Classical CAN format are limited to transport a maximum payload of 8 bytes at nominal rate. The following figure illustrates the message formats for Classical and FD frames with either standard or extended ID.

Provided under NXP SLA License
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b7...

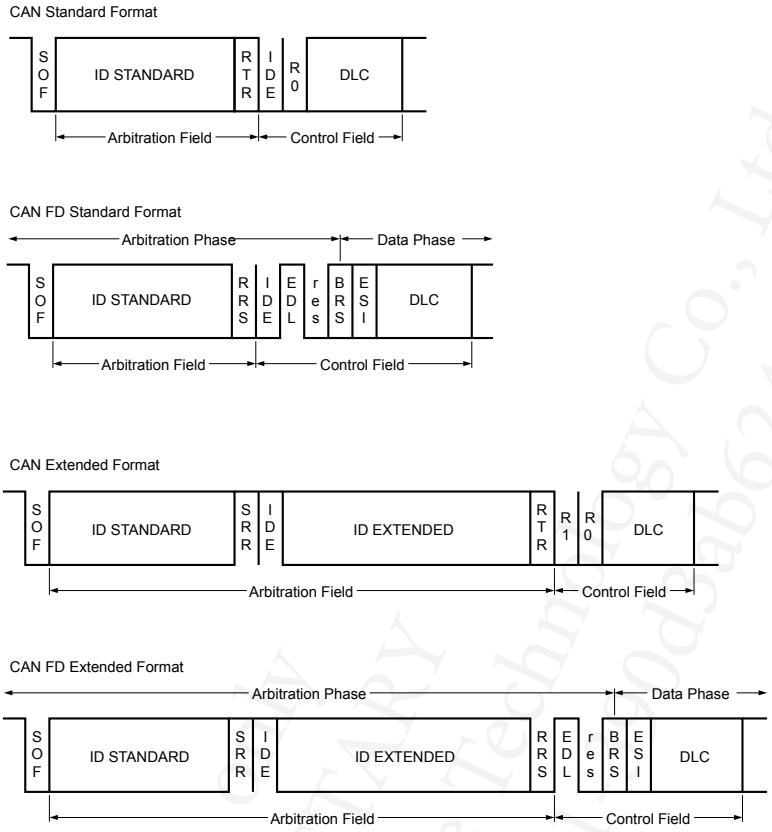


Figure 434. CAN message formats

The ability to receive and transmit CAN FD messages is enabled by MCR[FDEN]. Either a recessive R0 bit in CAN frames with 11-bit identifiers or a recessive R1 bit in CAN frames with 29-bit identifiers are decoded as an EDL bit (not a reserved one). A CAN FD frame is recognized by a recessive EDL bit, and a Classical CAN frame is recognized by a dominant EDL bit. The BRS bit specifies whether this frame switches the bit rate in its data phase. A long frame is decoded in accordance with the DLC field value (see DLC definition in Message buffer structure).

CAN FD messages can be transmitted with two different bit rates. The first part of a CAN FD frame, from the Start Of Frame (SOF) bit until the Bit Rate Switch (BRS) bit, also called the arbitration phase, is transmitted with the nominal bit rate based on a set of nominal CAN bit timing configuration values. The second part, from the BRS bit until the CRC Delimiter bit, also named the data phase, is transmitted with the data bit rate defined by a second set of CAN data bit timing configuration values. Finally, from the CRC Delimiter until the Intermission bits, the transmission returns to nominal bit rate. In CAN FD frames with bit rate switching, the bit timing is changed inside the frame at the sample point of the BRS bit if this bit is recessive. Before the BRS bit, in the CAN FD arbitration phase, the nominal CAN bit timing is used as defined by the CBT register (also by CTRL1 register for backward compatibility). Upon detecting a recessive BRS bit, the CAN data bit timing is used as defined by the FDCBT register.

NOTE

If the length of the time quantum in the nominal bit timing and the length of the time quantum in the data bit timing are not identical, a quantization error of up to one time quantum of the arbitration phase may be present as a phase error. This situation can occur after the switch from arbitration to data phase and will last until the next synchronization event. Thus, the length of the time quantum should be the same in nominal and data bit timing in order to minimize the chance of error frames on the CAN bus, and to optimize the clock tolerance in networks that use FD frames.

FDCTRL[FDRATE] enables the transmission of all frames with bit rate switching if the BRS bit in the selected Tx MB is set. If FDRATE is negated, the transmission is performed at nominal rate regardless of the BRS bit value. FDCTRL[FDRATE] can be written any time but takes effect only for the next message transmitted or received.

The nominal bit timing is resumed at either the sample point of the CRC Delimiter bit or when an error is detected, whichever occurs first. The following figure describes the mechanism for entering and leaving the data phase when BRS bit is recessive.

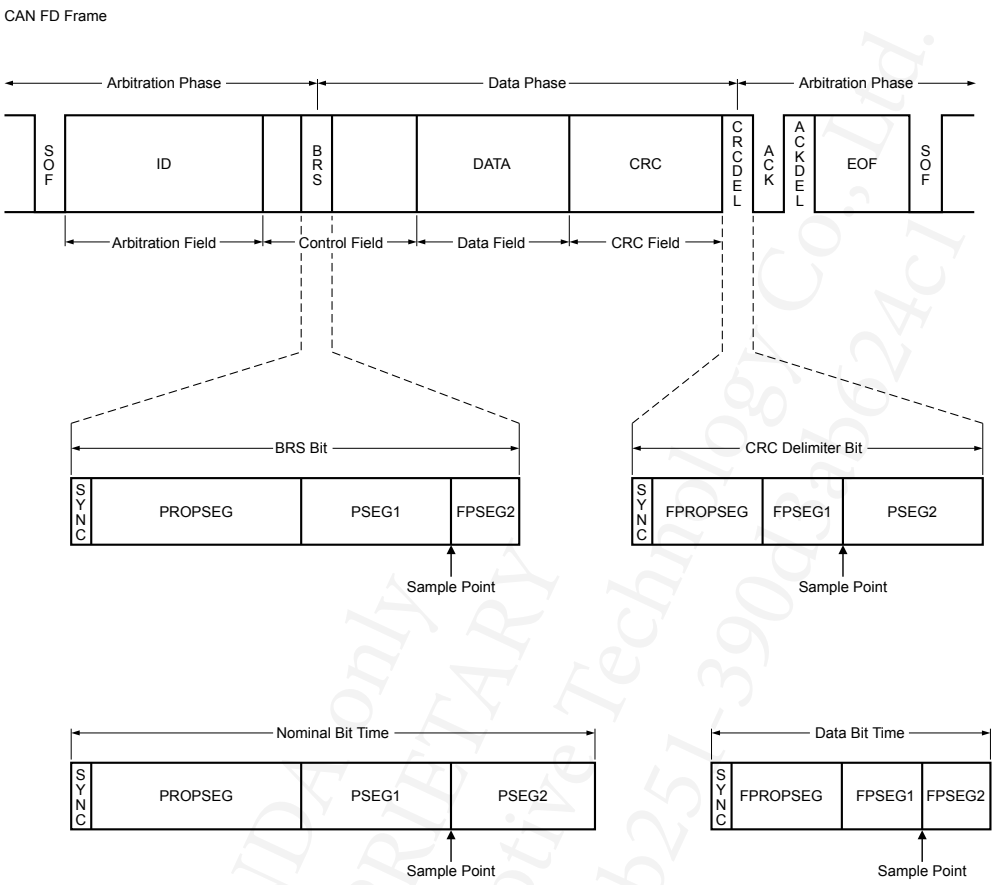


Figure 435. Bit rate switching mechanism for CAN FD messages

NOTE

In Classical CAN frames, the CRC delimiter is one single recessive bit. In CAN FD frames, the CRC delimiter may consist of one or two recessive bits. FlexCAN sends only one recessive bit as the CRC delimiter, but it accepts two recessive bits before the edge from recessive to dominant that starts the acknowledge slot. As a receiver, FlexCAN sends its acknowledge bit after the first CRC delimiter bit. In CAN FD frames, FlexCAN accepts a two-bit dominant ACK slot as a valid ACK to compensate for phase shifts between the receivers.

The maximum configurable bit rate in the CAN FD data phase depends on the clock frequency of CAN_PE subblock. For example, with a CAN_PE clock frequency of 40 MHz and the shortest configurable bit time of 5 time quanta, the bit rate in the data phase is 8 Mbit/s.[8]

The value of the ESI bit is determined either by the transmitter's error state at the start of the transmission, if the frame is originated in the FlexCAN node, or by the original transmitting node in case FlexCAN is acting as a gateway for the message. If the transmitter is error passive, ESI is transmitted recessive; otherwise, it is transmitted dominant. The permutations of the relationship between the written value and the transmitted value of the ESI are shown in this table.

[8] The frequency used in this example may not be supported on this chip; it is shown only to demonstrate how the maximum configurable bit rate is calculated.

Table 374. Written versus transmitted values of ESI field

FlexCAN fault confinement status at start of frame	ESI bit Of Tx MB	Transmitted ESI
Error active	0	0 (Error Active)
Error passive	0	1 (Error Passive)
Error active	1	1 (Error Passive)
Error passive	1	1 (Error Passive)

There are different CRC polynomials for different CAN frame formats. The first polynomial, CRC_15, is used for all frames in Classical CAN format. The second, CRC_17, is used for frames in CAN FD format with a data field up to sixteen bytes long. The third, CRC_21, is used for frames in CAN FD format with a data field longer than sixteen bytes. Each polynomial results in a Hamming distance of 6. At the start of the frame, all three CRC polynomials are calculated concurrently. The CRC sequence to be transmitted is selected by the values of the EDL bit and the DLC bit field. When receiving a message, FlexCAN decodes EDL and DLC to select the adequate CRC polynomial to check for a CRC error.

In CAN FD format frames, stuff bits are included in the bit stream for CRC calculation. In Classical CAN format frames, stuff bits are not included. After the transmission of the last bit relevant to the CRC calculation, the FDCRC register stores the calculated CRC for the transmitted message, with the adequate length in accordance to the type of message, for both CAN FD and non-FD messages. The CRCR register reports a valid CRC for Classical CAN messages only.

In CAN FD format frames, the CAN bit stuffing method is changed for the CRC sequence so that the stuff bits are inserted at fixed positions. When FlexCAN is transmitting a CAN FD frame, a fixed stuff bit is inserted just before the first bit of the CRC sequence, even if the last bits of the preceding field do not fulfill the CAN stuff condition. Additional stuff bits are inserted after each fourth bit of the CRC sequence. The value of any fixed stuff bit is the inverse value of its preceding bit. When FlexCAN is receiving a CAN FD frame, it discards the fixed stuff bits from the bit stream for the CRC check. A stuff error is detected if the fixed stuff bit has the same value as its preceding bit.

FlexCAN detects errors in CAN FD frames the same way as in Classical CAN frames. The error counters RXERRCNT and TXERRCNT in the ECR register accumulate the counts of Rx and Tx errors, respectively, for both FD and non-FD frames indiscriminately. There are two extra error counters (RXERRCNT_FAST and TXERRCNT_FAST) that accumulate Rx and Tx errors occurring in the data phase of CAN FD frames with the BRS bit set only. The rules for updating the error counters are the same for both CAN FD and non-FD frames (see ECR register).

Error Flags BITERR1, BITERR0, ACKERR, CRCERR, FRMERR, and STFERR in the ESR1 register report errors in both CAN FD and non-FD frames. They also generate the ERRINT interrupt if CTRL1[ERRMSK] is asserted. The ESR1 register has additional error flags (BITERR1_FAST, BITERR0_FAST, CRCERR_FAST, FRMERR_FAST, and STFERR_FAST) to individually indicate the occurrence of errors in the data phase of CAN FD frames with the BRS bit set. There is no ACKERR detected in the data phase of a CAN FD frame. Fault confinement status reported in ESR1[FLTCONF] is the same for both CAN FD and Classical CAN frames, and is based on RXERRCNT and TXERRCNT error counters only. Information contained in RXERRCNT_FAST and TXERRCNT_FAST counters may be considered as status to help detect the error nature related to the bit rate value.

When FlexCAN is in the data phase, either transmitting or receiving a CAN FD message, and detects an error, it immediately switches back to the arbitration phase and to the nominal rate to start an error flag.

Resynchronization and hard synchronization occur in CAN FD frames in the same way as in Classical CAN ones. Additionally, a hard synchronization is also performed at the recessive to dominant edge from EDL to R0 in CAN FD format frames. FlexCAN does not resynchronize while transmitting in the CAN FD data phase.

50.5.9.3 Transceiver delay compensation

The CAN FD protocol allows the transmission and reception of data at a higher bit rate than the nominal rate used in the arbitration phase when the message's BRS bit is set. This feature enables the use of rates up to 8 Mbps.

During the data phase of a CAN FD frame, the transmitter detects a bit error if it cannot receive its own latest transmitted bit at the sample point of that bit. When bit rate switching is enabled (BRS bit is asserted), the length of the CAN bit time in the data phase

can become shorter than the transceiver's loop delay, thus impeding the correct comparison between the transmitted bit and the received bit within the current CAN bit time interval.

Note that the TDC process defines a secondary sample point where the transmitted bit is correctly compared with the received bit in order to check for bit errors.

The TDC mechanism can be enabled by FDCTRL[TDCEN] or ETDC[ETDCEN] and is effective only during the data phase of FD frames having the BRS bit set. It has no effect either on non-FD frames, or on FD frames transmitted at normal bit rate. The TDC is active from the sample point of the BRS bit until the sample point of the CRC Delimiter bit, provided the respective message under transmission has the BRS bit set. When it is active, a comparison is done between the real received bit and the delayed transmitted bit, where the delay is calculated based on the measured transceiver loop delay.

NOTE

The actual value of the CRC Delimiter bit is disregarded by transmitters using the transceiver delay compensation mechanism. A global error at the end of the CRC field will cause the receivers to send error frames that the transmitter will detect during Acknowledge or End of Frame.

For every transmitted FD frame having the BRS bit set, the delay measurement is triggered by the transition from the recessive EDL bit to the dominant R0 bit (as shown in the next figure). The loop delay is measured in Protocol Engine (PE) clock periods (CANCLK, see Protocol timing), from the transmitted EDL-R0 edge to the received EDL-R0 edge. The position of the secondary sample point is defined by the measured loop delay time added to an offset value specified in FDCTRL[TDCOFF] or ETDC[ETDCOFF]. FDCTRL[TDCVAL] or ETDC[ETDCVAL] stores the result of this calculation. The TDCVAL and ETDCVAL value saturates at its maximum value of 63 CANCLK and 255 CANCLK when the delay measurement is too long.

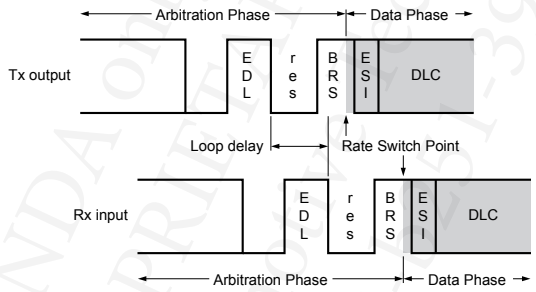


Figure 436. Transceiver loop delay measurement

The measured loop delay is not enough to be used to define the secondary sample point because it relates to the CAN bit edges. The transceiver delay compensation offset TDCOFF or ETDCOFF is used to shift the secondary sample point from the edge to an intermediate point inside the bit time, far away from its edges. Therefore, the TDCOFF or ETDCOFF value cannot be larger than the CAN bit duration in the data phase.

If the secondary sample point is set very near the CAN bit edge (SYNC field), then problems may occur during the bit sampling in the data phase. For the TDC to work reliably, the offset has to use optimal settings. To be sure the bit sampling is performed in the best region, the TDC offset should be configured as shown in this equation:

$$\text{Offset} = (\text{FPSEG1} + \text{FPROPSEG} + 2) \times (\text{FPRESDIV} + 1)$$
or
$$\text{Offset} = (\text{DTSEG1} + 2) \times (\text{EDPRESDIV} + 1), \text{ if ETDCEN}$$

The following figure shows the SSP position when these settings are used.

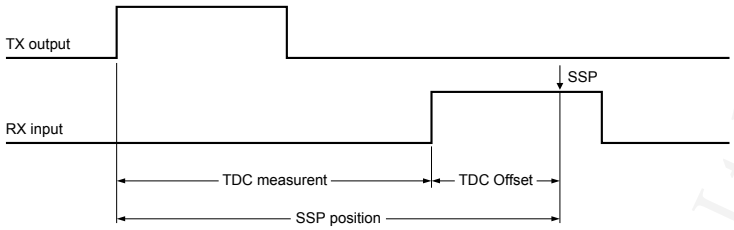


Figure 437. SSP position with optimal values

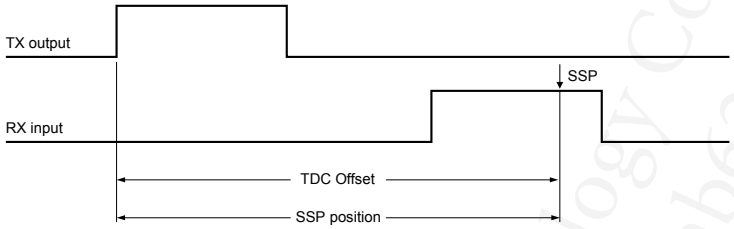


Figure 438. SSP position—TDC measurement disabled by ETDCEN and TDMDIS

Alternatively, if CTRL2[BTE] and ETDC[ETDCEN] are set, the TDMDIS bit can be set to disable the transceiver delay measurement. In this case, the SSP position is defined only by the ETDCOFF field. Figure 438 shows the secondary sample point position when the transceiver delay measurement is disabled.

During the data phase of CAN FD frames with bit rate switching enabled, at the onset of every Tx CAN bit, the transmitted Tx bit value is temporarily stored in a buffer and a time countdown based on FDCTRL[TDCVAL] or ETDC[ETDCVAL] is started which ends with the comparison of the received Rx bit (delayed by the external loop delay plus the specified offset) with the stored Tx bit. If a bit error is detected at the secondary sample point, the FlexCAN issues an error flag to the CAN bus at the next sample point.

During the arbitration phase the delay compensation is always disabled. The maximum delay which can be compensated by the FlexCAN's transceiver delay compensation during the data phase is 3 CAN bit times – 2 Tq. Beyond this limit, the FDCTRL[TDCFAIL] or ETDC[ETDCFAIL] flag is set to indicate when the transceiver delay compensation mechanism is out of range, unable to compensate the transceiver loop delay.

50.5.9.4 Remote frames

A remote frame is a special kind of frame. You can program a mailbox to be a remote request frame by configuring the mailbox as Transmit with the RTR bit set to one. After the remote request frame is transmitted successfully, the mailbox becomes a receive message buffer, with the same ID as before.

When a remote request frame is received by FlexCAN, it can be treated in four ways, depending on remote request storing (CTRL2[RRS]) and Rx FIFO Enable (MCR[RFEN]):

- If RRS is negated the frame's ID is compared to the IDs of the transmit message buffers with the CODE field 1010b. If there is a matching ID, then this mailbox frame will be transmitted. Note that if the matching mailbox has the RTR bit set, then FlexCAN will transmit a remote frame as a response. The received remote request frame is not stored in a receive buffer. It is only used to trigger a transmission of a frame in response. The mask registers are not used in remote frame matching, and all ID bits (except RTR) of the incoming received frame should match. In the case that a remote request frame is received and matches a mailbox, this message buffer immediately enters the internal arbitration process, but is considered as a normal Tx mailbox, with no higher priority. The data length of this frame is independent of the DLC field in the remote frame that initiated its transmission.
- If RRS is asserted the frame's ID is compared to the IDs of the receive mailboxes with the CODE field 0100b, 0010b, or 0110b. If there is a matching ID, then this mailbox will store the remote frame in the same fashion of a data frame. No automatic remote response frame will be generated. The mask registers are used in the matching process.
- If RFEN is asserted FlexCAN will not generate an automatic response for remote request frames that match the Legacy FIFO filtering criteria. If the remote frame matches one of the target IDs, it will be stored in the Legacy FIFO and presented to the

CPU. Note that for filtering formats A and B, it is possible to select whether remote frames are accepted or not. For format C, remote frames are always accepted (if they match the ID). Remote request frames are considered as normal frames, and generate a Legacy FIFO overflow when a successful reception occurs and the Legacy FIFO is already full.

- If ERFEN is set, FlexCAN will not generate an automatic response for remote request frames that match the Enhanced Rx FIFO filtering criteria. Remote Request Frames are considered normal frames, and generate an Enhanced Rx FIFO overflow when a sucessful reception occurs and Enhanced Rx FIFO is already full.

NOTE

There is no remote frame in the CAN FD format. The RTR bit is replaced by a fixed dominant RRS bit. FlexCAN receives and transmits remote frames in the Classical CAN format.

50.5.9.5 Overload frames

FlexCAN does transmit overload frames when the following conditions are detected on the CAN bus:

- Detection of a dominant bit in the first/second bit of Intermission
- Detection of a dominant bit at the 7th bit (last) of End of Frame field (Rx frames)
- Detection of a dominant bit at the 8th bit (last) of Error Frame Delimiter or Overload Frame Delimiter

50.5.9.6 Message buffer time stamp

The value of the free running timer is sampled at the beginning of the Identifier field on the CAN bus, and is stored at the end of move-in in the TIME STAMP field of message buffer, providing network behavior with respect to time.

When CTRL2[TIMER_SRC] is asserted, the free running timer is continuously clocked by an external time tick.

When CTRL2[TIMER_SRC] is negated, the free running timer is clocked by the FlexCAN bit-clock, which defines the baud rate on the CAN bus. During a message transmission/reception, it increments by one for each bit that is received or transmitted. When there is no message on the bus, it counts using the previously programmed baud rate.

The free running timer is not incremented during Disable and Freeze modes. It can be reset upon a specific frame reception, enabling network time synchronization. See the TSYN description in Control 1 register (CTRL1).

Alternatively, by configuring the MTSBASE field of the Control 2 register (CTRL2), the time stamp of the message buffer can capture the lower or higher 16 bits of the high-resolution dedicated counter.

50.5.9.7 High resolution time stamp

The high-resolution time stamp, HR_TIME_STAMP, uses a dedicated timer with a 32-bit counter operating in free running mode. The high-resolution time stamp is enabled by the TSTAMPCAP field of the Control 2 register (CTRL2). When this field is not zero , the dedicated 32-bit counter value is captured during a valid CAN frame and stored in the HR_TIME_STAMP register.

Each HR_TIME_STAMP*n* corresponds to a specific message buffer. For example, HR_TIME_STAMP0 stores the 32-bit time stamp associated with message buffer 0, HR_TIME_STAMP1 stores the 32-bit time stamp associated with message buffer 1, and so on.

The counter value is captured according to the TSTAMPCAP field of the Control 2 register (CTRL2). For classical CAN frames, the capture points can be the start of frame bit or the point in time a CAN frame is considered valid, which is the seventh bit of end of frame for transmission and the sixth bit of end of frame for reception. For CAN FD frames, the capture points can be the start of frame, the point in time a CAN FD frame is considered valid, or the res bit of a CAN FD frame.

The 16-bit time stamp of the message buffer can be configured to capture the lower or higher 16 bits of the high-resolution timer. This configuration is made by CTRL2[MBTSBASE].

50.5.9.8 Protocol timing

The following figure shows the structure of the clock generation circuitry that feeds the CAN Protocol Engine (PE) submodule.

NOTE

Please see the clock distribution chapter (module clocks table) to identify the proper clock source.



Figure 439. CAN engine clocking scheme

The FlexCAN module supports a variety of means to set up bit timing parameters that are required by the CAN protocol. The Control 1 register (CTRL1) has various fields used to control bit timing parameters: PRESDIV, PROPSEG, PSEG1, PSEG2, and RJW.

The CAN Bit Timing register (CBT) extends the range of the CAN bit timing variables in CTRL1. The CAN FD Bit Timing register (FDCBT) provides a second set of CAN bit timing variables to be applied at the data phase of CAN FD frames with the Bit Rate Switch (BRS) set.

The Enhanced Nominal CAN bit timing register (ENCBT) extends the range of CAN bit timing variables in CBT. The Enhanced Data Phase CAN bit timing register (EDCBT) extends the range of CAN bit timing variables in FDCBT. When using ENCBT and EDCBT, the bit timing nominal and data phase serial clock (Sclock) dividers must be programed in the Enhanced CAN Bit Timing Prescalers register (EPRS).

NOTE

When the CAN FD feature is enabled, always set CBT[BTF] or CTRL2[BTE] and configure the CAN bit timing variables in CBT or ENCBT. See CAN Bit Timing register (CBT) or Enhanced Nominal CAN Bit Timing (ENCBT).

The PRESDIV field (as well as its extended range EPRESDIV (or ENPRESDIV) and FDPRESDIV (or EDPRESDIV) for the data phase bits of CAN FD messages) defines the prescaler value (see the equation below) that generates the serial clock (Sclock), whose period defines the time quantum used to compose the CAN waveform. A time quantum (Tq) is the atomic unit of time managed by the CAN engine.

$$Tq = \frac{(PRESDIV + 1)}{f_{CANCLK}}$$

The bit rate, which defines the rate the CAN message is either received or transmitted, is given by the formula:

$$\text{CAN Bit Time} = (\text{Number of Time Quanta in 1 bit time}) * Tq$$

$$\text{Bit Rate} = \frac{1}{\text{CAN Bit Time}}$$

A bit time is subdivided into three segments[1] (see Figure 441, Figure 442 and Table 375):

[1] For further explanation of the underlying concepts, see ISO 11898-1. See also the CAN 2.0A/B protocol specification for bit timing.

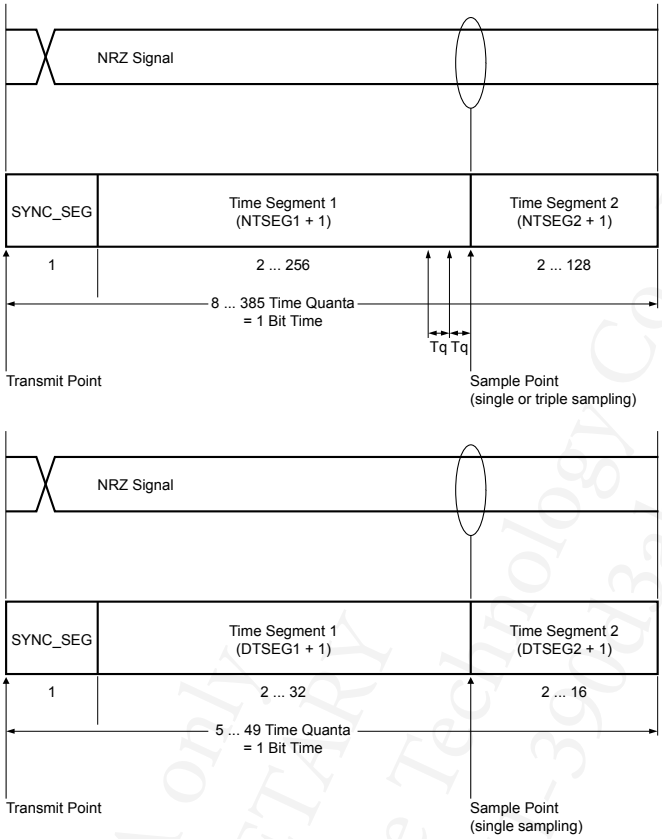


Figure 440. Segments within the bit timing (example using ENCBT and EDCBT bit timing variables)

- SYNC_SEG: This segment has a fixed length of one time quantum. Signal edges are expected to happen within this section.
- Time Segment 1: This segment includes the propagation segment and the phase segment 1 of the CAN standard. It can be programmed by setting CTRL1[PROPSEG] and CTRL1[PSEG1] so that their sum (plus 2) is in the range of 2 to 16 time quanta. When CBT[BTF] is asserted, FlexCAN uses EPROPSEG and EPSEG1 fields from CBT register so that their sum (plus 2) is in the range of 2 to 96 time quanta. For messages in CAN FD format with the BRS bit set, FlexCAN uses FDPROPSEG and FDPSEG1 from FDCBT instead, so that their sum (plus 1) is in the range of 2 to 39 time quanta.

  If CTRL2[BTE] is set, FlexCAN uses ENCBT[NTSEG1] to configure time segment 1 in the range of 2 to 256 time quanta. For the data phase in CAN FD messages with BRS set, EDCBT[DTSEG1] must be used for configuring time segment 1 in the range of 2 to 32 time quanta.
- Time Segment 2: This segment represents the phase segment 2 of the CAN standard. It can be programmed by setting CTRL1[PSEG2] (plus 1) to be 2 to 8 time quanta long. When CBT[BTF] is asserted, FlexCAN uses EPSEG2 fields of CBT register so that its value (plus 1) is in the range of 2 to 32 time quanta. For messages in CAN FD format with the BRS bit set, FlexCAN uses FDPSEG2 from FDCBT instead, so that its value (plus 1) is in the range of 2 to 8 time quanta. The time segment 2 cannot be smaller than the Information Processing Time (IPT), which value is 2 time quanta in FlexCAN.

  If CTRL2[BTE] is set, FlexCAN uses ENCBT[NTSEG2] to configure time segment 2 in the range of 2 to 128 time quanta. For the data phase in CAN FD messages with BRS set, EDCBT[DTSEG2] must be used for configuring time segment 2 in the range of 2 to 16 time quanta.

NOTE

The bit time defined by the above time segments must not be smaller than five time quanta. For bit time calculations, use an Information Processing Time (IPT) of two, which is the value implemented in the FlexCAN module.

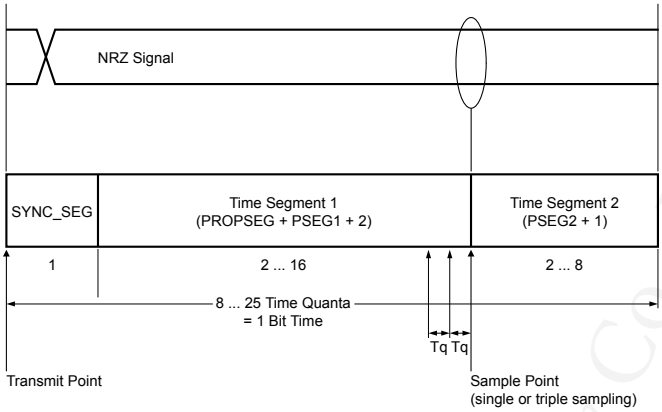


Figure 441. Segments within the bit time (example using CTRL1 bit timing variables for Classical CAN format)

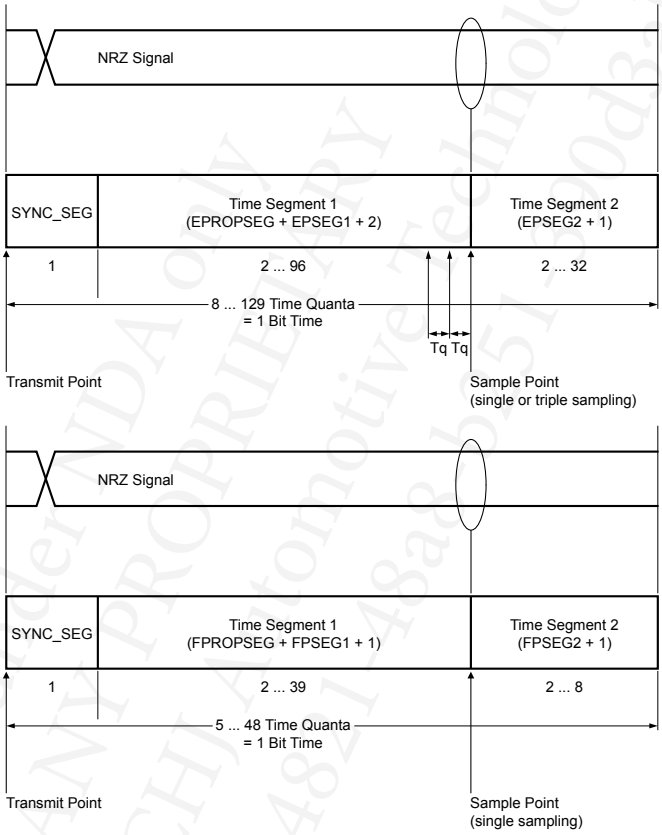


Figure 442. Segments within the bit time (example using CBT and FDCBT bit timing variables for CAN FD format)

Table 375. Time segment syntax

Syntax	Description
SYNC_SEG	System expects transitions to occur on the bus during this period.
TSEG1	Corresponds to the sum of PROPSEG and PSEG1.

Table continues on the next page...

Table 375. Time segment syntax (continued)

Syntax	Description
TSEG2	Corresponds to the PSEG2 value.
Transmit point	A node in transmit mode transfers a new value to the CAN bus at this point.
Sample point	A node samples the bus at this point. If the three samples per bit option is selected, then this point marks the position of the third sample.

The following table gives some examples of the CAN compliant segment settings for Classical CAN format (Bosch CAN 2.0B) (non-FD) messages.

Table 376. Bosch CAN 2.0B standard compliant bit time segment settings

Time segment 1	Time segment 2	Re-synchronization jump width
5 .. 10	2	1 .. 2
4 .. 11	3	1 .. 3
5 .. 12	4	1 .. 4
6 .. 13	5	1 .. 4
7 .. 14	6	1 .. 4
8 .. 15	7	1 .. 4
9 .. 16	8	1 .. 4

NOTE

The user must ensure the bit time settings are in compliance with the CAN Protocol standard (ISO 11898-1).

Whenever CAN bit is used as a measure of time duration (for example, estimating the occurrence of a CAN bit event in a message), the number of peripheral clocks in one CAN bit (NumClkBit) can be calculated as:

$$NumClkBit = \frac{f_{SYS}}{f_{CANCLK}} \times (PRESDIV + 1) \times (PROPSEG + PSEG1 + PSEG2 + 4)$$

or, if CTRL2[BTE] = 1:

$$NumClkBit = \frac{f_{SYS}}{f_{CANCLK}} \times (ENPRESDIV + 1) \times (NTSEG1 + NTSEG2 + 3)$$

where:

- NumClkBit is the number of peripheral clocks in one CAN bit.
- f_{CANCLK} is the Protocol Engine (PE) Clock (see Figure 439), in Hz.
- f_{SYS} is the frequency of operation of the system (CHI) clock, in Hz.
- PSEG1 is the value in CTRL1[PSEG1] field.
- PSEG2 is the value in CTRL1[PSEG2] field.

- PROPSEG is the value in CTRL1[PROPSEG] field.
- PRESDIV is the value in CTRL1[PRESDIV] field.
- ENPRESDIV is the value in EPRS[ENPRESDIV] field.
- NTSEG1 is the value in ENCBT[NTSEG1] field.
- NTSEG2 is the value in ENCBT[NTSEG2] field.

The formula above is also applicable to the alternative CAN bit timing variables described in the CAN Bit Timing register (CBT) or the Enhanced Nominal CAN bit timing (ENCBT) register and also to the CAN FD Bit Timing register (FDCBT) or the Enhanced Nominal CAN bit timing register(ENCBT).

For example, 180 CAN bits = (180 x NumClkBit) peripheral clock periods.

50.5.9.9 Arbitration and matching timing

During normal reception and transmission, the matching, arbitration, move-in, and move-out processes are executed during certain time windows inside the CAN frame, as shown in the following figures.

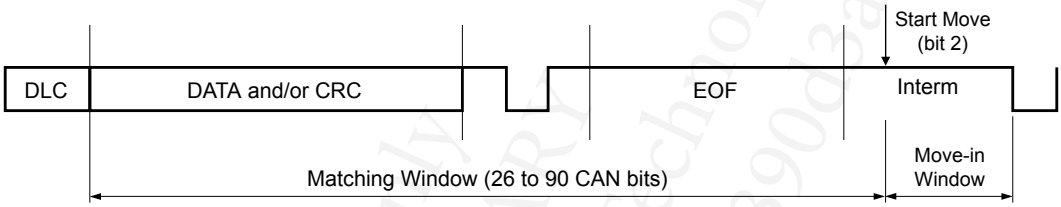


Figure 443. Matching and move-in time windows

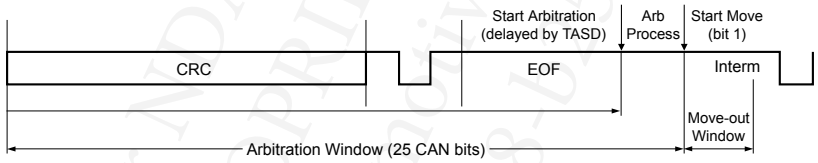


Figure 444. Arbitration and move-out time windows

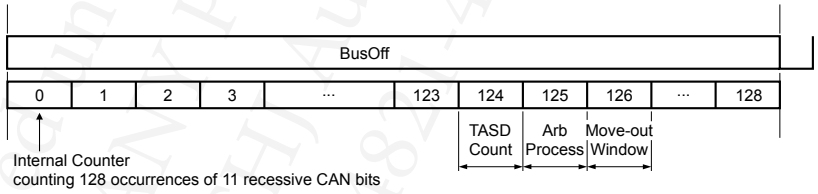


Figure 445. Arbitration at the end of bus off and move-out time windows

NOTE

In the preceding figures, the matching and arbitration timing does not take into account the delay caused by the concurrent memory access due to the CPU or other internal FlexCAN subblocks.

50.5.9.10 Tx arbitration start delay

CTRL2[TASD] (Tx Arbitration Start Delay) is a variable that indicates the number of CAN bits used by FlexCAN to delay the Tx arbitration process start point from the first bit of CRC field of the current frame. This variable can be written only in Freeze mode because it is blocked by hardware in other modes.

The transmission performance is impacted by the ability of the CPU to reconfigure message buffers (MBs) for transmission after the end of the internal arbitration process, where FlexCAN finds the winner MB for transmission (see Arbitration process). If the arbitration ends too early before the first bit of Intermission field, then there is a chance that the CPU reconfigures some Tx MBs and the winner MB is no longer the best candidate to be transmitted.

TASD is useful to optimize the transmission performance by defining the arbitration start point, as shown in the next figure, based on factors such as:

- Peripheral-to-oscillator clock ratio
- CAN bit timing variables that determine the CAN bit rate
- Number of message buffers (MBs) in use by the matching and arbitration processes

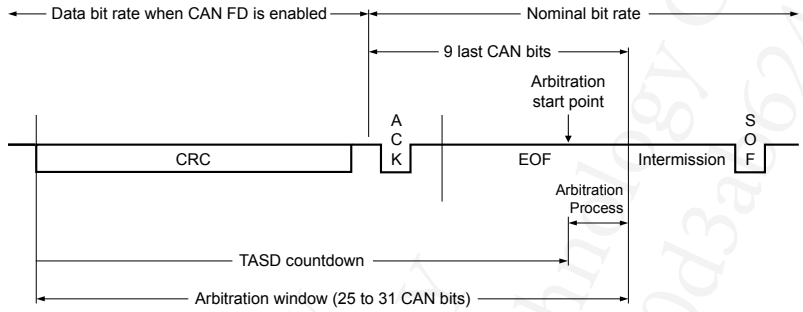


Figure 446. Optimal Tx Arbitration start point

The duration of an arbitration process, in terms of CAN bits, is directly proportional to the number of available MBs and to the CAN bit rate, and inversely proportional to the peripheral clock frequency.

The optimal arbitration timing is that in which the last MB is scanned just before the first bit of the Intermission field of a CAN frame. For instance, if there are few MBs and the peripheral/oscillator clock ratio is high and the CAN baud rate is low, then the arbitration can be placed closer to the frame's end, adding more delay to its start point, and vice-versa.

If TASD is set to 0 then the arbitration start is not delayed and more time is reserved for arbitration. On the other hand, if TASD is close to 24 then the CPU can configure a Tx MB later and less time is reserved for arbitration. If too little time is reserved for arbitration the FlexCAN may be not be able to find a winner MB in time to be transmitted with the best chance to win the bus arbitration against external nodes on the CAN bus.

The optimal TASD value can be calculated as follows:

For CAN FD frames and $(MAXMB + 1) \leq NMB_{END}$

$$TASD = 31 - \frac{2 * (MAXMB + 1) + 4}{CPCB_N}$$

For CAN FD frames and $(MAXMB + 1) > NMB_{END}$

$$TASD = 22 - \frac{2 * (MAXMB + 1) - NMB_{END}}{CPCB_F}$$

For non-FD frames

$$TASD = 25 - \frac{2 * (MAXMB + 1) + 4}{CPCB}$$

where:

$$NMB_{END} = \frac{(9 * CPCB_N) - 4}{2}$$

$$BITRATE_N = \left(\frac{f_{CANCLK}}{[1 + (EPSEG1 + 1) + (EPSEG2 + 1) + (EPROPSEG + 1)] \times (EPRESDIV + 1)} \right)$$

$$BITRATE_F = \left(\frac{f_{CANCLK}}{[1 + (FPSEG1 + 1) + (FPSEG2 + 1) + FPROPSEG] \times (FPRESDIV + 1)} \right)$$

$$CPCB_N = \frac{f_{SYS}}{BITRATE_N}$$

$$CPCB_F = \frac{f_{SYS}}{BITRATE_F}$$

$$CPCB = CPCB_N$$

- MAXMB is the value in CTRL1[MAXMB].
- NMB_{END} is the number of message buffers that can be scanned by the arbitration process during the 9 last CAN bits at the end of a frame (see the figure above).
- $BITRATE_N$ is the CAN bit rate in bits per second calculated by the nominal CAN bit time variables.
- $BITRATE_F$ is the CAN bit rate in bits per second calculated by the data CAN bit time variables.
- $CPCB_N$ is the number of peripheral clocks per CAN bit in nominal bit rate for CAN FD frames.
- $CPCB_F$ is the number of peripheral clocks per CAN bit in data bit rate for CAN FD frames.
- CPCB is the number of peripheral clocks per CAN bit for non-FD frames.
- f_{CANCLK} is the oscillator clock, in Hz.
- f_{SYS} is the peripheral clock, in Hz.
- EPSEG1 is the value in CBT[EPSEG1] (CTRL1[PSEG1] can also be used).
- EPSEG2 is the value in CBT[EPSEG2] (CTRL1[PSEG2] can also be used).
- EPROPSEG is the value in CBT[EPROPSEG] (CTRL1[PROPSEG] can also be used).
- EPRESDIV is the value in CBT[EPRESDIV] (CTRL1[PRESDIV] can also be used).
- FPSEG1 is the value in FDCBT[FPSEG1].
- FPSEG2 is the value in FDCBT[FPSEG2].
- FPROPSEG is the value in FDCBT[FPROPSEG].
- FPRESDIV is the value in FDCBT[FPRESDIV].
- NTSEG1 is the value in ENCBT[NTSEG1].
- NTSEG2 is the value in ENCBT[NTSEG2].
- ENPRESDIV is the value in EPRS[ENPRESDIV].
- DTSEG1 is the value in EDCBT[DTSEG1].
- DTSEG2 is the value in EDCBT[DTSEG2].
- EDPRESDIV is the value in EPRS[EDPRESDIV].

If CTRL2[BTE] is set, then:

$$BITRATE_N = \frac{f_{CANCLK}}{[1 + (NTSEG1 + 1) + (NTSEG2 + 1)] \times (ENPRESDIV + 1)}$$

Provided by IHS Markit
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a2-8251-39003b624c1

$$BITRATE_F = \frac{f_{CANCLK}}{[1 + (DTSEG1 + 1) + (DTSEG2 + 1)] \times (EDPRESDIV + 1)}$$

See also Protocol timing for more details.

The following tables give the TASD value calculated for some configuration cases.

Case 1:

- Clock ratio = 2:1 (example: peripheral clock 80 MHz and oscillator clock 40 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Table 377. TASD values:

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid
32	24	8.0
64	23	8.0
96	22	8.0
100	22	8.0
128	21	3.6

Case 2:

- Clock ratio = 1:1 (example: peripheral clock 40 MHz and oscillator clock 40 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Table 378. TASD values:

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid
32	23	6.67
54	22	5.0
64	21	3.33
96	20	1.6
128	18	1.05

Case 3:

- Clock ratio = 2:1 (example: peripheral clock 40 MHz and oscillator clock 20 MHz)
- Bit rate in arbitration phase = 1 Mbaud

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c262e5-4821-48a8-b251-990d3ab624c1

Table 379. TASD values:

Number of message buffers	TASD value	Maximum bit rate in data phase (MBd)
16	24	Invalid
32	23	4.0
54	22	4.0
64	21	3.33
96	20	1.54
128	18	1.05

50.5.10 Clock domains and restrictions

The FlexCAN module has two clock domains asynchronous to each other:

- The bus domain feeds the Control Host Interface (CHI) submodule and is derived from the peripheral clock.
- The oscillator domain feeds the CAN Protocol Engine (PE) submodule and is derived directly from a crystal oscillator clock, so that very low jitter performance can be achieved on the CAN bus.

When the two domains are connected to clocks with different frequencies and/or phases, there are restrictions on the frequency relationship between the two clock domains. In the case of asynchronous operation, the bus domain clock frequency must always be greater than the oscillator domain clock frequency.

NOTE

Asynchronous operation with a 1:1 ratio between peripheral and oscillator clocks is not allowed.

When doing matching and arbitration, FlexCAN needs to scan the whole message buffer memory during the time slot of one CAN frame, comprised of a number of CAN bits. In order to have sufficient time to do that, the following requirements must be observed:

- The peripheral clock frequency cannot be smaller than the oscillator clock frequency.
- There must be a minimum number of peripheral clocks per CAN bit, as specified in the table shown below.

Table 380. Minimum number of peripheral clocks per CAN bit for Classical CAN format

Number of mailboxes	Value of MCR[RFEN]	Value of ERFCR[ERFEN]	Minimum number of peripheral clocks per CAN bit
16	0	0	16
32	0	0	16
64	0	0	25
96	0	0	37
128	0	0	49
16	1	0	16
32	1	0	17

Table continues on the next page...

Table 380. Minimum number of peripheral clocks per CAN bit for Classical CAN format (continued)

Number of mailboxes	Value of MCR[RFEN]	Value of ERFCR[ERFEN]	Minimum number of peripheral clocks per CAN bit
64	1	0	30
96	1	0	42
128	1	0	54
16	0	1	16
32	0	1	19
64	0	1	31
96	0	1	42
128	0	1	55

For classical frame format, the minimum number of peripheral clocks per CAN bit specified in the preceding table determines the minimum peripheral clock frequency for a given number of mailboxes and for an expected CAN bit rate. The CAN bit rate depends on the number of time quanta in a CAN bit, which can be defined by adjusting one or more of the bit timing values contained in either the Control 1 register (CTRL1) or CAN Bit Time register (CBT) or Enhanced Nominal CAN bit timing register (ENCBT). The time quantum (Tq) is defined in Protocol timing. The minimum number of time quanta per CAN bit must be 8; therefore, the oscillator clock frequency should be at least 8 times the CAN bit rate.

For CAN FD frame format, there are some constraints that need to be satisfied. The number of peripheral clocks per CAN bit in nominal bit rate (NumClkNomBit) can be calculated by the equation below.

$$\text{NumClkNomBit} = \frac{f_{SYS}}{f_{CANCLK}} \times (\text{PRESDIV} + 1) \times (\text{PROPSEG} + \text{PSEG1} + \text{PSEG2} + 4)$$

$$= \frac{f_{SYS}}{\text{NomBitRate}}$$

where PRESDIV, PSEG1, and PSEG2 are CAN bit time values in CTRL1 register. Alternatively, EPRESDIV, EPSEG1, and EPSEG2 values in CBT register or the values of EPRS[ENPRESDIV], ENCBT[NTSEG1], and ENCBT[NTSEG2] can be used instead. NumClkNomBit can also be calculated as a function of the expected nominal bit rate used in the arbitration phase (NomBitRate), as shown in the equation above.

The number of CAN bits in the data phase of an FD frame with the BRS bit set (fast CAN bits, in short) depends on the number of data bytes in the payload. The number of fast CAN bits (NumOfFastBits) can be determined in the table below. The fewer the number of data bytes, the fewer the number of fast CAN bits, and less time is available for FlexCAN to scan the whole message buffer memory during the internal matching and arbitration processes.

Table 381. Number of fast CAN bits in a CAN FD frame

Minimum number of data bytes	DLC field	NumOfFastBits
0	0h	21
1	1h	29

Table continues on the next page...

Table 381. Number of fast CAN bits in a CAN FD frame (continued)

Minimum number of data bytes	DLC field	NumOfFastBits
2	2h	37
3	3h	45
4	4h	53
5	5h	61
6	6h	69
7	7h	77
8	8h	85
12	9h	117
16	Ah	149
20	Bh	186
24	Ch	218
32	Dh	282
48	Eh	410
64	Fh	538

The critical part of a CAN FD frame is during the data phase, where the CAN bit rate is faster than in the arbitration phase. The minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit) can be calculated to guarantee that enough time is available for FlexCAN to scan the message buffer memory during reception and transmission. The equation below calculates this constraint.

$$MinNumClkFastBit_A = \frac{(8.5 \times MaxNumOfMb) + [ERFEN \times (2 \times NFE + 4)] + 64 - (9 \times NumClkNomBit)}{NumOfFastBits}$$

where MaxNumOfMb is the maximum number of available mailboxes defined in MCR[MAXMB]. NFE and ERFEN are the fields defined in ERFCR register.

The clock domain crossing circuit between the CHI and PE subblocks also imposes a minimum number of peripheral clocks per fast CAN bit for the handshake mechanism to work properly without losing status information through the interface, as shown in the equation below.

$$MinNumClkFastBit_B = 3 \times \left(1 + \frac{f_{SYS}}{f_{CANCLK}}\right)$$

Therefore, the minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit) is determined by the larger of the two values calculated above.

$$MinNumClkFastBit = Maximum\ (\ MinNumClkFastBit_A, MinNumClkFastBit_B\ )$$

Then, the maximum CAN bit rate in the data phase of CAN FD frames (DataBitRateMAX) can be calculated as below.

$$DataBitrate_{MAX} = \frac{f_{CANCLK}}{ROUNDUP\left(\frac{MinNumClkFastBit \times f_{CANCLK}}{f_{SYS}}\right)}$$

The peripheral and oscillator clock frequencies, the maximum number of mailboxes, and the expected nominal bit rate affect the maximum data bit rate attainable by FlexCAN in CAN FD mode. Also, the data bit rate depends on the minimum payload size of FD frames used in a given application.

To illustrate how the CAN FD bit rate is affected by the configuration of FlexCAN variables, an application example with the peripheral and oscillator clock frequencies set to 50 MHz and 40 MHz, respectively, is considered.

1. Considering the nominal bit rate as 1 Mbps, the number of peripheral clocks per CAN bit in nominal bit rate is calculated as below.

$$NumClkNomBit = \frac{50 \times 10^6}{1 \times 10^6} = 50$$

2. The number of fast CAN bits (NumOfFastBits) is determined in the table presented above. For example, if the minimum payload in FD frames is 8 bytes, then there are 85 CAN bits in the data phase.
3. Assuming the maximum number of mailboxes is 96, and Enhanced Rx FIFO is disabled, the minimum number of peripheral clocks per fast CAN bit (MinNumClkFastBit) can be calculated.

$$MinNumClkFastBit_A = \frac{(8.5 \times 96) + 64 - (9 \times 50)}{85} = 5.06$$

$$MinNumClkFastBit_B = 3 \times \left(1 + \frac{50}{40}\right) = 6.75$$

$$MinNumClkFastBit = Maximum\ (5.06, 6.75) = 6.75$$

4. The maximum CAN bit rate in the data phase can finally be found.

$$DataBitRate_{MAX} = \frac{40 \times 10^6}{ROUNDUP\left(\frac{6.75 \times 40 \times 10^6}{50 \times 10^6}\right)} = 6.667\ Mbps$$

As demonstrated in this example, even though the oscillator clock frequency (40 MHz) is adequate to generate a data rate of 8 Mbps in CAN FD mode, the specific FlexCAN configuration limits this rate to 6.667 Mbps. This limitation is mainly due to the low peripheral clock frequency that imposes the MinNumClkFastBitB bound.

The table below shows the maximum data rate for CAN FD with Enhanced RX FIFO disabled according to clock frequencies, payload size, and number of available mailboxes. See in this table that, for some cases, if the number of available mailboxes is reduced, the FlexCAN can then achieve a data rate up to 8 Mbps.

Table 382. Maximum CAN bit rate in data phase on CAN FD frames with Enhanced Rx FIFO disabled

Peripheral clock frequency (MHz)	Payload size	Number of available mailboxes	Maximum data rate (Mbps)
40	8	94	6.667

Table continues on the next page...

Table 382. Maximum CAN bit rate in data phase on CAN FD frames with Enhanced Rx FIFO disabled (continued)

Peripheral clock frequency (MHz)	Payload size	Number of available mailboxes	Maximum data rate (Mbps)
40	8	114	5.0
40	12	117	6.667
40	12	128	5.714
50	12 to 64	128	6.667
60	8	126	8.0
60	12	128	8.0
67	6	128	8.0
80	3	128	8.0
100	0	128	8.0

50.5.11 Modes of operation details

The FlexCAN module has functional modes and low-power modes. See Modes of operation for an introductory description of all the modes of operation. The following sub-sections contain functional details on Freeze mode and the low-power modes.

CAUTION

"Permanent Dominant" failure on CAN Bus line is not supported by FlexCAN. If a Low-Power request or Freeze mode request is done during a "Permanent Dominant", the corresponding acknowledge can never be asserted.

50.5.11.1 Freeze mode

This mode is requested either by the CPU through the assertion of MCR[HALT] or when the chip is put into Debug mode. In both cases it is also necessary that MCR[FRZ] be asserted and the module not be in a low-power mode. This mode is also requested by FlexCAN through the automatic assertion of both MCR[HALT] and MCR[FRZ] when MECR[NCEFAFRZ] is set and a non-correctable error is detected in a memory read access performed by FlexCAN internal processes (see Response to errors).

The acknowledgement is obtained through the assertion by the FlexCAN of MECR[FRZ_ACK]. The CPU must only consider the FlexCAN in Freeze mode when both request and acknowledgement conditions are satisfied.

When Freeze mode is requested, FlexCAN does the following:

- Waits to be in either Intermission, Passive Error, Bus Off, or Idle state.
- Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. A pending move-in does not prevent entering Freeze mode.
- Ignores the Rx input pin and drives the Tx pin as recessive.
- Stops the prescaler, thus halting all CAN protocol activities.
- Grants write access to the Error Counters register, which is read-only in other modes.
- Sets MCR[NOTRDY] and MCR[FRZACK].

After requesting Freeze mode, the user must wait for MCR[FRZ_ACK] to be asserted before executing any other action; otherwise FlexCAN may operate in an unpredictable way. In Freeze mode, all memory-mapped registers are accessible.

Exiting Freeze mode is done in one of the following ways:

- CPU negates MCR[FRZ].
- The chip is removed from Debug mode and/or the HALT bit is negated.

MCR[FRZ_ACK] is negated after the protocol engine recognizes the negation of the freeze request. When out of Freeze mode, FlexCAN tries to resynchronize to the CAN bus by waiting for 11 consecutive recessive bits.

50.5.11.2 Module Disable mode

This low-power mode is normally used to temporarily disable a complete FlexCAN block, with no power consumption. It is requested by the CPU through the assertion of MCR[MDIS], and the acknowledgement is obtained through the assertion by the FlexCAN of MCR[LPMACK]. The CPU must only consider the FlexCAN in Disable mode when both request and acknowledgement conditions are satisfied.

If the module is disabled during Freeze mode, it requests to disable the clocks to the PE and CHI submodules, sets MCR[LPMACK] and negates MCR[FRZACK].

If the module is disabled during transmission or reception, FlexCAN does the following:

- Waits to be in either Idle or Bus Off state, or else waits for the third bit of Intermission and then checks it to be recessive.
- Waits for all internal activities like arbitration, matching, move-in, and move-out to finish. A pending move-in is not taken into account.
- Ignores its Rx input pin and drives its Tx pin as recessive.
- Shuts down the clocks to the PE and CHI submodules
- Sets MCR[NOTRDY] and MCR[LPMACK].

The Bus Interface Unit continues to operate, enabling the CPU to access memory-mapped registers, except the Rx Mailboxes Global Mask registers, the Rx Buffer 14 Mask register, the Rx Buffer 15 Mask register,. The message buffers, the Rx Individual Mask registers, and the reserved words within RAM may not be accessed when the module is in Disable mode. Exiting from this mode is done by negating the MDIS bit by the CPU, which causes the FlexCAN to request to resume the clocks and negate the LPMACK bit after the CAN protocol engine recognizes the negation of disable mode requested by the CPU.

50.5.12 Interrupts

The module has many interrupt sources: interrupts due to message buffers and interrupts due to the ORed interrupts from MBs, Bus Off, Bus Off Done, Error, Error Fast (errors detected in the data phase of CAN FD format messages with the BRS bit set), Tx Warning, and Rx Warning.

Each one of the message buffers can be an interrupt source, if its corresponding IMASK bit is set. There is no distinction between Tx and Rx interrupts for a particular buffer, under the assumption that the buffer is initialized for either transmission or reception. Each of the buffers has an assigned flag bit in the IFLAG registers. The bit is set when the corresponding buffer completes a successful transfer and is cleared when the CPU writes one to it (unless another interrupt is generated at the same time).

NOTE

It must be guaranteed that the CPU clears only the bit causing the current interrupt. For this reason, bit manipulation instructions (BSET) must not be used to clear interrupt flags. These instructions may cause accidental clearing of interrupt flags which are set after entering the current interrupt service routine.

If the Legacy Rx FIFO is enabled (MCR[RFEN] = 1) and DMA is disabled (MCR[DMA] = 0), the interrupts corresponding to MBs 0 to 7 have different meanings. Bit 7 of the IFLAG1 register becomes the "Legacy FIFO Overflow" flag; bit 6 becomes the "Legacy FIFO Warning" flag, bit 5 becomes the "Frames Available in Legacy FIFO" flag and bits 4-0 are unused. See the description of Interrupt Flags 1 register (IFLAG1) for more information.

If both Legacy Rx FIFO and DMA are enabled (MCR[RFEN] and MCR[DMA] = 1) the FlexCAN does not generate any Legacy FIFO interrupt. Bit 5 of the IFLAG1 register still indicates "Frames Available in Legacy FIFO" and generates a DMA request. Bits 7, 6, 4-0 are unused.

CAUTION

Legacy FIFO cannot be enabled when CAN FD feature is enabled.

For a combined interrupt where multiple MB interrupt sources are OR'd together, the interrupt is generated when any of the associated MBs (or FIFO, if applicable) generates an interrupt. In this case, the CPU must read the IFLAG registers to determine which MB or FIFO source caused the interrupt.

The following interrupt sources generate interrupts like the MB interrupt sources, and can be read from ESR1 register

- Bus Off
- Bus Off Done
- Error
- Error Fast
- Tx Warning
- Rx Warning

The Bus Off, Error, Tx Warning, and Rx Warning interrupt mask bits are located in the CTRL1 register.

50.5.13 Bus interface

CPU access to FlexCAN registers is subject to the following rules:

- Read and write access to implemented reserved address space results in an access error.
- Write access to positions whose bits are all currently read-only results in an access error. If at least one of the bits is not read-only then no access error is issued. Write permission to positions or some of their bits can change depending on the mode of operation or transitory state. See register and field descriptions for details.
- Read and write access to unimplemented address space results in access error.
- Read and write access to RAM located positions during Low Power mode results in an access error.
- It is possible for the RXIMR memory region to be considered as general purpose memory and available for access. There are two ways of doing this:
    1. If MCR[IRMQ] is cleared, the individual masks (RXIMR) are disabled. In this case the RXIMR memory region is considered as general purpose memory.
    2. If MCR[MAXMB] is programmed with a value smaller than the available number of MBs, then the unused memory space can be used as general purpose RAM space. Note that reserved words within RAM cannot be used. As an example, suppose FlexCAN's RAM can support up to 16 MBs, CTRL2[RFFN] is 0h, and MCR[MAXMB] is programmed with zero. The maximum number of MBs in this case becomes one. The RAM starts at 0080h, and the space 0080h–008Fh is used by the one MB. The memory space 0090h–017Fh is available. The space 0180h–087Fh is reserved. The space 0880h–0883h is used by the one individual mask and the available memory in the mask registers space would be 0884h–08BFh. From 08C0h–09DFh there are reserved words for internal use which cannot be used as general purpose RAM. As a general rule, free memory space for general purpose depends only on MAXMB.
    3. If MCR[FDEN] is enabled, general purpose memory can be used out of Freeze mode only.

50.5.14 Detection and correction of memory errors

NOTE

All FlexCAN memory must be initialized before starting its operation in order to have the parity bits in memory properly updated. CTRL2[WRMFRZ] grants write access to all memory positions that require initialization, ranging from 080h–ADFh and from C20h–31FFh. The RXMGMASK, RX14MASK, RX15MASK, and RXFGMASK registers need to be initialized as well. MCR[RFEN] and ERF[ERFEN] must not be set during memory initialization.

FlexCAN supports detection and correction of errors in memory read accesses. Each byte of FlexCAN memory is associated to five parity bits that ensure a Hamming distance of four. The error correction mechanism ensures that in this 13-bit word, errors in one bit can be corrected (correctable errors) and errors in two bits can be detected but not corrected (non-correctable errors). Errors in more than two bits may not be detected. In case of non-correctable errors, the corrupted data is not changed by the error correction logic. When a read access is performed, the parity bits are used to calculate a syndrome, which indicates the error in each byte.

FlexCAN detects a noncorrectable error in the event either an all-zeros or an all-ones read occurs. See description of Error Report Syndrome register (RERRSYNR).

Memory errors are indicated to the host through status register (Error Status Register (ERRSR)) and bus transfer errors, and reported through report registers (Error Report Address register (RERRAR), Error Report Data register (RERRDR) and Error Report Syndrome register (RERRSYNR)).

The error detection and correction mechanism can be activated or not, controlled by MECR[ECCDIS]. When disabled, updates on indications and reporting registers are stopped, but the parity bits are still calculated and written along with data in memory write operations to ensure that memory has consistent parity bits associated with the data.

To avoid accidentally changing the critical error correction configuration, this protocol must be followed to enable the update of the Memory Error Control register (MECR):

1. By default, CTRL2[ECRWRE] is zero and MECR[ECRWRDIS] is one.
2. Set CTRL2[ECRWRE].
3. Clear MECR[ECRWRDIS].
4. All writes to Memory Error Control register (MECR) must keep MECR[ECRWRDIS] cleared.
5. After configuration is done, lock the Memory Error Control register (MECR) by either setting MECR[ECRWRDIS] or clearing CTRL2[ECRWRE].

50.5.14.1 Sources of the memory access

The FlexCAN memory can be accessed by two major sources (or requestors):

- Host (CPU): the largest word accessed is 32-bit.
- FlexCAN internal processes (Rx matching, Tx arbitration, move-in on reception, move-out on transmission): the largest word accessed is 64-bit.

The way that noncorrectable errors are indicated and reported depends on the source of access.

50.5.14.2 Error indication

Memory errors are indicated by flags HANCEIF, FANCEIF, and CEIF in the Error Status Register (ERRSR). Noncorrectable errors detected in memory reads requested by the host are indicated separately than the ones detected in requests by FlexCAN internal processes. FlexCAN makes no distinction of the source of the access when correctable errors are detected. There are three independent flags for these three cases. If both noncorrectable and correctable errors are found in different bytes in the same read operation, both flags are set.

A noncorrectable error detected in host access is also indicated as a bus transfer error. A bus wait request may be asserted to extend the memory transaction to the moment the report registers are updated. This indication cannot be masked.

Each indication flag has one overrun flag in Error Status Register (ERRSR). The overrun flags do not request interrupts. Overrun flags for noncorrectable errors indicate that other errors of the same nature were detected after current error being treated; overrun flags for correctable errors indicate that other errors of the same nature were detected before the current error being treated.

This is the recommended handling sequence for error indication:

1. Get error report information from report registers.
2. Use this information to take proper measures in the application.
3. Clear the HANCEIF, FANCEIF, and CEIF flags.

4. If the overrun flag is active:
    a. Alert application that at least one error could not be managed.
    b. Clear the overrun flag.

The FlexCAN internal processes can access memory in transactions larger than 32 bits. For the indication, this kind of access is considered a consecutive sequence of 32-bit accesses. If errors are found in two or more 32-bit words the interrupt and overrun flags are set simultaneously.

50.5.14.3 Error reporting

The report registers Error Report Address register (RERRAR), Error Report Data register (RERRDR), and Error Report Syndrome register (RERRSYNR) provide detailed information about the address read, raw data, and syndrome read with error, and are indicated by the flags described in Error indication. The address, data, and syndrome registers are updated simultaneously along with the error flags, according to these rules:

1. If either of the two non-correctable error flags is currently set, the report registers are not updated (the previous non-correctable error reporting is preserved).
2. Otherwise (either no error flag is currently set or only the correctable error flag is currently set), the report registers are updated according to the new error; or according to the most severe of new errors if non-correctable and correctable errors are simultaneously detected.

Reporting of errors detected in accesses larger than 32-bit follows the rules described in Error indication and in the description of Error Report Address register (RERRAR).

The address reported in RERRAR and defined in ERRIAR are not the same listed in the module memory map. The relation between the reported addresses and the respective ones in the module memory map is shown in the description of Error Injection Address register (ERRIAR).

Addresses reported when reading memory portions organized as FIFOs, such as the Legacy Rx FIFO Structure, Enhanced RX FIFO Structure and the Rx FIFO Information Register (RXFIR), refer to the address of the specific entry accessed in the FIFO, not to the FIFO base address.

To assure coherence of the error report registers it is necessary to turn off the report update by setting MECR[RERRDIS] before reading the report registers.

50.5.14.4 Response to errors

Correctable errors have no consequence on FlexCAN operation because affected data is corrected before its use by the host or FlexCAN internal processes.

For host-initiated reads, a noncorrectable error may affect the host, but does not affect FlexCAN operation.

Non-correctable errors detected on memory reads requested by the FlexCAN internal processes may result in incorrect operation depending on the state of MECR[NCEFAFRZ], as follows:

- During reception (either matching or move-in processes), when a noncorrectable error occurs, an incorrect destination may be selected to store the incoming frame, a corrupted frame may be stored in the correct destination, or both. If MECR[NCEFAFRZ] is set, FlexCAN stops operation automatically and enters in Freeze mode to prevent corrupted data from being treated as valid by FlexCAN internal processes. When MECR[NCEFAFRZ] is negated, FlexCAN continues working and a corrupted frame is received.
- During arbitration process, when a noncorrectable error occurs, either a non-highest priority Tx message buffer may be mistakenly selected for transmission or its data may be corrupted. If MECR[NCEFAFRZ] is set, FlexCAN stops operation automatically and enters Freeze mode before starting the move-out. When MECR[NCEFAFRZ] is negated, FlexCAN proceeds to move-out with a corrupted frame that will be transmitted on the CAN bus.
- During move-out process, when a noncorrectable error occurs, a corrupted frame is copied from the selected Tx MB that won the arbitration to the Tx SMB for transmission. If MECR[NCEFAFRZ] is set, FlexCAN stops operation automatically and enters Freeze mode before starting the transmission. When MECR[NCEFAFRZ] is negated, the corrupted frame is transferred from the Tx SMB to the Protocol Engine (PE) sub-block and is transmitted on the CAN bus.

- A noncorrectable error can also be detected beyond the move-out process, when Tx data is read from Tx SMB (buffer located in RAM) to be transferred to the PE subblock for transmission. In this case, a frame with corrupted ID and/or data is transmitted on the CAN bus.

  To prevent the frame from being successfully received by the external nodes, FlexCAN inverts all bits in the CRC field (CRC sequence plus CRC delimiter), and transmits an error flag just after CRC delimiter as a result of self-detecting a Bit1 error and a form error due to the CRC field inversion. When MECR[NCEFAFRZ] is set, FlexCAN stops operation automatically and enters Freeze mode just after the error frame. When MECR[NCEFAFRZ] is negated, FlexCAN may attempt to re-transmit the same frame, as long as no other higher priority Tx MB is subsequently configured for transmission.

  In the event the noncorrectable error persists, FlexCAN eventually reaches the Bus Off state because of consecutive error detections. ECR[TXERRCNT] is updated every time FlexCAN inverts the CRC field, causing errors as described above.

When MECR[NCEFAFRZ] is set and FlexCAN enters Freeze mode, only the CPU can cause FlexCAN to exit Freeze mode and resume Normal mode. The assertion of MECR[NCEFAFRZ] is the only way to prevent corrupted frames from being transmitted on the CAN bus up to the move-out internal process.

The error report registers can provide information to the application for customized management of these situations.

50.5.14.5 Error injection

The error injection registers ERRIAR, ERRIDPR, and ERRIPPR are used to inject errors in memory reads in order to force errors and consequently update the indication and reporting registers. The relation between the error injection addresses and the respective ones in the module memory map is shown in the description of Error Injection Address register (ERRIAR).

The injection is done by flipping the data and parity bits correspondent to the bits in 1 in ERRIDPR and ERRIPPR. Injection can be selected specifically for memory accesses requested by the host or by FlexCAN internal processes.

In case of accesses larger than 32-bits, the EXTERRIE bit in Memory Error Control register (MECR) extends the injection pattern, replicating it in 32-bit words to fill the width of the access.

NOTE

It is very unlikely, but error injection may correct a bit with error. This will not raise the error flags and reports as expected.

To ensure coherence among error injection registers and avoid spurious error injections, MECR[HAERRIE] and MECR[FAERRIE] must be cleared when configuring the memory injection registers.

50.6 Initialization/application information

This section provides instructions for initializing the FlexCAN module.

50.6.1 FlexCAN initialization sequence

You can reset FlexCAN in the following ways:

1. Chip level hard reset, which resets all memory-mapped registers asynchronously.
2. MCR[SOFTRST], which resets some of the memory-mapped registers synchronously. See Table 338 to see what registers are affected by soft reset.

Soft reset is synchronous and has to follow an internal request/acknowledge procedure across clock domains. Therefore, it may take some time to fully propagate its effects. MCR[SOFTRST] remains asserted while soft reset is pending, so software can poll this bit to know when the reset has completed. Also, soft reset cannot be applied while clocks are shut down in a low-power mode. The low-power mode should be exited and the clocks resumed before applying soft reset.

When the module is enabled (MCR[MDIS] negated), FlexCAN automatically enters Freeze mode. In Freeze mode, FlexCAN is un-synchronized to the CAN bus, MCR[HALT] and MCR[FRZ] are set, the internal state machines are disabled, then MCR[FRZACK] and MCR[NOTRDY] are set. The Tx pin is in recessive state and FlexCAN does not initiate any transmission or reception of CAN frames. Note that the message buffers and the Rx Individual Mask registers are not affected by reset, so they are not automatically initialized.

For any configuration change/initialization it is required that FlexCAN be put into Freeze mode (see Freeze mode). The following is a generic initialization sequence applicable to the FlexCAN module:

1. Initialize the Module Configuration register (MCR).
    a. Enable the individual filtering per MB and reception queue features by setting IRMQ.
    b. Enable the warning interrupts by setting WRNEN.
    c. If required, disable frame self reception by setting SRXDIS.
    d. Enable the Legacy Rx FIFO by setting RFEN or the Enhanced Rx FIFO by setting ERFEN.
    e. If Legacy Rx FIFO or Enhanced Rx FIFO is enabled and DMA is required, set DMA.
    f. Enable the abort mechanism by setting AEN.
    g. Enable the local priority feature by setting LPRIOEN.
2. Initialize the Control 1 register (CTRL1) and optionally the CAN Bit Timing register (CBT). Initialize also the CAN FD CAN Bit Timing register (FDCBT).
    a. Determine the bit timing parameters: PROPSEG, PSEG1, PSEG2, and RJW.
    b. Optionally determine the bit timing parameters: EPROPSEG, EPSEG1, EPSEG2, and ERJW.
    c. Determine the CAN FD bit timing parameters: FPROPSEG, FPSEG1, FPSEG2, and FRJW.
    d. Determine the bit rate by programming the PRESDIV field and optionally the EPRESDIV field.
    e. Determine the CAN FD bit rate by programming the FPRESDIV field.
    f. Determine the internal arbitration mode (LBUF).
3. All FlexCAN memory must be initialized if Error Code Correction (ECC) is enabled. See Detection and correction of memory errors.
4. Initialize the message buffers.
    a. The control and status word of all message buffers must be initialized.
    b. If Rx FIFO was enabled, the ID filter table must be initialized.
    c. Other entries in each message buffer should be initialized as required.
5. Initialize the Rx Individual Mask registers (RXIMRn).
6. Set required interrupt mask bits in
    - IMASK registers (for all MB interrupts)
    - CTRL1 / CTRL2 registers (for Bus Off and Error interrupts)
7. Negate MCR[HALT].

After the last step listed above, FlexCAN attempts to synchronize to the CAN bus.

50.7 Glossary

Active message buffer	A message buffer is *active* if it can participate in the current matching or arbitration process.
Bus interface unit	FlexCAN submodule responsible for the interface to the CPU.
Bus off	See CAN Specification.[76]
CAN	Controller Area Network, a serial communication protocol defined in CAN Specification[76] and ISO International Standard.[77]

[9] Controller Area Network – CAN Specification Version 2.0 Part A, Part B, Robert Bosch GmbH, 1991.

CHI	Controller-host interface, a FlexCAN submodule responsible for message buffer matching and arbitration algorithms.
CRC	Cyclic redundancy check
Dominant bit	A dominant bit wins the arbitration on the CAN bus. It is transmitted as 0.
Doze mode	A system low-power mode where the CPU bus remains active and a global Doze mode request is sent to all peripherals asking them to enter low power mode.
Error active	See CAN Specification.[76]
Error frame	See CAN Specification.[76]
Error passive	See CAN Specification.[76]
Form error	See CAN Specification.[76]
Hard reset	A reset coming from an external pin and/or following power-on. It resets everything.
Idle	See CAN Specification.[76]
Information processing time	See CAN Specification.[76]
Intermission	See CAN Specification.[76]
Matching elements	Data used in the matching process, such as ID, filter, mask, etc.
MB	See Message buffer.
Message buffer	Internal FlexCAN data structure containing bytes received from, or to be transmitted to, the CAN line, as well as information about this data.
Overload frame	See CAN Specification.[76]
Phase buffer segment	See CAN Specification.[76]
Recessive bit	A recessive bit loses the arbitration on the CAN bus. It is transmitted as 1.
Sclock	Serial clock. Obtained by dividing the clock feeding the CAN engine (oscillator or bus clock) by a prescaler factor. The Sclock period defines the time quantum for CAN protocol timing.
Soft reset	Global reset typically used by peripherals to reinitialize some of its registers, but not all of them.
Stop mode	A system low-power mode in which all chip clocks are stopped for maximum power savings.
Stuffing error	See CAN Specification.[76]
Time quantum	This is equal to the Sclock period. It is the minimum time period used to compose the CAN protocol bit timing.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4c8b-b251-90d3ab624c1

[10] ISO International Standard – ISO 11898 First Edition 1993 Road Vehicles – Interchange of Digital Information – Controller Area Network (CAN) for high-speed Communication.

Chapter 51
Timestamp

51.1 Timestamping and distribution

The CAN Timestamp function enables the distribution of a common timestamp base to all CAN interfaces on the chip, which enables you to temporally correlate frames received across all CAN module instances.

CAN Timestamp supports:

- Multiple timestamp sources, selectable via software:
  - Ethernet PTP timestamp counter (32-bit)
  - PFE PTP Ethernet PTP timestamp counter (32-bit)
  - STM_7 (System Timer Module) counter output (32-bit)
- A timestamp synchronization and distribution submodule

51.2 Block diagram

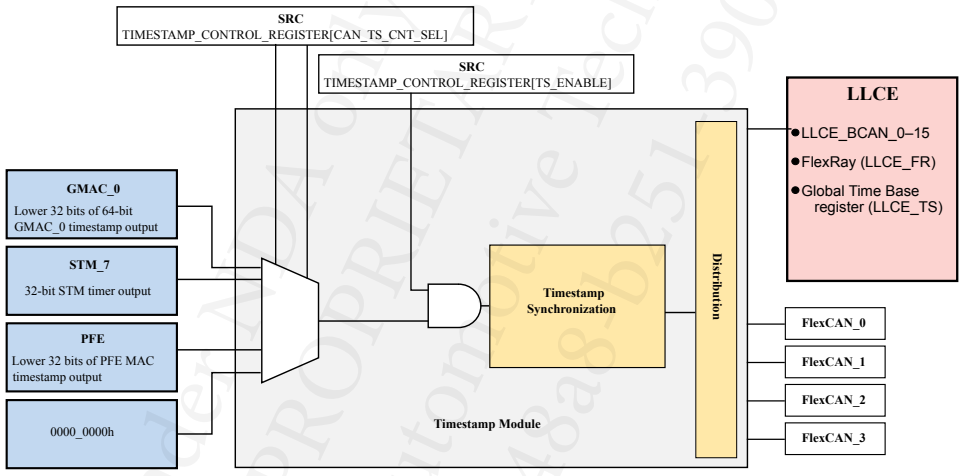


Figure 447. Timestamping connectivity diagram

The block diagram shows:

- Logic that selects the timestamp source
- Timestamping enabling logic

The timestamp logic uses the XBAR_DIV2_CLK 200 MHz clock for the timestamp clock domain to ensure the timestamp value delivered to the CAN modules (timestamp sinks) is:

- As accurate as possible
- Delivered without any missing counts

Refer to the LLCE and CAN chapters for details on how the timestamp value is used.

51.3 Timestamp synchronization controls

Configure and control CAN Timestamp via the Time Stamp Control register (TIME_STAMP_CONTROL_REGISTER) in the SRC module. See the System Resource Controller (SRC) chapter for details.

- Enable timestamp synchronization and distribution via the TS_EN field.
- Select a timestamp counter source via the CAN_TS_CNT_SEL field.

  The following table shows the available timestamp sources:

Table 383. Timestamp source selection

CAN_TS_CNT_SEL value	Timestamp source
00b	Lower 32 bits of the GMAC_0 timestamp output
01b	32-bit STM_7 timer output
10b	Lower 32 bits of the PFE GMAC timestamp output
11b	Reserved. Timestamp value is 0000_0000h

51.4 Application information

Perform the procedures in the following sections when using CAN Timestamp functions.

51.4.1 Enable timestamping

To enable CAN Timestamp:

1. Verify that all blocks related to timestamping are out of reset
2. Verify that you have enabled the required clocks
3. Select the timestamp source by setting the desired value in the SRC module's TIME_STAMP_CONTROL_REGISTER[CAN_TS_CNT_SEL] field. See Table 383.
4. Enable the timestamp synchronization submodule by writing 1 to the SRC module's TIME_STAMP_CONTROL_REGISTER[TS_EN] field.
5. Confiigure and enable the source selected in step 3 to start counting.
6. Enable the timestamp sinks (CAN modules) to send out the timestamp. Refer to the LLCE and CAN chapters for details on how to enable for each module.

51.4.2 Disable timestamping or changing the timestamp source

To disable CAN Timestamp:

1. Disable the timestamp sinks (CAN modules). Refer to the LLCE and CAN chapters for details on how to do this.
2. Disable the timestamp synchronization submodule by writing 0 to the SRC module's TIME_STAMP_CONTROL_REGISTER[TS_EN] field. This effectively pauses the timestamp output from the timestamp synchronization submodule by gating the internal clocks and causing the output timestamp flops to hold their most recent value.
3. You can now change the timestamp source, if desired, by setting the desired value in the SRC module's TIME_STAMP_CONTROL_REGISTER[CAN_TS_CNT_SEL] field. See Table 383.
4. To resume timestamping, see Enable timestamping.

NOTE

When pausing or changing timestamp sources, there is a large jump or change in the timestamp values.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c6253c4e1f4a8f123f390d3ab62f1

Chapter 52
SerDes Subsystem

52.1 Chip-specific SerDes information

52.1.1 Mapping of SerDes signals to general purpose registers

For supporting configurability before starting of PCIe_CTRLR (PCIexpress Controller) during run mode, a few SerDes signals are mapped to general purpose registers. For debugging process, a few status signals are mapped to general purpose registers. These registers are a part of SerDes_SS register descriptions section of S32SERDESSUBSYSRM.

52.1.2 SerDes subsystem working modes

This chip implements two SerDes instances:

- SerDes_0 starts at 4040_0000.
- SerDes_1 starts at 4410_0000.

Each row below represents a valid configuration for SerDes.

NOTE
On S32G234M, only PCIe_1 and SerDes_1 are supported.

Table 384. SerDes_0 working modes

Mode	PHY lane 0	PHY lane 1	PHY Ref Clock (MHz)
PCIe only	PCIe0_X2	PCIe0_X2	100
PCIe/SGMII	PCIe0_X1	SGMII 1.25 Gbps (GMAC0 or PFE_MAC2)	100
SGMII only	SGMII 1.25 Gbps (GMAC0)	SGMII 1.25 Gbps (PFE_MAC2)	100/125 [1]

1. The SerDes subsystem's default setting is 125 MHz.

Table 385. SerDes_1 working modes

Mode	PHY lane 0	PHY lane 1	PHY Ref Clock (MHz)
PCIe only	PCIe1_X2	PCIe1_X2	100
PCIe/SGMII	PCIe1_X1	SGMII 1.25 Gbps (PFE_MAC0 or PFE_MAC1)	100
SGMII only	SGMII 1.25 Gbps (PFE_MAC0)	SGMII 1.25 Gbps (PFE_MAC1)	100/125 [1]
SGMII only	SGMII 3.125 Gbps (PFE_MAC0)	Not available	125

1. The SerDes subsystem's default setting is 125 MHz.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a1-8351-390d3ab624c7

52.1.3 Resets required for operation

To accomplish this task...	... do this:
Start a functional reset of the chip	In the MC_RGM, write a 1 to RGM_PRST0_0[PERIPH_5_RST].
Start a reset of the SerDes subsystem	In the MC_RGM, write a 1 to RGM_PRST0_0[PERIPH_4_RST].
Finish the functional reset of the chip	In the MC_RGM, write a 0 to RGM_PRST0_0[PERIPH_5_RST].
Finish the reset of the SerDes subsystem	In the MC_RGM, write a 0 to RGM_PRST0_0[PERIPH_4_RST].

52.1.4 Register configuration for SerDes base spec

SerDes base spec electrical limits for 8Gbps operation requires specific reset values to be set for the PE0_GEN_CTRL_3[LTSSM_EN]. Configure them before the SerDes block is enabled through PE0_GEN_CTRL_3[LTSSM_EN]. These registers are described in the SerDes_SS register descriptions section of S32SERDESSUBSYSRM.

52.1.5 Linking up mode0 and SerDes

Here are the steps for linking mode0 to SerDes:

- Lift the partition reset of SerDes through RGM PRST registers
- Wait for MPLLA to get locked in PCIE_SS_PCIE_PHY_MPLLA_CTRL register
- Set equalization parameters in PORT_LOGIC_GEN3_EQ_CONTROL_OFF
- Program the BARs
- Enable link training in the PCIE_SS_PE0_GEN_CTRL_3 register

52.1.6 Unexpected behavior when enabling error injection in DMA Channel context memories of SerDes

When you enable error injection in DMA channel context memories of SerDes and perform the DMA operations, you get the errors. Even after disabling the injection, error will still be visible on the first access after disabling the error injection because the memories work in read before write fashion. During the error injection enabled stage, there is a write operation in the end that results in the error getting injected in the memory in the end.

52.1.7 Converting nonsecure accesses to secure accesses

This is the procedure to to convert nonsecure accesses to secure accesses on this chip:

1. Write 00b to XRDC.MDA_W0_*n*_DFMT1[SA].
2. Write 1b to XRDC.MDA_W0_*n*_DFMT1[VLD].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a8-1251-390d3a024c1

Chapter 53
SerDes General Purpose Registers (SERDES_GPR)

53.1 PCIe GPRs

53.1.1 Introduction

This module supplements the programming model for the SerDes subsystem.

53.1.2 SERDES_GPR register descriptions

53.1.2.1 SERDES_GPR memory map

SERDES_0_GPR base address: 4007_C500h

Offset	Register	Width (In bits)	Access	Reset value
0h	PCIE Configuration 0 (PCIE_Config_0)	32	W1C	0000_0000h

53.1.2.2 PCIE Configuration 0 (PCIE_Config_0)

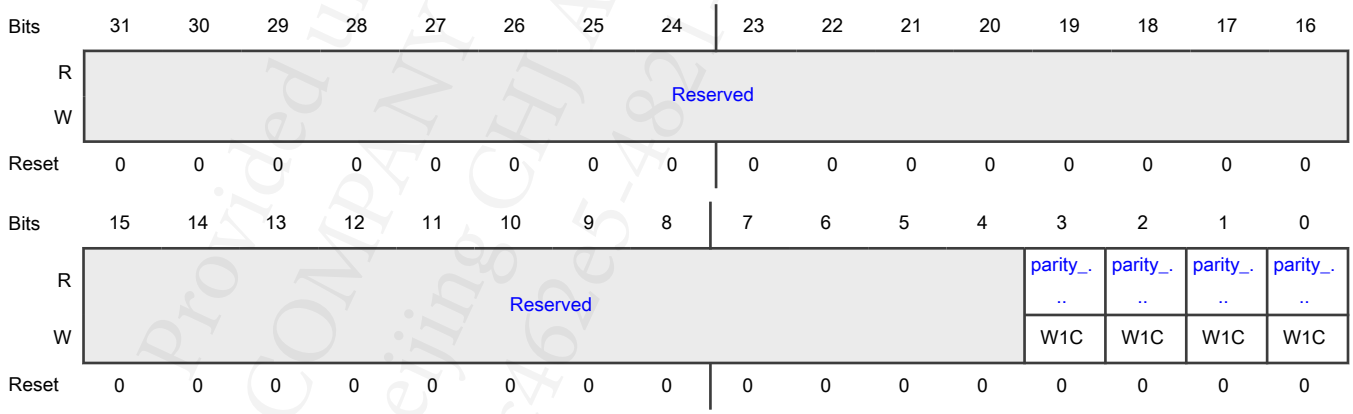
Offset

Register	Offset
PCIE_Config_0	0h

Function

Provides error status of PCIe AXI address or data reads and writes.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482a-48a8-b251-3903ab624c1

Fields

Field	Function
31-4 —	Reserved
3 parity_chk_slv_rdatap_err	parity_chk_slv_rdatap_err Parity Check Error Status Provides the error status from comparing the PCIe controller-generated AXI slave read data bits and the associated parity bits. The error is routed to FCCU. After your application manages the error, write a 1 to this field to change its value to 0. 0b - Parity error did not occur 1b - Parity error occurred
2 parity_chk_mstr_wdatap_err	parity_chk_mstr_wdatap_err Parity Check Error Status Provides the error status from comparing the PCIe controller-generated AXI master write data bits and the associated parity bits. The error is routed to FCCU. After your application manages the error, write a 1 to this field to change its value to 0. 0b - Parity error did not occur 1b - Parity error occurred
1 parity_chk_mstr_awaddrp_err	mstr_awaddrp Parity Check Error Status Provides the error status from comparing the PCIe controller-generated AXI master write address bits and the associated parity bits. The error is routed to FCCU. After your application manages the error, write a 1 to this field to change its value to 0. 0b - Parity error did not occur 1b - Parity error occurred
0 parity_chk_mstr_araddrp_err	mstr_araddrp Parity Check Error Status Provides the error status of the AXI master read address parity check. It is the result of comparing the PCIe controller-generated AXI master read address bits and the associated parity bits. The error is routed to FCCU. After your application manages the error, write a 1 to this field to change its value to 0. 0b - Parity error did not occur 1b - Parity error occurred

53.2 PCIe GPRs

53.2.1 Introduction

This module supplements the programming model for the SerDes subsystem.

53.2.2 SERDES_GPR register descriptions

53.2.2.1 SERDES_GPR memory map

SERDES_1_GPR base address: 4007_CC00h

Offset	Register	Width (In bits)	Access	Reset value
0h	PCIE Configuration 0 (PCIE_Config_0)	32	W1C	0000_0000h

53.2.2.2 PCIE Configuration 0 (PCIE_Config_0)

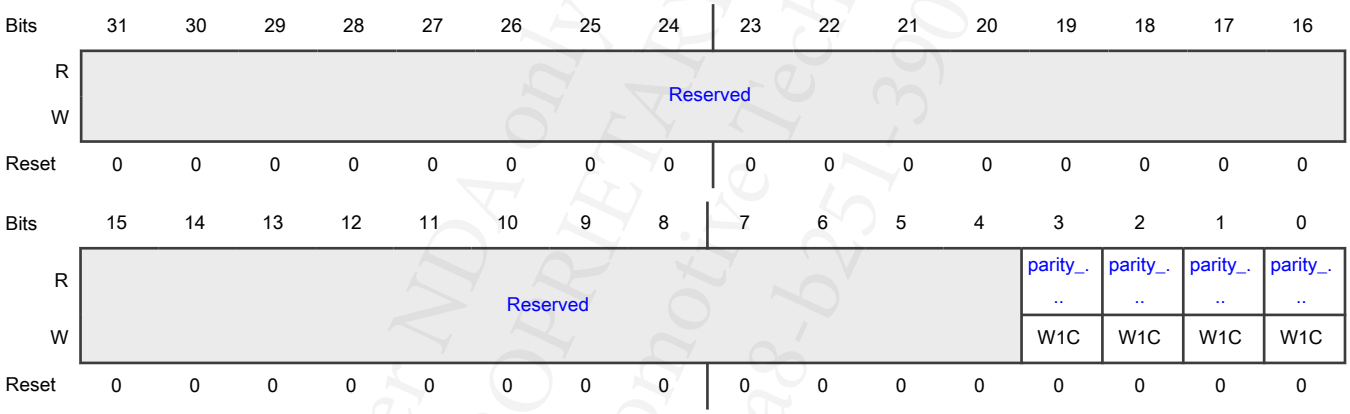
Offset

Register	Offset
PCIE_Config_0	0h

Function

Provides error status of PCIe AXI address or data reads and writes.

Diagram



Fields

Field	Function
31-4 —	Reserved
3 parity_chk_slv_rdatap_err	parity_chk_slv_rdatap_err Parity Check Error Status Provides the error status from comparing the PCIe controller-generated AXI slave read data bits and the associated parity bits. The error is routed to FCCU. After your application manages the error, write a 1 to this field to change its value to 0. 0b - Parity error did not occur 1b - Parity error occurred
2	parity_chk_mstr_wdatap_err Parity Check Error Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
parity_chk_mstr_wdatap_err	Provides the error status from comparing the PCIe controller-generated AXI master write data bits and the associated parity bits. The error is routed to FCCU. After your application manages the error, write a 1 to this field to change its value to 0. 0b - Parity error did not occur 1b - Parity error occurred
1 parity_chk_mstr_awaddrp_err	mstr_awaddrp Parity Check Error Status Provides the error status from comparing the PCIe controller-generated AXI master write address bits and the associated parity bits. The error is routed to FCCU. After your application manages the error, write a 1 to this field to change its value to 0. 0b - Parity error did not occur 1b - Parity error occurred
0 parity_chk_mstr_araddrp_err	mstr_araddrp Parity Check Error Status Provides the error status of the AXI master read address parity check. It is the result of comparing the PCIe controller-generated AXI master read address bits and the associated parity bits. The error is routed to FCCU. After your application manages the error, write a 1 to this field to change its value to 0. 0b - Parity error did not occur 1b - Parity error occurred

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-3[illegible]d3ab6[illegible]4c1

Chapter 54
Low Latency Communication Engine (LLCE)

54.1 Introduction

LLCE is a combination of cores, memory, hardware acceleration IP blocks and firmware developed by NXP.

LLCE supports the following channels:

- 16 BCAN channels
- 2 FlexRay channels (A+B)
- 4 LINFlexD channels
- 4 LPSPI channels

NOTE

This chip does not support the single wire mode for LPSPI0 and LPSPI3.

The communication hardware channels abstract the communication that is fully supported through a firmware-defined host interface. A key advantage of this approach is that it delivers an optimized host interface to reduce the load on the host cores and provides flexibility because additional firmware-defined features are added over the LLCE lifetime. Contact your local NXP representative for further details of firmware extension.

The figure below shows the high-level architecture of LLCE.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390a8b624c1

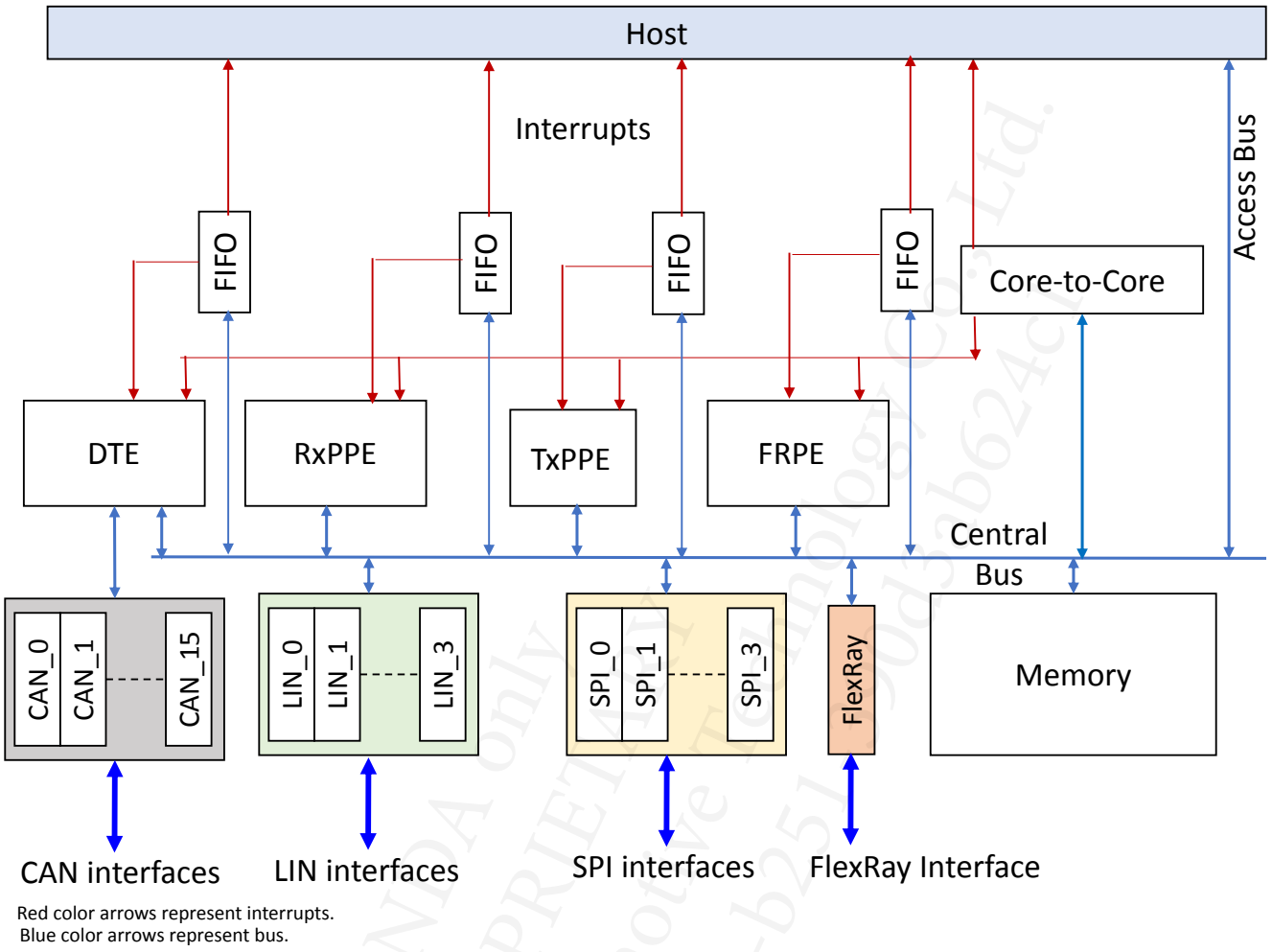


Figure 448. High-level LLCE block diagram

54.2 High-level features

- 16 BCAN interfaces fully compliant with ISO11898-1: 2015
- One FlexRay interface (2 channels) fully compliant with FlexRay communications system protocol specification, Version 2.1 Rev A
- Four LINFlexD interfaces
  - Fully compliant with LIN protocol version 1.3, 2.0, 2.1, and 2.2
  - Support master and slave mode on all channels
- Dedicated path to the HSE_H subsystem to optimize security tasks
- Global timestamping across all BCAN, FlexRay and LIN interfaces, with a global timebase that can be driven by:
  - IEEE802.1AS timebase from PFE
  - IEEE802.1AS timebase from GMAC
  - Free-running 32-bit STM counter
- Advanced BCAN and FlexRay receive filtering
- Interrupt and polled operation by the host core

- Optimized FIFO based host interface
- Abstracts communication with the host cores
- Decouples FlexRay host communication from the FlexRay schedule
- The LLCE cores can access all peripherals on the chip.

54.3 Reset and boot

LLCE is a firmware-based machine. You must load the firmware before using the LLCE. The LLCE cores boot during LLCE initialization by the host. After initialization completes, the LLCE moves into normal mode and is ready for full operation. The delivered host interface driver and LLCE firmware abstracts initialization.

54.4 Modes of operation

After initialization, all interfaces are available and operate at full performance. The abstracted host interface and firmware control communication with LLCE.

LLCE does not support low-power modes. When the chip is in Standby mode, the entire LLCE is power-gated.

54.5 Subsystem control

LLCE has the following system control features that the host configures prior to operation.

Reset control

Each of the LLCE core individually resets via LLCE_SYSRSTR. If you deassert the chip's system functional reset then all LLCE cores default to the reset state. Writing to the respective field in the LLCE_SYSRSTR allows each core to be taken out of reset. See LLCE register descriptions for details.

54.5.1 Reset control

Each of the LLCE core individually resets via LLCE_SYSRSTR. If you deassert the chip's system functional reset then all LLCE cores default to the reset state. Writing to the respective field in the LLCE_SYSRSTR allows each core to be taken out of reset. See LLCE register descriptions for details.

54.6 Host interface

All communication with LLCE is via the host interface and not directly via the individual BCAN, FlexRay, LINFlexD, and SPI communication channels. Data RAM and a number of hardware FIFO IP blocks within LLCE implement the interface. The main functional blocks of the interface are as follows:

- Hardware FIFOs: Used for transmission and reception flows. In general the FIFOs transfer indexes to the message buffers that contain the communication frames (for example, BCAN, FlexRay, and LINFlexD).
- Shared memory: Implemented with SRAM stored in LLCE. The shared memory is mostly used for exchanging message buffer content (communication frames), commands and error or status information between LLCE and the host core.
- Core2Core interrupt module: Used to generate interrupts between LLCE and the host.

The host interface implements a software architecture that transfers indexes between LLCE and the host core via the FIFOs to decrease the number of memory copy operations required for message buffers. The content of the message buffers is stored only once in the shared memory during the entire processing flow. The host core writes to and copies from LLCE shared memory.

The architecture of the host interface is shown in the following figure.

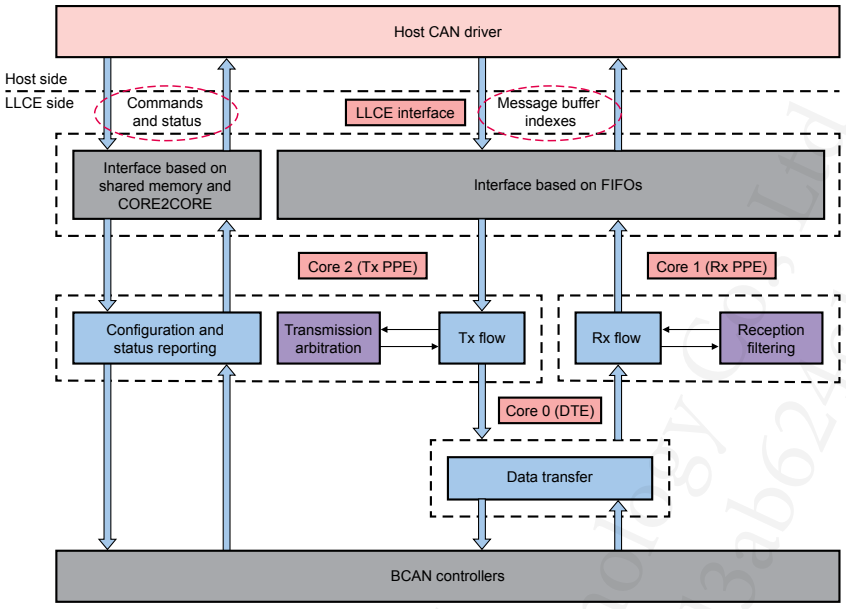


Figure 449. LLCE firmware architecture

The host and LLCE interact via data structures stored in the S32G SRAM. The following data structures exist:

- Command
- Receive message buffer
- Transmit message buffer
- Transmit acknowledge buffer
- Receive object count

The host interface is fully described in the LLCE firmware user guide.

See the LLCE firmware user guide for the following details:

- LLCE host interface
- Booting LLCE
- Initialization of the host interface
- Initialization of communication channels
- Transmission and reception
- LLCE global RAM

54.7 Clocking options

LLCE needs the following clocks to be configured correctly for full operation. See the "Clocking Overview" chapter for details.

Table 386. LLCE clocks

Clock name	Description
LLCE_SYS_CLK	This is the main LLCE clock. All cores and interconnects derive from this clock source. The source is a divided version of the XBAR_CLK that clocks the ARM Cortex-M7 cores.

Table continues on the next page...

Table 386. LLCE clocks (continued)

Clock name	Description
	LLCE_SYS_CLK runs at a maximum frequency of 200 MHz. Clocking LLCE slower than the maximum reduces its functionality.
CAN_PE_CLK[0:15]	This source clocks the protocol engine of the BCAN channels. Its source is selected from the FXOSC or output of the PERPH PLL. Each FlexCAN channel uses the same clock. For example, if you select 40 MHz FXOSC as a source, then it becomes the source for all the channels.
FLEXRAY_PE_CLK	This source clocks the protocol engine of the FlexRay channels. Its source is selected from the FXOSC or output of the PERPH PLL.
LINFLEX_CLK[0:3]	This source clocks the protocol engine of the LINFlexD channels. The LINFLEX_CLK source is LIN_BAUD_CLK, which is selected from the FXOSC or output of the PERPH PLL. To generate LINFLEX_CLK, divide LIN_BAUD_CLK by 2 within LLCE.
LPSPI_CLK[0:3]	This source clocks the LPSPI channels. The LPSPI_CLK source is PER_CLK, which is selected from the FXOSC or output of the PERPH PLL.

54.8 Global timestamping

Frames transmitted and received via the LLCE CAN and FlexRay are timestamped against a global timebase. Selection of the global timebase is configurable. See the Timestamp chapter for global timestamp sources and how they can be selected.

Also see the LLCE host interface and firmware description for supported timestamping of specific channels.

54.9 Host interface events or interrupts

The host core can generate events or interrupts on the individual LLCE cores by writing to the core2core interface. With this action, the host cores generate events or interrupts on LLCE cores 0, 1, 2, and 3. See the host interface documentation for further details.

It is also possible for the LLCE cores 0, 1, 2, and 3 to generate events or interrupts on the chip's host cores. See the interrupt map file attached to this document.

54.10 FIFO interrupts

Each LLCE host interface FIFO generates an event or interrupt on the respective host core. See the interrupt map file attached to this document.

54.11 LLCE generic FIFO

The generic FIFO handles all functionalities needed for different FIFO applications inside LLCE. Specific instances are marked with instance name as defined in LLCE block diagram. All FIFO instances have 16 entries.

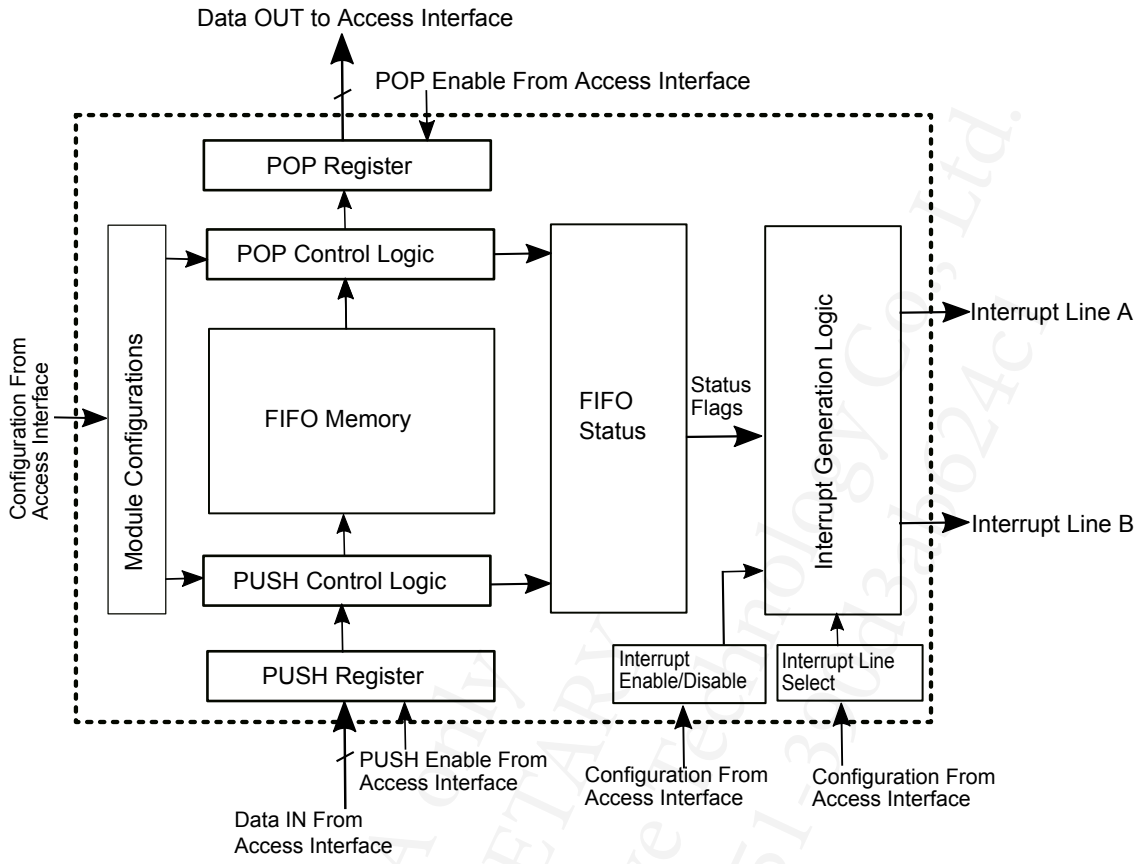


Figure 450. LLCE generic FIFO block diagram

54.11.1 Features

- FIFO full, not-full, empty, and not-empty flags
- FIFO flush control
- Consecutive Entry push operation support (back-to-back clock cycle)
- Consecutive Entry pop operation support (back-to-back clock cycle)
- Configuration of PUSH register Fixed MSB pattern
- Configurable watermark cross flag for number of filled entries
- Configurable watermark cross flag for number of empty entries
- POP operation support when automatic POP-on-read is disabled
- FIFO count status
- Control to overwrite last FIFO entry when FIFO is full
- Maskable event/error flags which can be configured to be routed to any of the two interrupt lines

54.11.2 Modes of operation

LLCE FIFO supports two operation modes:

- **FIFO run mode:** FIFO operates with full functionality.
- **FIFO disabled mode:** FIFO PUSH and POP operations are disabled. Register configurations and flush operation can still work in this mode.

54.11.3 LLCE FIFO register descriptions

54.11.3.1 LLCE FIFO registers memory map

There are multiple FIFOs used for communication between host and LLCE cores. For the base addresses of used FIFOs, see the LLCE firmware user guide.

Offset	Register	Width (In bits)	Access	Reset value
0h	FIFO Configuration Register (FCR)	32	RW	0000_0000h
4h	FIFO Status Register 0 (SR0)	32	W1C	0000_0602h
8h	FIFO Status Register 1 (SR1)	32	W1C	0000_0002h
Ch	FIFO interrupt enable register (IER)	32	RW	0000_0000h
10h	FIFO Interrupt Line Register (ILR)	32	RW	0000_0000h
14h	FIFO push register 0 (PUSHR0)	32	RW	0000_0000h
24h	FIFO POP Register 0 (POPR0)	32	RO	FFFF_FFFFh
34h	FIFO Fixed MSB Register (FMR0)	32	RW	0000_0000h

54.11.3.2 FIFO Configuration Register (FCR)

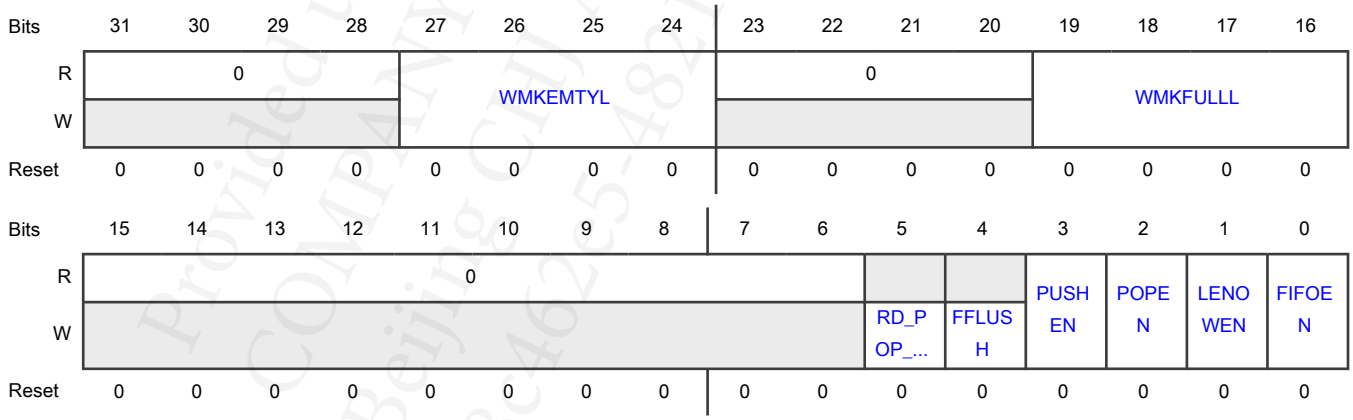
Offset

Register	Offset
FCR	0h

Function

Configure this register to control module level features

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8462e5-4821-48a8-b251-39ddab6041

Fields

Field	Function
31-28 —	Reserved
27-24 WMKEMTYL	Watermark Empty Level Number of entries for FIFO_SR[WMKEFLG] decision. This helps in bulk transfer during write operation.
23-20 —	Reserved
19-16 WMKFULLL	Watermark Full Level Number of entries for FIFO_SR[WMKFFLG] decision. This watermark helps in bulk transfer during read operation.
15-6 —	Reserved
5 RD_POP_DIS	Read when POP Disabled Writing 1 to this bit results in an entry being popped when FIFO_FCR[POPEN] is 0 and FIFO_FCR[FIFOEN] is 1. The FIFO pointers and status are updated. Writing 0 has no effect. When FIFOFCR[POPEN] = 1, writing 1 to this bit has no effect. 0b - No operation 1b - POP operation
4 FFLUSH	FIFO Flush Writing 1 to this flushes the FIFO memory content. It does not clear any set w1c status flag. Writing 0 has no effect. 0b - No operation 1b - FIFO Flush
3 PUSHEN	PUSH enable If enabled, any write to FIFO PUSH register is considered a FIFO PUSH operation and FIFO pointers/ status is updated. If disabled, FIFO logic is unaffected and only the register write takes place. 0b - PUSH Disabled 1b - PUSH Enabled
2 POPEN	POP enable If enabled, any read to FIFO POP register is taken as a FIFO POP operation and the FIFO pointers/ status is updated. If disabled, FIFO logic is unaffected and only the register read takes place. 0b - POP Disabled 1b - POP Enabled
1	Last entry overwrite enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
LENOWEN	Last FIFO entry can be overwritten when FIFO is full, when this bit is enabled. If disabled, overwrite is not allowed. 0b - Overwrite disabled 1b - Overwrite enabled
0 FIFOEN	FIFO enable This field controls the FIFO PUSH and POP operations. Register configurations and flush operation can work for both enabled and disabled values of this field. 0b - FIFO Disabled 1b - FIFO Enabled

54.11.3.3 FIFO Status Register 0 (SR0)

Offset

Register	Offset
SR0	4h

Function

The w1c bits of this register are accessible or reserved based on the programming of corresponding bits in FIFO_ILR. When the corresponding bit in FIFO_ILR is 0 then the bit is accessible to the user for w1c and read operations. When the corresponding bit in FIFO_ILR is 1 then the bit is reserved to the user. The RO bits of this register show the dynamic status of the FIFO independent of FIFO_ILR programming.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			FCOUNT					0						MSBNOMT	PUSHOVR
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POPUND	WMKEMTY	WMKFULL	POPEVT	FNEMTY	FEMTY	FNFULL	FFULL	0						FEMTYD	FFULLD
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESWIN Automotive Technology Co., Ltd.
c8c46b5c-48a2-4a8a-9251-390d3ab624c1

Fields

Field	Function
31-29 —	Reserved
28-24 FCOUNT	FIFO count This field gives the dynamic count of the entries present inside the FIFO.
23-18 —	Reserved
17 MSBNOMT	MSB not matching flag This bit is set when the MSB bit pattern of FIFO_PUSHR does not match the MSB bit pattern programmed inside FIFO_FMR. The bit is cleared by writing 1 into it. 0b - MSB match 1b - MSB mismatch
16 PUSHOVR	PUSH overflow Flag This bit is set when a write access is made to FIFO_PUSHR register when the FIFO is full. The bit is cleared by writing 1 into it. 0b - No PUSH overflow 1b - PUSH overflow
15 POPUND	POP underflow flag This bit is set when a read access is made to FIFO_POPR register when the FIFO is empty. The bit is cleared by writing 1 into it. 0b - No POP underflow 1b - POP underflow
14 WMKEMTY	Watermark empty flag This bit is set when FIFO_SR0[FCOUNT] goes lower than FIFO_FCR[WMKEMTY]. The bit is cleared by writing 1 into it. 0b - No Watermark Empty Event 1b - Watermark Emty Event
13 WMKFULL	Watermark Full Flag This bit is set when FIFO_SR0[FCOUNT] goes higher than FIFO_FCR[WMKFULL]. The bit is cleared by writing 1 into it. 0b - No Watermark Full Event 1b - Watermark Full Event
12	POP event flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
POPEVT	This bit stores the pop event status whenever a POP access is made to FIFO_POPR. The bit is cleared by writing 1 into it. 0b - No POP event 1b - POP event
11 FNEMTY	FIFO not empty flag This bit stores the not empty status of the FIFO when the condition occurs in the hardware. The bit is cleared by writing 1 to it. If the hardware condition that sets this bit is persisting when a w1c operation is happening, the bit is not cleared. 0b - FIFO empty 1b - FIFO not empty
10 FEMTY	FIFO empty This bit stores the empty status of the FIFO when the condition occurs in the hardware. The bit is cleared by writing 1 to it. If the hardware condition that sets this bit is persisting when a w1c operation is happening, the bit is not cleared. 0b - FIFO not empty 1b - FIFO empty
9 FNFULL	FIFO not full flag This bit stores the not full status of the FIFO when the condition occurs in the hardware. The bit is cleared by writing 1 to it. If the hardware condition that sets this bit is persisting when a w1c operation is happening, the bit is not cleared. 0b - FIFO Full 1b - FIFO not Full
8 FFULL	FIFO Full Flag This bit stores the full status of the FIFO when the condition occurs in the hardware. The bit is cleared by writing 1 to it. If the hardware condition that sets this bit is persisting when a w1c operation is happening, the bit is not cleared. 0b - FIFO not Full 1b - FIFO Full
7-2 —	Reserved
1 FEMTYD	FIFO Empy Dynamic This bit gives the dynamic empty status of the FIFO. 0b - FIFO not empty 1b - FIFO empty

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 FFULLD	FIFO Full Dynamic This bit gives the dynamic full status of the FIFO. 0b - FIFO not Full 1b - FIFO Full

54.11.3.4 FIFO Status Register 1 (SR1)

Offset

Register	Offset
SR1	8h

Function

The w1c bits of this register are accessible or reserved based on the programming of corresponding bits in FIFO_ILR. When corresponding bit in FIFO_ILR is 1 then the bit is accessible to the user for w1c operation. When corresponding bit in FIFO_ILR is 0 then the bit is reserved to the user. The RO bits of this register show the dynamic status of the FIFO independent of FIFO_ILR programming.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			FCOUNT					0						MSBNOMT	PUSHOVR
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	POPUND	WMKEMTY	WMKFULL	POPEVT	FNEMTY	FEMTY	FNFULL	FFULL	0						FEMTYD	FFULLD
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Fields

Field	Function
31-29 —	Reserved
28-24	FIFO count

Table continues on the next page...

Table continued from the previous page...

Field	Function
FCOUNT	This bit field gives the dynamic count of the entries present inside the FIFO.
23-18 —	Reserved
17 MSBNOMT	MSB not matching flag This bit is set when the MSB bit pattern of FIFO_PUSHR does not match the MSB bit pattern programmed inside FIFO_FMR. The bit is cleared by writing 1 into it. 0b - MSB match 1b - MSB mismatch
16 PUSHOVR	PUSH overflow flag This bit is set when a write access is made to FIFO_PUSHR register when the FIFO is full. The bit is cleared by writing 1 into it. 0b - No PUSH overflow 1b - PUSH overflow
15 POPUND	POP underflow flag This bit is set when a read access is made to FIFO_POPR register when the FIFO is empty. The bit is cleared by writing 1 into it. 0b - No POP underflow 1b - POP underflow
14 WMKEMTY	Watermark Empty Flag This bit is set when FIFO_SR1[FCOUNT] goes lower than FIFO_FCR[WMKEMTY]. The bit is cleared by writing 1 into it. 0b - No Watermark Empty Event 1b - Watermark Emty Event
13 WMKFULL	Watermark Full Flag This bit is set when FIFO_SR1[FCOUNT] goes higher than FIFO_FCR[WMKFULL]. The bit is cleared by writing 1 into it. 0b - No Watermark Full Event 1b - Watermark Full Event
12 POPEVT	POP event flag This bit stores the pop event status whenever a POP access is made to FIFO_POPR. The bit is cleared by writing 1 into it. 0b - No POP event 1b - POP event

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 FNEMTY	FIFO not empty flag This bit stores the not empty status of the FIFO when the condition occurs in the hardware. The bit is cleared by writing 1 to it. If the hardware condition that sets this bit is persisting when a w1c operation is happening, the bit is not cleared. 0b - FIFO empty 1b - FIFO not empty
10 FEMTY	FIFO empty Flag This bit stores the empty status of the FIFO when the condition occurs in the hardware. The bit is cleared by writing 1 to it. If the hardware condition that sets this bit is persisting when a w1c operation is happening, the bit is not cleared. 0b - FIFO not empty 1b - FIFO empty
9 FNFULL	FIFO not full flag This bit stores the not full status of the FIFO when the condition occurs in the hardware. The bit is cleared by writing 1 to it. If the hardware condition that sets this bit is persisting when a w1c operation is happening, the bit is not cleared. 0b - FIFO Full 1b - FIFO not Full
8 FFULL	FIFO Full Flag This bit stores the full status of the FIFO when the condition occurs in the hardware. The bit is cleared by writing 1 to it. If the hardware condition that sets this bit is persisting when a w1c operation is happening, the bit is not cleared. 0b - FIFO not Full 1b - FIFO Full
7-2 —	Reserved
1 FEMTYD	FIFO Not Empy Dynamic This bit gives the dynamic empty status of the FIFO. 0b - FIFO not empty 1b - FIFO empty
0 FFULLD	FIFO Full Dynamic This bit gives the dynamic full status of the FIFO. 0b - FIFO not Full 1b - FIFO Full

54.11.3.5 FIFO interrupt enable register (IER)

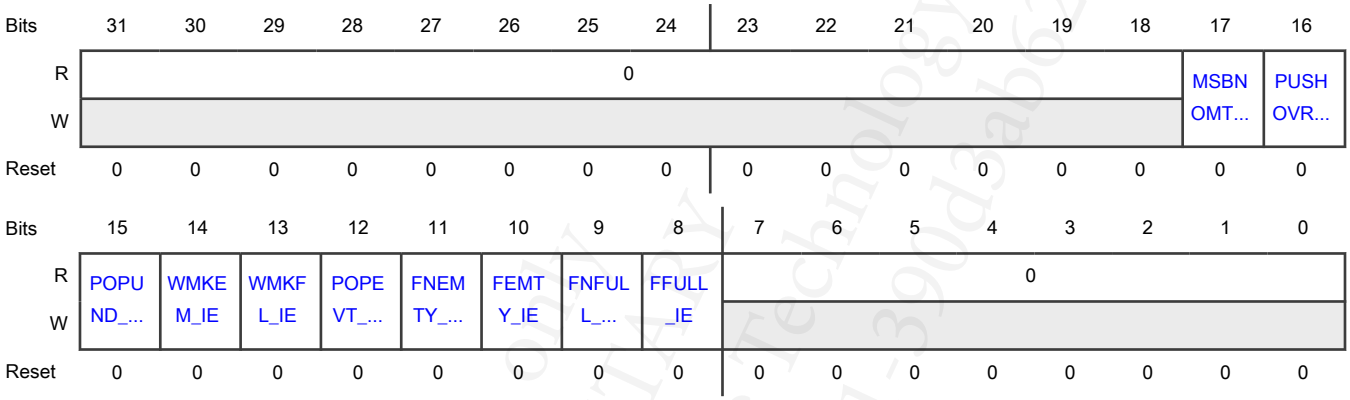
Offset

Register	Offset
IER	Ch

Function

This register is used to enable/disable the propagation of corresponding flags in FIFO_SR0/FIFO_SR1 to the interrupt line

Diagram



Fields

Field	Function
31-18 —	Reserved
17 MSBNOMT_IE	MSB not matching interrupt enable MSB not matching flag propagation to interrupt line can be enabled/disabled by configuring this bit. 0b - Interrupt Disabled 1b - Interrupt Enabled
16 PUSHOVR_IE	PUSH overflow interrupt enable PUSH overflow flag propagation to interrupt line can be enabled/disabled by configuring this bit. 0b - Interrupt Disabled 1b - Interrupt Enabled
15 POPUND_IE	POP underflow interrupt enable POP undeflow flag propagation to interrupt line can be enabled/disabled by configuring this bit. 0b - Interrupt Disabled 1b - Interrupt Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 WMKEM_IE	Watermark empty interrupt enable Watermark empty flag propagation to interrupt line can be enabled/disabled by configuring this bit. 0b - Interrupt Disabled 1b - Interrupt Enabled
13 WMKFL_IE	Watermark full interrupt enable Watermark full flag propagation to interrupt line can be enabled/disabled by configuring this bit. 0b - Interrupt Disabled 1b - Interrupt Enabled
12 POPEVT_IE	POP event interrupt enable POP event flag propagation to interrupt line can be enabled/disabled by configuring this bit. 0b - Interrupt Disabled 1b - Interrupt Enabled
11 FNEMTY_IE	FIFO not empty interrupt enable FIFO not empty flag propagation to interrupt line can be enabled/disabled by configuring this bit. 0b - Interrupt Disabled 1b - Interrupt Enabled
10 FEMTY_IE	FIFO empty interrupt enable FIFO empty flag propagation to interrupt line can be enabled/disabled by configuring this bit. 0b - Interrupt Disabled 1b - Interrupt Enabled
9 FNFULL_IE	FIFO not full interrupt enable FIFO not full flag propagation to interrupt line can be enabled/disabled by configuring this bit. 0b - Interrupt Disabled 1b - Interrupt Enabled
8 FFULL_IE	FIFO Full interrupt enable FIFO full flag propagation to interrupt line can be enabled/disabled by configuring this bit. 0b - Interrupt Disabled 1b - Interrupt Enabled
7-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-b251-390d3ab624c1

54.11.3.6 FIFO Interrupt Line Register (ILR)

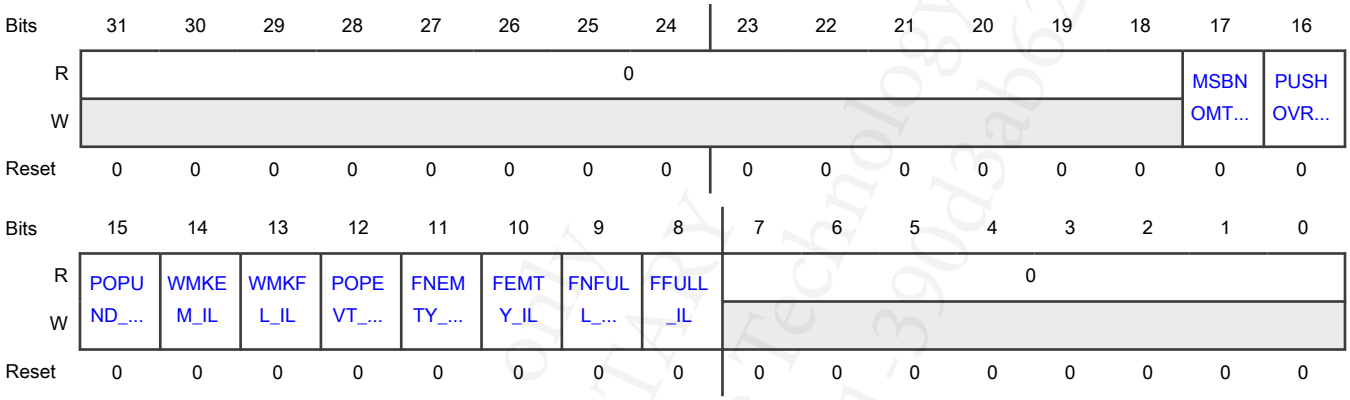
Offset

Register	Offset
ILR	10h

Function

This register selects which interrupt Line, A or B, the individual flag is routed to.

Diagram



Fields

Field	Function
31-18 —	Reserved
17 MSBNOMT_IL	MSB not matching interrupt line This field selects the interrupt line for MSB not matching flag 0b - Flag routed to interrupt line A 1b - Flag routed to interrupt line B
16 PUSHOVR_IL	PUSH overflow interrupt line This field selects the interrupt line for PUSH overflow flag 0b - Flag routed to interrupt line A 1b - Flag routed to interrupt line B
15 POPUND_IL	POP underflow interrupt line This field selects the interrupt line for POP underflow flag 0b - Flag routed to interrupt line A 1b - Flag routed to interrupt line B

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 WMKEM_IL	Watermark empty interrupt line This field selects the interrupt line for watermark empty flag 0b - Flag routed to interrupt line A 1b - Flag routed to interrupt line B
13 WMKFL_IL	Watermark full interrupt line This field selects the interrupt line for watermark full flag 0b - Flag routed to interrupt line A 1b - Flag routed to interrupt line B
12 POPEVT_IL	POP event interrupt line This field selects the interrupt line for POP event flag 0b - Flag routed to interrupt line A 1b - Flag routed to interrupt line B
11 FNEMTY_IL	FIFO not empty interrupt line This field selects the interrupt line for FIFO not empty flag 0b - Flag routed to interrupt line A 1b - Flag routed to interrupt line B
10 FEMTY_IL	FIFO empty interrupt line This field selects the interrupt line for FIFO empty flag 0b - Flag routed to interrupt line A 1b - Flag routed to interrupt line B
9 FNFULL_IL	FIFO not full interrupt line This field selects the interrupt line for FIFO not full flag 0b - Flag routed to interrupt line A 1b - Flag routed to interrupt line B
8 FFULL_IL	FIFO Full interrupt line This field selects the interrupt line for FIFO full flag 0b - Flag routed to interrupt line A 1b - Flag routed to interrupt line B
7-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca4605c-4821-48a8-b251-390d3ab624c1

54.11.3.7 FIFO push register 0 (PUSHR0)

Offset

Register	Offset
PUSHR0	14h

Function
The word entry to be pushed inside the FIFO is written into this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ENTRPUSH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ENTRPUSH															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ENTRPUSH	Entry PUSH The word entry which needs to be added to the FIFO is written into this field.

54.11.3.8 FIFO POP Register 0 (POPR0)

Offset

Register	Offset
POPR0	24h

Function
The FIFO word is read from this register

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-488-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ENTRPOP															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ENTRPOP															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1. This value is read when the FIFO is empty.

Fields

Field	Function
31-0 ENTRPOP	Entry POP This field reads FIFO word to be popped.

54.11.3.9 FIFO Fixed MSB Register (FMR0)

Offset

Register	Offset
FMR0	34h

Function

This register is configured to hold the fixed MSB bit pattern to be compared against the PUSHR MSB bit pattern. If the two bit patterns are not same, MSBNOMT bit is set in the status register and MSB bit pattern of this register is considered correct by the module.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			ENTRFMSB												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e54-8214-48a8-b251-390d3ab614c1

Fields

Field	Function
31-13 —	Reserved
12-0 ENTRFMSB	Entry Fixed MSB The fixed MSB pattern to be compared against the PUSHR MSB bit pattern. **NOTE** To change this value, FIFO needs to be flushed and there must not be any entry populated to avoid having wrong pointer value.

54.11.4 Functional description

The following sections describe the functional details of the LLCE FIFO module.

54.11.4.1 Basic push operation

Once the FIFO module has been initialized (See Programming Guidelines section), and automatic PUSH on write operation has been enabled (FIFO_FCR[PUSHEN] = 1), write into FIFO_PUSHR0 to add the entry to the FIFO memory by polling FIFO not full flag or using the interrupt indication of this flag. Once the PUSH register has been written, the FIFO write pointer increments to point to next location and module status is updated in the status register.

54.11.4.2 Mulit-word push operation

Generic FIFO supports multi-word support to write and read multiple words to and from the FIFO. When multi-word PUSH is being used, always write into PUSH register 0 at the end, after writing into PUSH register of higher index. Otherwise, incomplete data will be loaded into the memory. Once all the writes are complete, one PUSH operation is completed, the write pointer points to the next FIFO location and module status is updated in the status register.

54.11.4.3 Basic pop operation

Once the FIFO module has been initialized (See Programming Guidelines Section), and automatic POP on read operation has been enabled (FIFO_FCR[POPEN]=1), read FIFO_POPR0 to read an entry from FIFO memory by polling FIFO not empty flag or using the interrupt indication of this flag. Once the POP register has been read, the FIFO read pointer increments to point to next location and module status is updated in the status register.

54.11.4.4 Multi-word pop operation

When multi-word POP is being used, always read POP register 0 at the end, after reading POP register of higher index. Otherwise, incorrect data will be read from the FIFO. Once all the reads are complete, one POP operation is completed, the FIFO read pointer increments to point to next location and module status is updated in the status register.

54.11.4.5 Pop operation when FIFO_FCR[POPEN] is 0

There can be use cases when FIFO_FCR[POPEN] is reset to 0 so that a read access on FIFO_POPR does not result in an automatic POP operation. A POP access can still be made by writing to FIFO_FCR[RD_POP_DIS]. This is a write-only-1 bit and any write into this bit when FIFO_FCR[POPEN] = 0 qualifies a POP operation which results in an update of the FIFO status and POP register content. This operation will only work when FIFO_FCR[FIFOEN] = 1.

54.11.4.6 FIFO flush

The FIFO entries can be flushed at any time by writing 1 to FIFO_FCR[FLUSH] to completely empty the FIFO.

NOTE

W1C status flags which are set to 1 are not cleared by FIFO flush operation. They should be cleared by writing 1.

54.11.4.7 FIFO watermark operation

Generic FIFO has separate watermark indicators when the entry count is nearing full and empty levels. The application can configure the watermark levels for both the cases and use the corresponding status flags WMKFULL and WMKEMTY for bulk data transfer.

Example 1

FIFO DEPTH=16

FIFO_FCR[WMKFULLL] = 11

FIFO_SR[WMKFULL] flag is set when FIFO_SR[FCOUNT] goes higher than 11. This flag indicates that at least 12 entries are present inside the FIFO to be read. This can help in bulk read transfer.

Example 2

FIFO DEPTH=16

FIFO_FCR[WMKEMTYL] = 5

FIFO_SR[WMKEMTY] flag is set when FIFO_SR[FCOUNT] goes lower than five. This flag indicates that at least 12 (DEPTH-FIFO_FCR[WMKEMTYL] + 1) entries are vacant inside the FIFO to be written. This can help in bulk write transfer.

54.11.4.8 Last FIFO entry overwrite operation

FIFO_FCR[LENOWEN] when set to 1 allows an entry to be added to the FIFO when it is full, overwriting the last written FIFO entry with this new incoming entry. The PUSH overflow flag, which indicates a FIFO PUSH when FIFO is full, will be set in the status register even when FIFO_FCR[LENOWEN] = 1.

54.11.4.9 Error scenarios

Scenario	Description
PUSH overflow	PUSH overflow error occurs when FIFO_PUSHR0 is written when FIFO is full as indicated by FIFO full flag of status register or FIFO full dynamic bit of status register. PUSH overflow flag is set to 1 in status register when this error condition occurs.
POP underflow	POP underflow error occurs when FIFO_POPR0 is read when FIFO is empty as indicated by FIFO empty flag of status register or FIFO empty dynamic bit of status register. POP underflow flag is set to 1 in status register when this error condition occurs.
PUSH MSB pattern mismatch	The entry pushed into FIFO_PUSHRn can have a fixed MSB pattern. This is true when memory pointers are stored inside the FIFO and not all 32 bits are used to address the memory. The FIFO hardware compares the MSB pattern of every incoming entry against a fixed MSB pattern configured in FIFO_FMRn register. In case the two patterns don't match, MSBNOMT flag is set to 1 in the status register and the MSB pattern of FIFO_FMRn is appended to entries which are read from FIFO_POPRn.

54.11.4.10 FIFO performance

Generic FIFO supports writes into the PUSH register in consecutive clock cycles. Similarly, entries can be read from the POP register in consecutive clock cycles when there are valid entries available.

54.11.5 Interrupts

The generic FIFO has two interrupt outputs, A and B. The status flags inside the status register can be programmed to be routed to either of the two interrupt lines. A value 0 in a bit of FIFO_ILR will route the corresponding flag to interrupt line A. A value 1 in a bit of FIFO_ILR will route the corresponding flag to interrupt line B.

There are two status registers implemented for software to access flags corresponding to each interrupt line with minimum latency. Core receiving interrupt A should access Status register 0 to read/clear status flags which are driving interrupt A. Flags driving interrupt A will be accessible only in FIFO_SR0 and the corresponding flag in FIFO_SR1 will read 0. Core receiving interrupt B should access Status register 1 to read/clear status flags which are driving interrupt B. Flags driving interrupt B will be accessible only in FIFO_SR1 and will read 0 in FIFO_SR0.

Interrupt line selection can be modified, dynamically, in run mode by configuring FIFO_ILR bits.

Example

When FIFO_ILR[POP_EVT] = 1 and FIFO_IER[POP_EVT] = 1, an active POP event flag is routed to interrupt line B.
The core receiving interrupt B should always access FIFO_SR1 to read and clear the status of the FIFO_SR1[POP_EVT].
FIFO_SR0[POP_EVT] will read 0 even if the event has occurred because FIFO_ILR[POP_EVT] = 1.

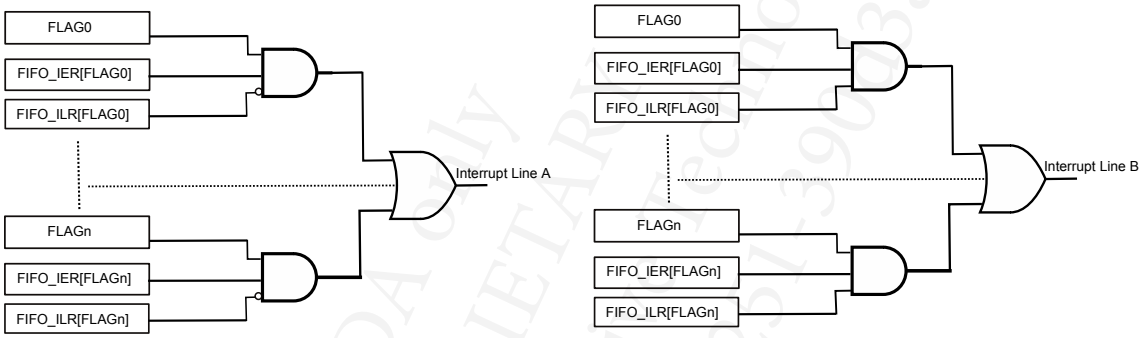


Figure 451. Interrupt generation scheme

54.11.6 Programming

- General guidelines
  1. To write into a vacant entry of the FIFO, use FIFO not full flag of the status register.
  2. To read a valid entry from the FIFO, use FIFO not empty flag of the status register.
  3. For bulk entry transfers, use watermark full and watermark empty flags of the status register See FIFO watermark operation.
- Initialization
  1. Select the interrupt line of each flag of FIFO_SR0/FIFO_SR1 register by writing into corresponding bits of FIFO_ILR register
  2. Enable the required interrupts by setting bits in the FIFO_IER register.
  3. Configure Fixed MSB pattern of FIFO_PUSHR inside FIFO_FMR register. This configuration can be changed only after flushing the FIFO.
  4. Configure the FIFO_FCR register:
     - Automatic Push operation: write 1 to FIFO_FCR[PUSHEN] (must be done for FIFO operation).
     - Automatic POP operation: write 1 to FIFO_FCR[POPEN].
     - FIFO watermark levels FIFO_FCR[WMKFULLL] and FIFO_FCR[WMKEMTYL]
     - Overwrite last entry in case of full: write 1 to FIFO_FCR[LENOWEN].

5. Select FIFO run mode: write 1 to FIFO_FCR[FIFOEN].

- FIFO RUN mode
  1. PUSH operation: Write into FIFO_PUSHR0 to add an entry to the FIFO.
  2. POP operation: Read from FIFO_POPR0 to read an entry from the FIFO.
  3. FIFO Flush: The FIFO can be flushed by writing 1 to FIFO_FCR[FFLUSH] to empty the contents of the FIFO. The flush operation does not clear w1c status flags in SR0/SR1. The status flags still should be cleared by writing 1 into them.
  4. To qualify a POP operation when FIFO_FCR[POPEN] is 0, write 1 to FIFO_FCR[RD_POP_DIS].
  5. Configuration change: The configurations done during initialization can be modified during RUN mode and the change will take effect in the next clock cycle.
  6. Select FIFO disabled mode: write 0 to FIFO_FCR[FIFOEN] to freeze the FIFO. FIFO PUSH and POP operations are disabled. Register configurations and flush operation can still work in this mode.

54.12 Core-to-Core module

Core to core communication module provides LLCE cores and HOST core the capability to indicate system events among themselves via flags and associated mask-able interrupts. Exclusive registers are provided for each core for flag bits and interrupt masking. A flag bit for a receiving core can be set by a transmitting core to indicate an event to it. Once set, the flag can be cleared by the receiving core only.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390...ab624...

54.12.1 Block diagram

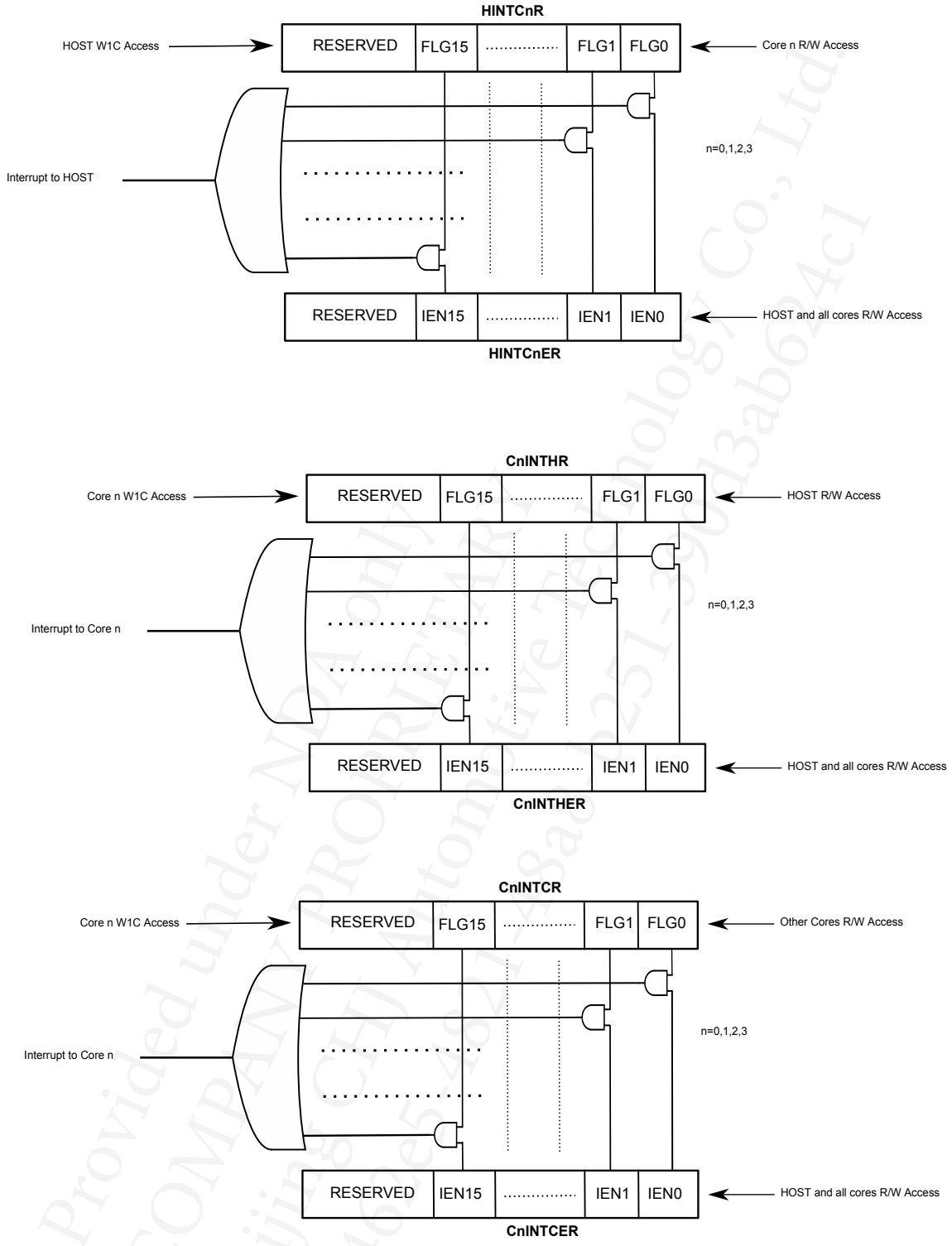


Figure 452. Core-to-core block diagram

54.12.2 Features

- Support of event communication between four LLCE cores and one host

- Interrupt to host from LLCE cores
  - Support of 16 flags with masks from each LLCE core
  - One interrupt line from each LLCE core (16 flags OR-ed after mask)
- Interrupt to LLCE cores from host
  - Support of 16 flags with masks from Host to each LLCE core
  - One interrupt line for each LLCE core (16 flags OR-ed after mask)

54.12.3 Core-to-core register descriptions

Throughout the register description the following convention is to be followed:

- Core 0 is DTE.
- Core 1 is RXPPE.
- Core 2 is TXPPE.
- Core 3 is FRPE.

The host is the core accessing through the host AHB port of LLCE.

54.12.3.1 Core_to_core memory map

LLCE_Core_to_core base address: 43C2_6000h

Offset	Register	Width (In bits)	Access	Reset value
0h	HOST interrupt from Core 0 Register (HINTC0R)	32	RW	0000_0000h
4h	HOST Interrupt from Core 1 Register (HINTC1R)	32	RW	0000_0000h
8h	HOST Interrupt from Core 2 Register (HINTC2R)	32	RW	0000_0000h
Ch	HOST Interrupt from Core 3 Register (HINTC3R)	32	RW	0000_0000h
20h	Core 0 Interrupt from HOST (C0INTHR)	32	RW	0000_0000h
24h	Core 1 Interrupt From HOST Register (C1INTHR)	32	RW	0000_0000h
28h	Core 2 Interrupt from HOST Register (C2INTHR)	32	RW	0000_0000h
2Ch	Core 3 Interrupt from HOST Register (C3INTHR)	32	RW	0000_0000h
60h	HOST Interrupt from Core 0 Enable Register (HINTC0ER)	32	RW	0000_0000h
64h	HOST Interrupt from Core 1 Enable Register (HINTC1ER)	32	RW	0000_0000h
68h	HOST Interrupt from Core 2 Enable Register (HINTC2ER)	32	RW	0000_0000h
6Ch	HOST Interrupt from Core 3 Enable Register (HINTC3ER)	32	RW	0000_0000h
80h	Core 0 Interrupt from Host Enable Register (C0INTHER)	32	RW	0000_0000h
84h	Core 1 Interrupt from HOST Enable Register (C1INTHER)	32	RW	0000_0000h
88h	Core 2 Interrupt from HOST Enable Register (C2INTHER)	32	RW	0000_0000h
8Ch	Core 3 Interrupt from HOST Enable Register (C3INTHER)	32	RW	0000_0000h

NXP Confidential Proprietary Information - For NXP and Authorized Parties Only
Freescale Semiconductor, Inc. 2015-2016. All rights reserved.
Huawei Technologies Co., Ltd.
Huawei Confidential
Huawei Proprietary
Copyright © 2018-2020
Version: 1.0
Date: 2020-08-28
ID: 5f4a8a8-b251-390d3ab624c1

54.12.3.2 HOST interrupt from Core 0 Register (HINTC0R)

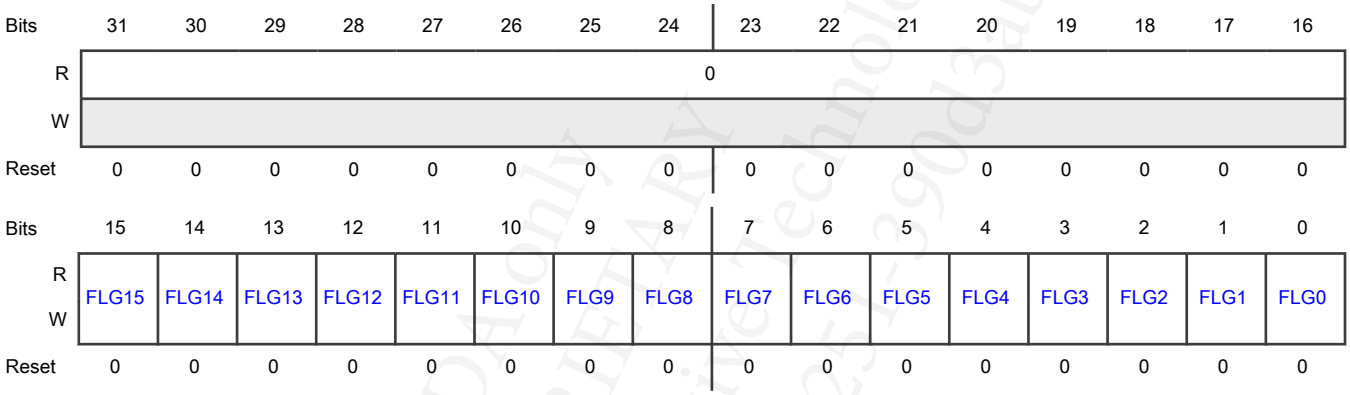
Offset

Register	Offset
HINTC0R	0h

Function

This register controls the event indication to HOST from Core 0. 16 flags can be set by Core 0 to indicate 16 separate events. All flags combine to generate a single interrupt to HOST. A flag once set, can be cleared only by HOST with w1c operation. All cores and HOST have read access to this register. Only HOST and Core 0 have write access to this register. ACCESS GUIDELINE- HOST: Write 1 to clear flag. Core 0: Write 1 to set flag.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 FLG15	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
14 FLG14	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
13 FLG13	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Event
12 FLG12	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
11 FLG11	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
10 FLG10	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
9 FLG9	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
8 FLG8	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
7 FLG7	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
6 FLG6	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
5 FLG5	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
4	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLG4	0b - No Event 1b - Event
3 FLG3	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
2 FLG2	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
1 FLG1	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
0 FLG0	This flag is set by Core 0 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event

54.12.3.3 HOST Interrupt from Core 1 Register (HINTC1R)

Offset

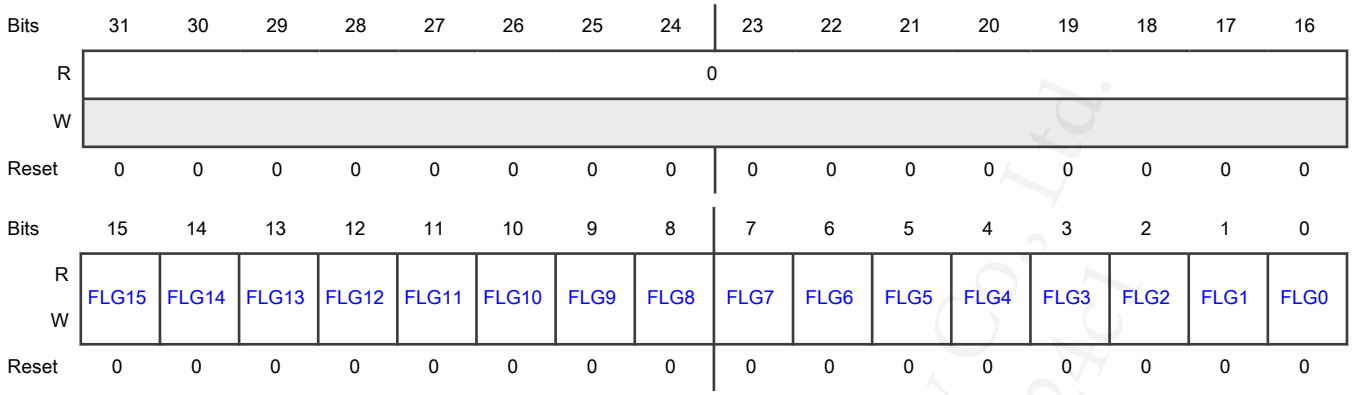
Register	Offset
HINTC1R	4h

Function

This register controls the event indication to HOST from Core 1. 16 flags can be set by Core 1 to indicate 16 separate events. All flags combine to generate a single interrupt to HOST. A flag once set, can be cleared only by HOST with w1c operation. All cores and HOST have read access to this register. Only HOST and Core 1 have write access to this register. ACCESS GUIDELINE- HOST: Write 1 to clear flag. Core 1: Write 1 to set flag.

Beijing HJ Automotive Technology Co., Ltd.
CONFIDENTIAL NDA only
Proprietary under NDA only
c8c462e0-821-48a8-b251-390daab624c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15 FLG15	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
14 FLG14	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
13 FLG13	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
12 FLG12	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
11 FLG11	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
10 FLG10	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No Event 1b - Event
9 FLG9	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
8 FLG8	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
7 FLG7	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
6 FLG6	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
5 FLG5	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
4 FLG4	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
3 FLG3	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
2 FLG2	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 FLG1	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
0 FLG0	This flag is set by Core 1 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event

54.12.3.4 HOST Interrupt from Core 2 Register (HINTC2R)

Offset

Register	Offset
HINTC2R	8h

Function

This register controls the event indication to HOST from Core 2. 16 flags can be set by Core 2 to indicate 16 separate events. All flags combine to generate a single interrupt to HOST. A flag once set, can be cleared only by HOST with w1c operation. All cores and HOST have read access to this register. Only HOST and Core 2 have write access to this register. ACCESS GUIDELINE- HOST: Write 1 to clear flag. Core 2: Write 1 to set flag.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FLG15	FLG14	FLG13	FLG12	FLG11	FLG10	FLG9	FLG8	FLG7	FLG6	FLG5	FLG4	FLG3	FLG2	FLG1	FLG0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 FLG15	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
14 FLG14	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
13 FLG13	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
12 FLG12	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
11 FLG11	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
10 FLG10	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
9 FLG9	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
8 FLG8	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
7	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLG7	0b - No Event 1b - Event
6 FLG6	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
5 FLG5	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
4 FLG4	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
3 FLG3	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
2 FLG2	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
1 FLG1	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
0 FLG0	This flag is set by Core 2 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-1821-48a8-9251-890aab624c1

54.12.3.5 HOST Interrupt from Core 3 Register (HINTC3R)

Offset

Register	Offset
HINTC3R	Ch

Function

This register controls the event indication to HOST from Core 3. 16 flags can be set by Core 3 to indicate 16 separate events. All flags combine to generate a single interrupt to HOST. A flag once set, can be cleared only by HOST with w1c operation. All cores and HOST have read access to this register. Only HOST and Core 3 have write access to this register. ACCESS GUIDELINE- HOST: Write 1 to clear flag. Core 3: Write 1 to set flag.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FLG15	FLG14	FLG13	FLG12	FLG11	FLG10	FLG9	FLG8	FLG7	FLG6	FLG5	FLG4	FLG3	FLG2	FLG1	FLG0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 FLG15	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
14 FLG14	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
13 FLG13	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Event
12 FLG12	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
11 FLG11	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
10 FLG10	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
9 FLG9	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
8 FLG8	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
7 FLG7	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
6 FLG6	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
5 FLG5	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
4	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLG4	0b - No Event 1b - Event
3 FLG3	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
2 FLG2	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
1 FLG1	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event
0 FLG0	This flag is set by Core 3 by writing 1 into it to indicate an event. The flag is cleared by HOST with w1c operation. 0b - No Event 1b - Event

54.12.3.6 Core 0 Interrupt from HOST (C0INTHR)

Offset

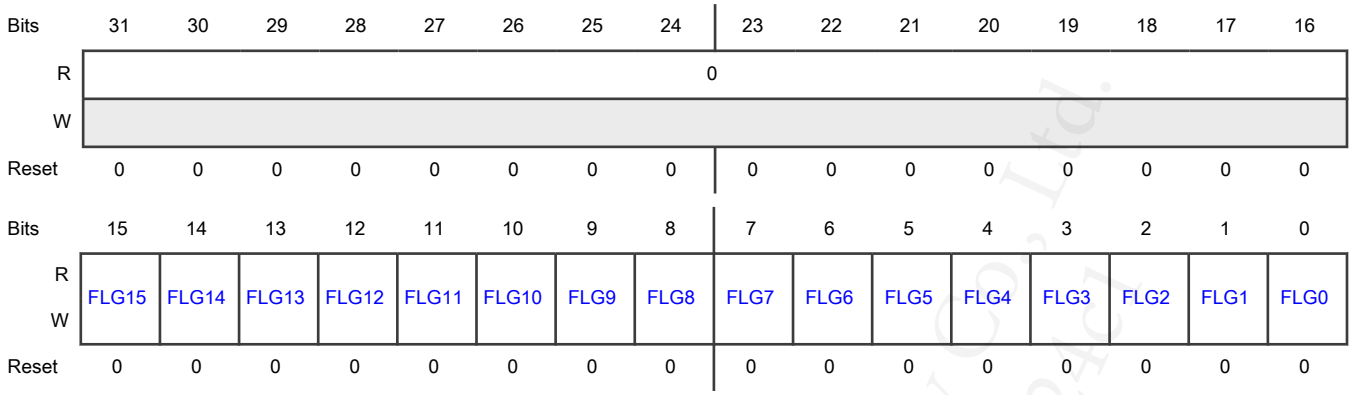
Register	Offset
C0INTHR	20h

Function

This register controls the event indication to Core 0 from HOST. 16 flags can be set by HOST to indicate 16 separate events. All flags combine to generate a single interrupt to Core 0. A flag once set, can be cleared only by Core 0 with w1c operation. All cores and HOST have read access to this register. Only HOST and Core 0 have write access to this register. ACCESS GUIDELINE- HOST: Write 1 to set flag. Core 0: Write 1 to clear flag.

Provided under NDA only
CONFIDENTIAL PROPRIETARY
Beijing ECHJ Automotive Technology Co., Ltd.
c8c462e8-821f-48a8-b251-390daab624c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15 FLG15	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event
14 FLG14	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event
13 FLG13	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event
12 FLG12	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event
11 FLG11	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event
10 FLG10	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No Event 1b - Event
9 FLG9	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event
8 FLG8	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event
7 FLG7	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event
6 FLG6	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event
5 FLG5	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event
4 FLG4	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event
3 FLG3	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event
2 FLG2	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 FLG1	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event
0 FLG0	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 0 with w1c operation. 0b - No Event 1b - Event

54.12.3.7 Core 1 Interrupt From HOST Register (C1INTHR)

Offset

Register	Offset
C1INTHR	24h

Function

This register controls the event indication to Core 1 from HOST. 16 flags can be set by HOST to indicate 16 separate events. All flags combine to generate a single interrupt to Core 1. A flag once set, can be cleared only by Core 1 with w1c operation. All cores and HOST have read access to this register. Only HOST and Core 1 have write access to this register. ACCESS GUIDELINE- HOST: Write 1 to set flag. Core 1: Write 1 to clear flag.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FLG15	FLG14	FLG13	FLG12	FLG11	FLG10	FLG9	FLG8	FLG7	FLG6	FLG5	FLG4	FLG3	FLG2	FLG1	FLG0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
15 FLG15	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
14 FLG14	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
13 FLG13	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
12 FLG12	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
11 FLG11	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
10 FLG10	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
9 FLG9	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
8 FLG8	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
7	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLG7	0b - No Event 1b - Event
6 FLG6	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
5 FLG5	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
4 FLG4	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
3 FLG3	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
2 FLG2	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
1 FLG1	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event
0 FLG0	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 1 with w1c operation. 0b - No Event 1b - Event

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-821-18a8-9251-890daab624c1

54.12.3.8 Core 2 Interrupt from HOST Register (C2INTHR)

Offset

Register	Offset
C2INTHR	28h

Function

This register controls the event indication to Core 2 from HOST. 16 flags can be set by HOST to indicate 16 separate events. All flags combine to generate a single interrupt to Core 2. A flag once set, can be cleared only by Core 2 with w1c operation. All cores and HOST have read access to this register. Only HOST and Core 2 have write access to this register. ACCESS GUIDELINE- HOST: Write 1 to set flag. Core 2: Write 1 to clear flag.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FLG15	FLG14	FLG13	FLG12	FLG11	FLG10	FLG9	FLG8	FLG7	FLG6	FLG5	FLG4	FLG3	FLG2	FLG1	FLG0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 FLG15	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event
14 FLG14	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event
13 FLG13	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Event
12 FLG12	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event
11 FLG11	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event
10 FLG10	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event
9 FLG9	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event
8 FLG8	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event
7 FLG7	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event
6 FLG6	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event
5 FLG5	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event
4	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLG4	0b - No Event 1b - Event
3 FLG3	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event
2 FLG2	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event
1 FLG1	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event
0 FLG0	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 2 with w1c operation. 0b - No Event 1b - Event

54.12.3.9 Core 3 Interrupt from HOST Register (C3INTHR)

Offset

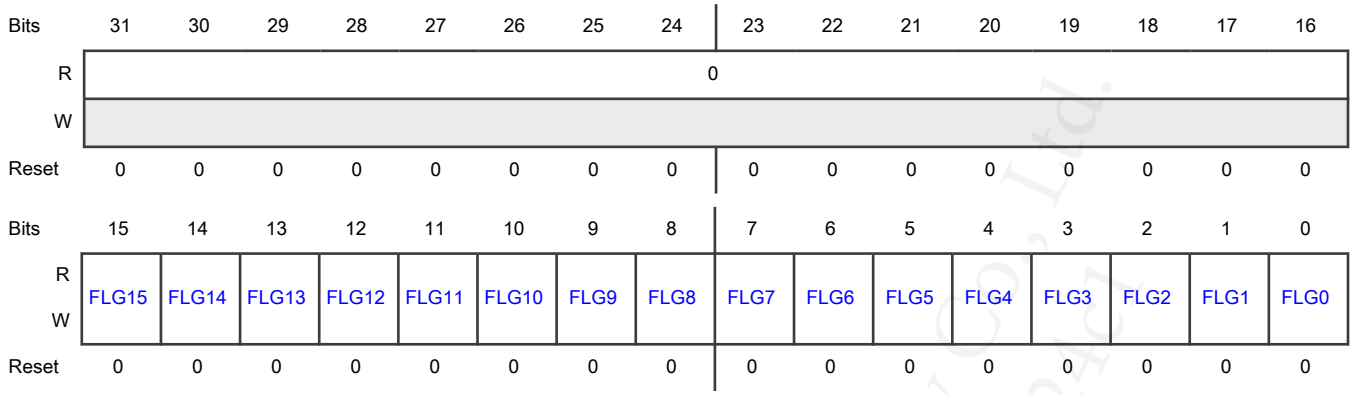
Register	Offset
C3INTHR	2Ch

Function

This register controls the event indication to Core 3 from HOST. 16 flags can be set by HOST to indicate 16 separate events. All flags combine to generate a single interrupt to Core 3. A flag once set, can be cleared only by Core 3 with w1c operation. All cores and HOST have read access to this register. Only HOST and Core 3 have write access to this register. ACCESS GUIDELINE- HOST: Write 1 to set flag. Core 3: Write 1 to clear flag .

Provided under NDA only
CONFIDENTIAL PROPRIETARY
Beijing ECHJ Automotive Technology Co., Ltd.
c8c462e8-821f-48a8-b251-390daab624c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15 FLG15	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event
14 FLG14	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event
13 FLG13	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event
12 FLG12	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event
11 FLG11	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event
10 FLG10	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No Event 1b - Event
9 FLG9	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event
8 FLG8	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event
7 FLG7	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event
6 FLG6	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event
5 FLG5	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event
4 FLG4	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event
3 FLG3	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event
2 FLG2	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 FLG1	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event
0 FLG0	This flag is set by HOST by writing 1 into it to indicate an event. The flag is cleared by Core 3 with w1c operation. 0b - No Event 1b - Event

54.12.3.10 HOST Interrupt from Core 0 Enable Register (HINTC0ER)

Offset

Register	Offset
HINTC0ER	60h

Function

This register controls the masking of corresponding flags inside HINTC0R. ACCESS GUIDELINE- Write/read access for all cores and HOST.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IEN15	IEN14	IEN13	IEN12	IEN11	IEN10	IEN9	IEN8	IEN7	IEN6	IEN5	IEN4	IEN3	IEN2	IEN1	IEN0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEC Automotive Technology Co., Ltd.
8c462e5-4621-48a8-b231-390d3ab624c

Table continued from the previous page...

Field	Function
15 IEN15	This bit enables/disables FLG15 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
14 IEN14	This bit enables/disables FLG14 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
13 IEN13	This bit enables/disables FLG13 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
12 IEN12	This bit enables/disables FLG12 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
11 IEN11	This bit enables/disables FLG11 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
10 IEN10	This bit enables/disables FLG10 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
9 IEN9	This bit enables/disables FLG9 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
8 IEN8	This bit enables/disables FLG8 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
7 IEN7	This bit enables/disables FLG7 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
6 IEN6	This bit enables/disables FLG6 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
5 IEN5	This bit enables/disables FLG5 propagation to the interrupt line

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Flag disabled 1b - Flag enabled
4 IEN4	This bit enables/disables FLG4 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
3 IEN3	This bit enables/disables FLG3 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
2 IEN2	This bit enables/disables FLG2 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
1 IEN1	This bit enables/disables FLG1 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
0 IEN0	This bit enables/disables FLG0 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled

54.12.3.11 HOST Interrupt from Core 1 Enable Register (HINTC1ER)

Offset

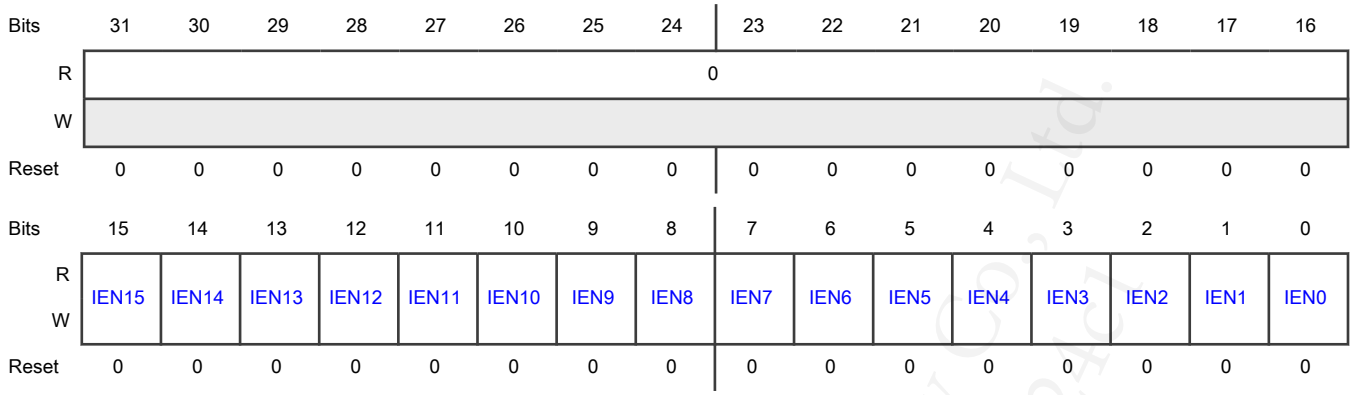
Register	Offset
HINTC1ER	64h

Function

This register controls the masking of corresponding flags inside HINTC1R. ACCESS GUIDELINE- Write/read access for all cores and HOST.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a2-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15 IEN15	This bit enables/disables FLG15 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
14 IEN14	This bit enables/disables FLG14 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
13 IEN13	This bit enables/disables FLG13 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
12 IEN12	This bit enables/disables FLG12 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
11 IEN11	This bit enables/disables FLG11 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
10 IEN10	This bit enables/disables FLG10 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
9 IEN9	This bit enables/disables FLG9 propagation to the interrupt line

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Flag disabled 1b - Flag enabled
8 IEN8	This bit enables/disables FLG8 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
7 IEN7	This bit enables/disables FLG7 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
6 IEN6	This bit enables/disables FLG6 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
5 IEN5	This bit enables/disables FLG5 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
4 IEN4	This bit enables/disables FLG4 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
3 IEN3	This bit enables/disables FLG3 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
2 IEN2	This bit enables/disables FLG2 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
1 IEN1	This bit enables/disables FLG1 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
0 IEN0	This bit enables/disables FLG0 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled

Provided by IHS Markit
COMPANY CONFIDENTIAL
Beijing CRI Automotive Technology Co., Ltd.
c8c46265-4a21-48a8-b251-390d3ab624c1

54.12.3.12 HOST Interrupt from Core 2 Enable Register (HINTC2ER)

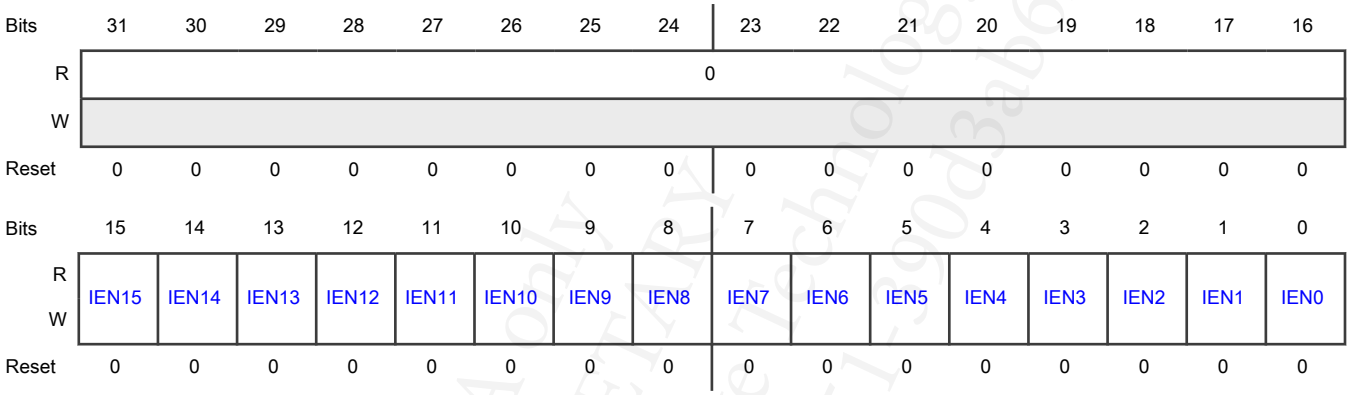
Offset

Register	Offset
HINTC2ER	68h

Function

This register controls the masking of corresponding flags inside HINTC2R. ACCESS GUIDELINE- Write/read access for all cores and HOST.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 IEN15	This bit enables/disables FLG15 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
14 IEN14	This bit enables/disables FLG14 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
13 IEN13	This bit enables/disables FLG13 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
12 IEN12	This bit enables/disables FLG12 propagation to the interrupt line 0b - Flag disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Flag enabled
11 IEN11	This bit enables/disables FLG11 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
10 IEN10	This bit enables/disables FLG10 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
9 IEN9	This bit enables/disables FLG9 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
8 IEN8	This bit enables/disables FLG8 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
7 IEN7	This bit enables/disables FLG7 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
6 IEN6	This bit enables/disables FLG6 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
5 IEN5	This bit enables/disables FLG5 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
4 IEN4	This bit enables/disables FLG4 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
3 IEN3	This bit enables/disables FLG3 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
2 IEN2	This bit enables/disables FLG2 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 IEN1	This bit enables/disables FLG1 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
0 IEN0	This bit enables/disables FLG0 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled

54.12.3.13 HOST Interrupt from Core 3 Enable Register (HINTC3ER)

Offset

Register	Offset
HINTC3ER	6Ch

Function

This register controls the masking of corresponding flags inside HINTC3R. ACCESS GUIDELINE- Write/read access for all cores and HOST.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IEN15	IEN14	IEN13	IEN12	IEN11	IEN10	IEN9	IEN8	IEN7	IEN6	IEN5	IEN4	IEN3	IEN2	IEN1	IEN0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 IEN15	This bit enables/disables FLG15 propagation to the interrupt line

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Flag disabled 1b - Flag enabled
14 IEN14	This bit enables/disables FLG14 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
13 IEN13	This bit enables/disables FLG13 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
12 IEN12	This bit enables/disables FLG12 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
11 IEN11	This bit enables/disables FLG11 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
10 IEN10	This bit enables/disables FLG10 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
9 IEN9	This bit enables/disables FLG9 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
8 IEN8	This bit enables/disables FLG8 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
7 IEN7	This bit enables/disables FLG7 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
6 IEN6	This bit enables/disables FLG6 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
5 IEN5	This bit enables/disables FLG5 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 IEN4	This bit enables/disables FLG4 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
3 IEN3	This bit enables/disables FLG3 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
2 IEN2	This bit enables/disables FLG2 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
1 IEN1	This bit enables/disables FLG1 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
0 IEN0	This bit enables/disables FLG0 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled

54.12.3.14 Core 0 Interrupt from Host Enable Register (C0INTHER)

Offset

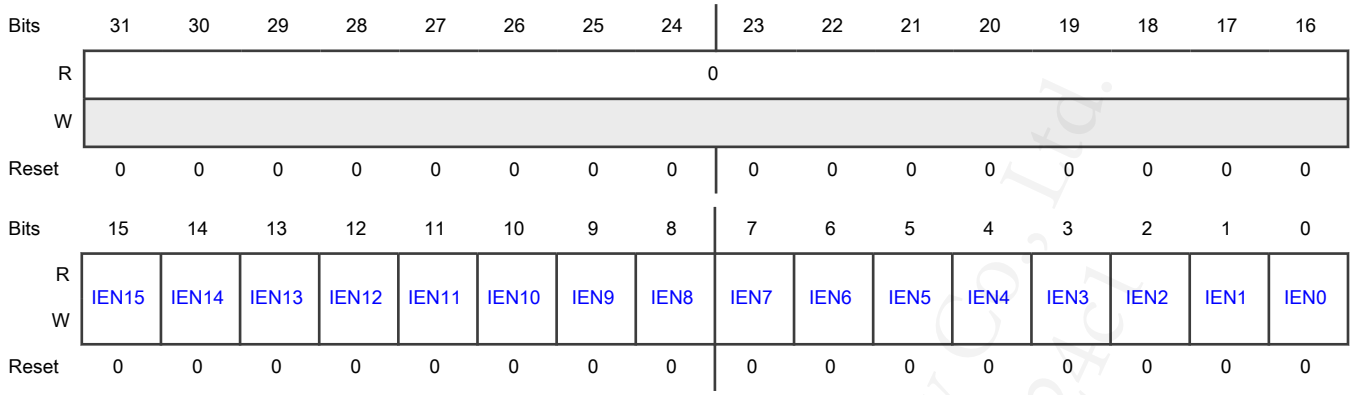
Register	Offset
C0INTHER	80h

Function

This register controls the masking of corresponding flags inside C0INTHR. ACCESS GUIDELINE- Write/read access for all cores and HOST.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15 IEN15	This bit enables/disables FLG15 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
14 IEN14	This bit enables/disables FLG14 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
13 IEN13	This bit enables/disables FLG13 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
12 IEN12	This bit enables/disables FLG12 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
11 IEN11	This bit enables/disables FLG11 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
10 IEN10	This bit enables/disables FLG10 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
9 IEN9	This bit enables/disables FLG9 propagation to the interrupt line

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Flag disabled 1b - Flag enabled
8 IEN8	This bit enables/disables FLG8 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
7 IEN7	This bit enables/disables FLG7 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
6 IEN6	This bit enables/disables FLG6 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
5 IEN5	This bit enables/disables FLG5 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
4 IEN4	This bit enables/disables FLG4 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
3 IEN3	This bit enables/disables FLG3 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
2 IEN2	This bit enables/disables FLG2 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
1 IEN1	This bit enables/disables FLG1 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
0 IEN0	This bit enables/disables FLG0 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled

Provided by IHS Markit
COMPANY CONFIDENTIAL
Beijing CPU Automotive Technology Co., Ltd.
c8c46265-4a21-48a8-b251-390d3ab624c1

54.12.3.15 Core 1 Interrupt from HOST Enable Register (C1INTHER)

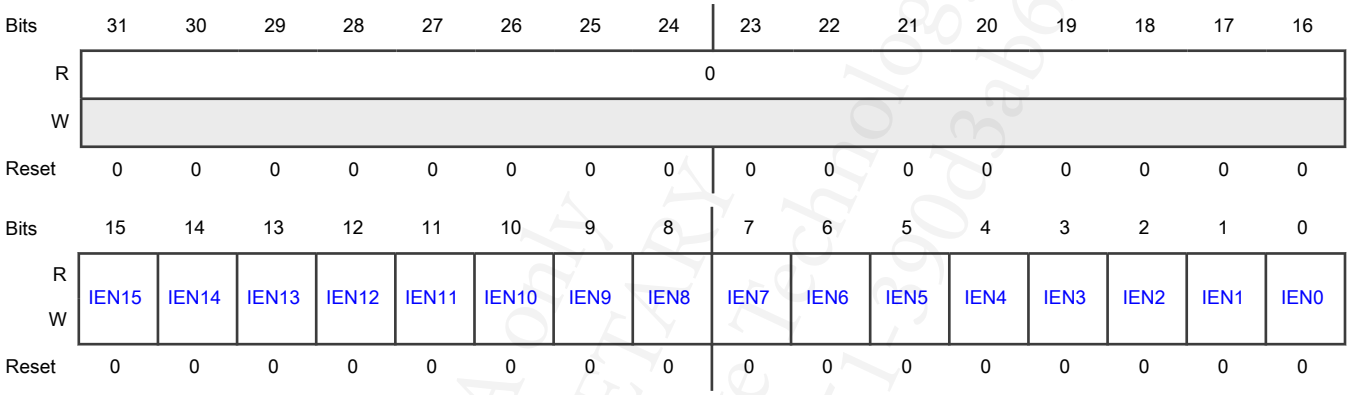
Offset

Register	Offset
C1INTHER	84h

Function

This register controls the masking of corresponding flags inside C1INTHR. ACCESS GUIDELINE- Write/read access for all cores and HOST.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 IEN15	This bit enables/disables FLG15 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
14 IEN14	This bit enables/disables FLG14npropagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
13 IEN13	This bit enables/disables FLG13 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
12 IEN12	This bit enables/disables FLG12 propagation to the interrupt line 0b - Flag disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Flag enabled
11 IEN11	This bit enables/disables FLG11 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
10 IEN10	This bit enables/disables FLG10 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
9 IEN9	This bit enables/disables FLG9 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
8 IEN8	This bit enables/disables FLG8 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
7 IEN7	This bit enables/disables FLG7 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
6 IEN6	This bit enables/disables FLG6 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
5 IEN5	This bit enables/disables FLG5 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
4 IEN4	This bit enables/disables FLG4 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
3 IEN3	This bit enables/disables FLG3 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
2 IEN2	This bit enables/disables FLG2 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 IEN1	This bit enables/disables FLG1 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
0 IEN0	This bit enables/disables FLG0 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled

54.12.3.16 Core 2 Interrupt from HOST Enable Register (C2INTHER)

Offset

Register	Offset
C2INTHER	88h

Function

This register controls the masking of corresponding flags inside C2INTHR. ACCESS GUIDELINE- Write/read access for all cores and HOST.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IEN15	IEN14	IEN13	IEN12	IEN11	IEN10	IEN9	IEN8	IEN7	IEN6	IEN5	IEN4	IEN3	IEN2	IEN1	IEN0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15 IEN15	This bit enables/disables FLG15 propagation to the interrupt line

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Flag disabled 1b - Flag enabled
14 IEN14	This bit enables/disables FLG14 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
13 IEN13	This bit enables/disables FLG13 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
12 IEN12	This bit enables/disables FLG12 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
11 IEN11	This bit enables/disables FLG11 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
10 IEN10	This bit enables/disables FLG10 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
9 IEN9	This bit enables/disables FLG9 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
8 IEN8	This bit enables/disables FLG8 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
7 IEN7	This bit enables/disables FLG7 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
6 IEN6	This bit enables/disables FLG6 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
5 IEN5	This bit enables/disables FLG5 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 IEN4	This bit enables/disables FLG4 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
3 IEN3	This bit enables/disables FLG3 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
2 IEN2	This bit enables/disables FLG2 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
1 IEN1	This bit enables/disables FLG1 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
0 IEN0	This bit enables/disables FLG0 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled

54.12.3.17 Core 3 Interrupt from HOST Enable Register (C3INTHER)

Offset

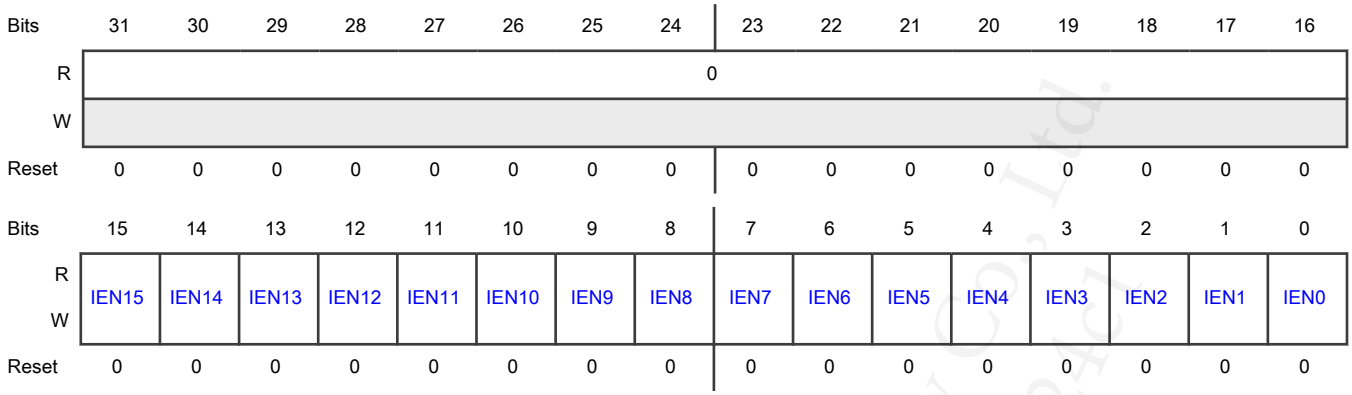
Register	Offset
C3INTHER	8Ch

Function

This register controls the masking of corresponding flags inside C3INTHR. ACCESS GUIDELINE- Write/read access for all cores and HOST.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15 IEN15	This bit enables/disables FLG15 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
14 IEN14	This bit enables/disables FLG14 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
13 IEN13	This bit enables/disables FLG13 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
12 IEN12	This bit enables/disables FLG12 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
11 IEN11	This bit enables/disables FLG11 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
10 IEN10	This bit enables/disables FLG10 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
9 IEN9	This bit enables/disables FLG9 propagation to the interrupt line

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Flag disabled 1b - Flag enabled
8 IEN8	This bit enables/disables FLG8 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
7 IEN7	This bit enables/disables FLG7 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
6 IEN6	This bit enables/disables FLG6 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
5 IEN5	This bit enables/disables FLG5 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
4 IEN4	This bit enables/disables FLG4 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
3 IEN3	This bit enables/disables FLG3 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
2 IEN2	This bit enables/disables FLG2 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
1 IEN1	This bit enables/disables FLG1 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled
0 IEN0	This bit enables/disables FLG0 propagation to the interrupt line 0b - Flag disabled 1b - Flag enabled

54.12.4 Functional description

The core-to-core communication interface provides the capability for each core, including the host, to transmit and receive events via flags and maskable interrupts to each other.

The event indication between transmitting core and receiving core is facilitated by two types of registers: (symbolic notation)

1. (RxCORE)INT(TxCORE)R – Receive Core Interrupt from Transmit Core Register. This register is accessed to set or clear event flags. Transmit core sets a flag. Receive core clears the flag.
2. (RxCORE)INT(TxCORE)ER – Receive Core Interrupt from Transmit Core Enable Register. This register is accessed to enable or disable the propagation of the corresponding flag in (RxCORE)INT(TxCORE)R to interrupt line.

A flag(bit) written by Transmit core into (RxCORE)INT(TxCORE)R sets the flag in the register. A flag can be set when transmitting core wants a service from receiving core. The receiving core will be interrupted if enable bit for that flag is set in (RxCORE)INT(TxCORE)ER. Otherwise, the flag will not generate an interrupt but will remain set. Only the receiving core can reset the flag by writing 1 into the bit.

ACCESS Guidelines:

1. (RxCORE)INT(TxCORE)R can be written by only TxCORE (To set a flag) and RxCORE (To clear a flag). It can be read by all cores and HOST.
2. (RxCORE)INT(TxCORE)ER can be written and read by all cores and HOST.

Block diagram shows the interrupt generation scheme with access types.

For example, assume Core 0 wants to indicate two events to HOST. One event is required to generate an interrupt and the other required to set a flag only.

1. Write 1 to HINTC0ER[0] to generate an interrupt from the flag HINTC0R[0], keeping all other bits 0. This can be done by any core or host.
2. Core 0 sets bit HINTC0R[1] to indicate the first event to HOST. No interrupt is generated for this as it is not enabled (HINTC0ER[1]=0), only the flag is set.
3. Core 0 sets bit HINTC0R[0] to indicate the second event to HOST. This generates an interrupt to HOST along with this flag because it is enabled (HINTC0ER[0]=1).
4. The HOST, in reaction to the interrupt, reads the HINTC0R register and clears flags HINTC0R[0] and HINTC0R[1] by writing 1 to each of them after completing the requested task.
5. Flags HINTC0R[0] and HINTC0R[1] are now free to be written by Core 1 again.

Each core has its own flag/interrupt set registers and flag mask registers.

The transmitting core sets the interrupt/flag to the register of receiving core. The same physical register is read/write for transmitting core and write only one (W1C) for the receiving core. Each bit of this Flag/interrupt register can generate interrupt if enabled otherwise it is used as a flag and to be polled by receiving core.

The receiving core clears the interrupts/flags in its own Flag/interrupt register. So, the transmitting core should check the current flag status before setting it. There is no protection taken if the transmitting core clears a flag/interrupt itself before the receiving core is actually read and acted on it.

Each master has fixed Master IDs in the system. The below figure shows the decoding of master IDs to generate access protection.

Cores of LLCE have master ID values are hard coded to identify them uniquely (DTE=3, RXPPE=4, TXPPE=5, FRPE=6, Host=8).

The Core-to-Core module has five master ID input ports, each dedicated to one masters, 1 for host and 4 for four cores. The host port is fixed at value 0x8. Internal core ports are assigned to fixed indexes as below::

- Core 0 is DTE.
- Core 1 is RXPPE.
- Core 2 is TXPPE.
- Core 3 is FRPE.

Index assignment and fixed values of master IDs:

- Host master ID 8h

- Core0 master ID 3h (DTE)
- Core1 master ID 4h (RXPPE)
- Core2 master ID 5h (TXPPE)
- Core3 master ID 6h (FRPE)

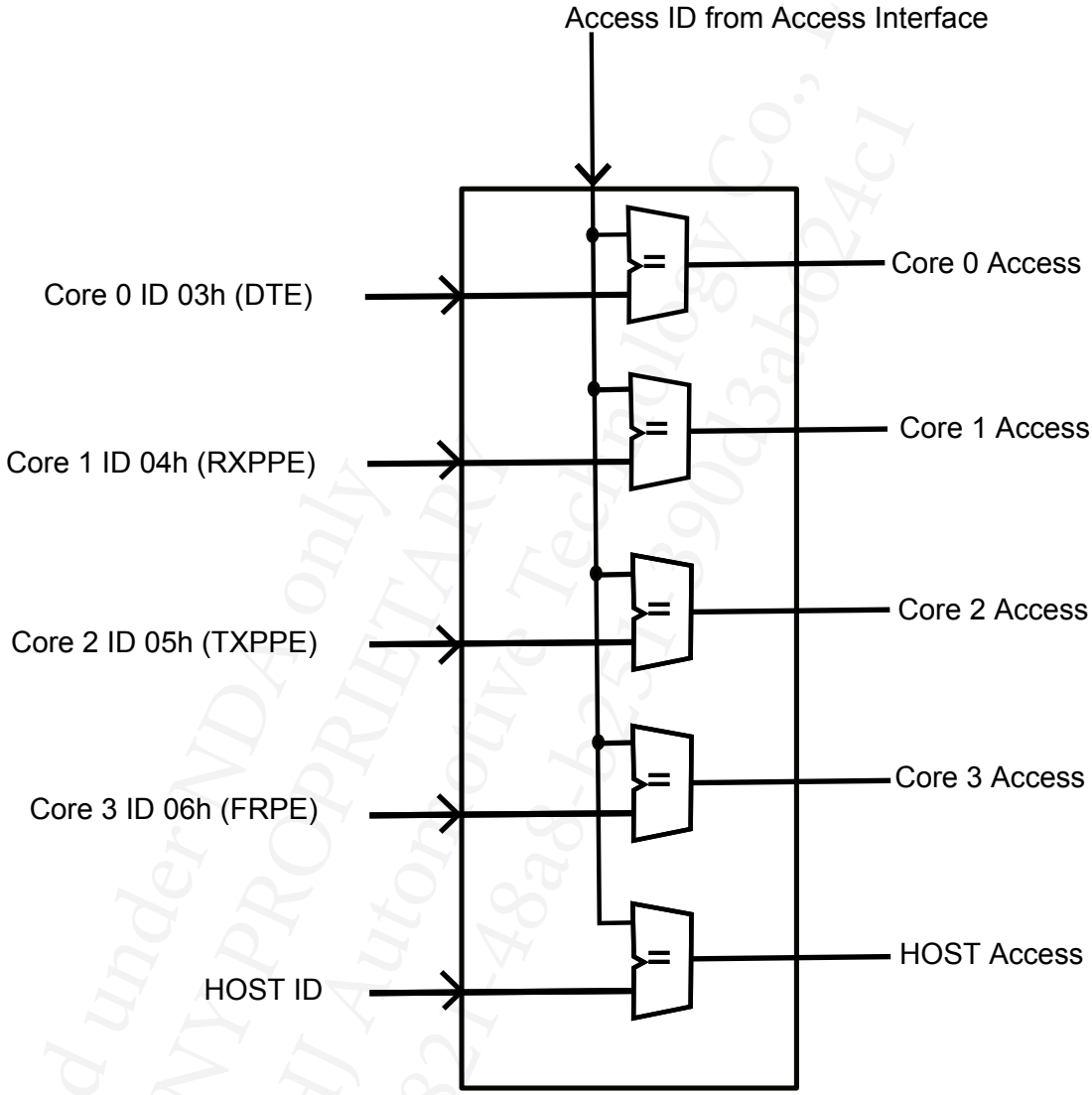


Figure 453. Master ID decoding scheme

Access protection

Access to (RxCORE)INT(TxCORE)R registers is protected against access from disallowed cores. The master ID from access interface is compared against the value of the master ID of Core 0-3 and HOST.

- Access to set the flag to a core if its ID matches TxCORE ID.
- Access to clear the flag is granted to a core if its ID matches RxCORE ID

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8d65e5-4821-48a8-b25-90d3ab024c1

54.13 Interrupts and status

Multiple interrupts generate from LLCE's internal modules. The interrupt logic of LLCE does concentration, keeps status, and routes them per configuration. The output goes to IRQ lines of LLCE cores and to host.

There are two modules inside LLCE:

- Interrupt concentrator module (IC)
- Interrupt routing configuration module (IRCM)

54.13.1 Interrupt concentrator (IC)

The IC takes interrupts from FIFOs and Core2Core and concentrates (OR-ed) them and present to IRCM.

Concentration occurs in two stages:

- First Level: concentration of interrupts from a single module
- Second level: concentration of interrupts from first level or directly from different modules (where first level is not required)

There are status registers for the interrupts for which second level of concentration helps in identifying the interrupting module by reading one register only. These registers are read only and corresponding status is automatically cleared when its source is cleared. The output of concentrator module goes to interrupt routing configuration module (IRCM).

54.13.2 Interrupt Routing Configuration Module (IRCM)

The IRCM module holds the configuration that defines the destinations of an interrupt coming to its input. It has a configuration register IRSPRC n (interrupt router shared peripheral routing control register) corresponding to each input. This register has as many bits as the number of destination possible for the interrupt. The number of destination for each input interrupt is fixed per its functional requirements. One input may have single or multiple destinations. In both the cases, the bit specified for a destination enables or disables the interrupt towards it. An interrupt which has multiple destination can go to all or some of the selected destinations if enabled in its register.

The destinations are:

- LLCE Core 0 (DTE)
- LLCE Core 1 (RxPPE)
- LLCE Core 2 (TxPPE)
- LLCE Core 3 (FRPE)
- Host

B line interrupts from each 8 FIFOs are captured on same status register (ICSR*) inside IC and concentrated (OR-ed) at second level to feed to IRCM.The IRCM register IRSPRC* holds the bits to route the interrupt to different locations. The below table shows the FIFO instances and corresponding registers to check the interrupt status and control the routing.

Module	Module Instances	Interrupt Concentrator Status Register	Interrupt Description (Input to IRCM)	Routing Control Register	Host	C3 (FRPE)	C2 (TxPPE)	C1 (RxPPE)	C0 (DTE)
				Register Bit	HostE	CP3E	CP2E	CP1E	CP0E
Core2Core	Core2Core	NA	Core0 Interrupt from Host	IRSPRC8	-	-	-	-	2

Table continues on the next page...

Table continued from the previous page...

			Core1 Interrupt from Host	IRSPRC 10	-	-	-	2	-
			Core2 Interrupt from Host	IRSPRC 12	-	-	2	-	-
			Core3 Interrupt from Host	IRSPRC 14	-	2	-	-	-
			Host Interrupt from Core0	IRSPRC 16	0	-	-	-	-
			Host Interrupt from Core1	IRSPRC 17	1	-	-	-	-
			Host Interrupt from Core2	IRSPRC 18	2	-	-	-	-
			Host Interrupt from Core3	IRSPRC 19	3	-	-	-	-
FIFO	RX-IN_FIFO_7_0	ICSR0	Combined A line interrupt	IRSPRC 28	-	-	4	4	4
	RX-IN_FIFO_15_8	ICSR1	Combined A line interrupt	IRSPRC 29	-	-	5	5	5
	RX-OUT_FIFO_7_0	ICSR2	Combined A line interrupt	IRSPRC 30	-	-	6	6	6
	RX-OUT_FIFO_15_8	ICSR3	Combined A line interrupt	IRSPRC 31	-	-	7	7	7
	BLR-IN_FIFO_7_0	ICSR4	Combined A line interrupt	IRSPRC 32	-	-	8	8	8
	BLR-IN_FIFO_15_8	ICSR5	Combined A line interrupt	IRSPRC 33	-	-	9	9	9

Table continues on the next page...

Table continued from the previous page...

	BLR-OUT_FIFO_7_0	ICSR6	Combined A line interrupt	IRSPRC 34	-	-	10	10	10
	BLR-OUT_FIFO_15_8	ICSR7	Combined A line interrupt	IRSPRC 35	-	-	11	11	11
	TX_ACK_FIFO_7_0	ICSR8	Combined A line interrupt	IRSPRC 36	-	-	12	12	12
	TX_ACK_FIFO_15_8	ICSR9	Combined A line interrupt	IRSPRC 37	-	-	13	13	13
	RX-OUT_FIFO_19_16, RX-IN_FIFO_19_16	ICSR10	Combined A line interrupt	IRSPRC 38	-	14	-	14	14
	BLR-OUT_FIFO_19_16, BLR-IN_FIFO_19_16	ICSR11	Combined A line interrupt	IRSPRC 39	-	15	-	15	15
	TX_ACK_FIFO_19_16	ICSR12	Combined A line interrupt	IRSPRC 40	-	16	16	-	16
	TX_ACK_FIFO_20 BLR-OUT_FIFO_20 BLR-IN_FIFO_20 RX-OUT_FIFO_20 RX-IN_FIFO_20	ICSR13	Combined A line interrupt	IRSPRC 41	-	17	17	-	17

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

	RX-IN_FIFO_7_0	ICSR14	Combined B line interrupt	IRSPRC 42	4	-	-	-	-
	RX-IN_FIFO_15_8	ICSR15	Combined B line interrupt	IRSPRC 43	5	-	-	-	-
	RX-OUT_FIFO_7_0	ICSR16	Combined B line interrupt	IRSPRC 44	6	-	-	-	-
	RX-OUT_FIFO_15_8	ICSR17	Combined B line interrupt	IRSPRC 45	7	-	-	-	-
	BLR-IN_FIFO_7_0	ICSR18	Combined B line interrupt	IRSPRC 46	8	-	-	-	-
	BLR-IN_FIFO_15_8	ICSR19	Combined B line interrupt	IRSPRC 47	9	-	-	-	-
	BLR-OUT_FIFO_7_0	ICSR20	Combined B line interrupt	IRSPRC 48	10	-	-	-	-
	BLR-OUT_FIFO_15_8	ICSR21	Combined B line interrupt	IRSPRC 49	11	-	-	-	-
	TX_ACK_FIFO_7_0	ICSR22	Combined B line interrupt	IRSPRC 50	12	-	-	-	-
	TX_ACK_FIFO_15_8	ICSR23	Combined B line interrupt	IRSPRC 51	13	-	-	-	-
	RX-OUT_FIFO_19_16, RX-IN_FIFO_19_16	ICSR24	Combined B line interrupt	IRSPRC 52	14	-	-	-	-
	BLR-OUT_FIFO_19_16, BLR-IN_FIFO_19_16	ICSR25	Combined B line interrupt	IRSPRC 53	15	-	-	-	-

Table continues on the next page...

Table continued from the previous page...

	TX_ACK_FIFO_19_16	ICSR26	Combined B line interrupt	IRSPRC54	16	-	-	-	-
	TX_ACK_FIFO_20 BLR-OUT_FIFO_20 BLR-IN_FIFO_20 RX-OUT_FIFO_20 RX-IN_FIFO_20	ICSR27	Combined B line interrupt	IRSPRC55	17	-	-	-	-

54.14 External signal description

The LLCE hardware communication channels control the following I/O. The host controller configures the I/O through SIUL2_*n*.

Table 387. LLCE external signal description

Name	Function
LLCE_CAN[0:15]_TX	FlexCAN transmit data
LLCE_CAN[0:15]_RX	FlexCAN receive data
LLCE_LIN[0:3]_TX	LINFlexD transmit data
LLCE_LIN[0:3]_RX	LINFlexD receive data
LLCE_FR_TXD_A	FlexRay channel A transmit data
LLCE_FR_TXD_B	FlexRay channel B transmit data
LLCE_FR_TXE_A	FlexRay channel A transmit enable
LLCE_FR_TXE_B	FlexRay channel B transmit enable
LLCE_FR_RXD_A	FlexRay channel A receive data
LLCE_FR_RXD_B	FlexRay channel B receive data
LLCE_FR_DBG_[0:3]	FlexRay debug channels
LLCE_LPSPI[0_3]_SIN	LPSPI serial data in
LLCE_LPSPI[0_3]_SOUT	LPSPI serial data out
LLCE_LPSPI[0_3]_SCLK	LPSPI serial data clock

Table continues on the next page...

Table 387. LLCE external signal description (continued)

Name	Function
LLCE_LPSPI[0_3]_PCS[0:3]_I	LPSPI chip select in
LLCE_LPSPI[0_3]_PCS[0:3]_O	LPSPI chip select out

54.15 LLCE register descriptions

54.15.1 System control memory map

LLCE_SYSCTRL base address: 43FF_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	System Reset Control (LLCE_SYSRSTR)	32	RW	0000_0000h

54.15.2 System Reset Control (LLCE_SYSRSTR)

Offset

Register	Offset
LLCE_SYSRSTR	0h

Function

Controls the reset of various modules inside LLCE. The default value keeps the modules under reset. You must write 1 to deassert the resets of each module. This register is accessible only from the host CPU port.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												CPU_ RST3	CPU_ RST2	CPU_ RST1	CPU_ RST0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4821-48ab-b251-390d8a624c1

Fields

Field	Function
31-4 —	Reserved
3 CPU_RST3	CPU Reset 3 Resets core 3 (CM0+ inside FRPE). 0b - CPU under reset 1b - CPU out of reset
2 CPU_RST2	CPU Reset 2 Resets core 2 (CM0+ inside TxPPE). 0b - CPU under reset 1b - CPU out of reset
1 CPU_RST1	CPU Reset 1 Resets core 1 (CM0+ inside RxPPE). 0b - CPU under reset 1b - CPU out of reset
0 CPU_RST0	CPU Reset 0 Resets core 0 (CM0+ inside DTE). 0b - CPU under reset 1b - CPU out of reset

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 55
Universal Serial Bus On-The-Go Controller (USBOTG)

55.1 Chip-specific USB information

55.1.1 Chip-specific USBOTG information

Weak pulldown is required on the DIR pin of ULPI interface. In case, the external PHY is completely powered down, all the inputs need to be pulled down so that the pads should not remain floating.

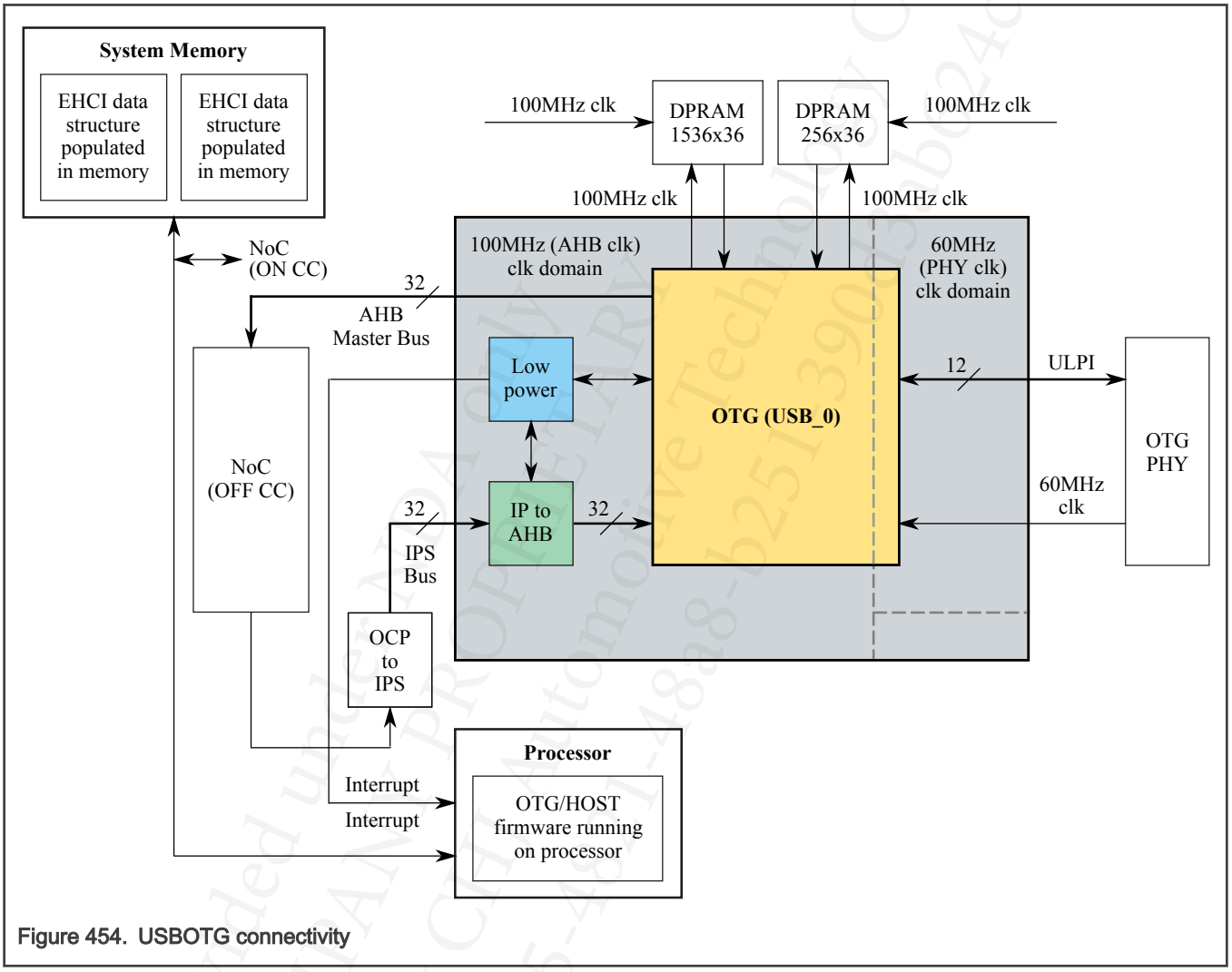


Figure 454. USBOTG connectivity

55.1.2 SBUSCFG[AHBBRST] field reset value

AHB Master Interface Burst Configuration (AHBBRST) field controls the AHB master transfer type sequence (or priority). This chip is designed to support only 011b reset value for this field. The chip operation is not guaranteed for any other programmed values.

55.2 Introduction

This chapter describes the USB high-speed (HS) OTG controller (USBOTG), which implements many industry standards. See the standards' governing specifications for more information. To use this chapter most effectively, you need to be fluent in the operation and requirements of a USB network.

See the USB implementers forum at http://www.usb.org/developers/docs for:

- *Universal Serial Bus Specification, Revision 2.0*
- *On-The-Go Supplement to the USB 2.0 Specification, Revision 1.0a*

See the Intel USB specifications at http://www.intel.com/ for:

- *Enhanced Host Controller Interface Specification for Universal Serial Bus, Revision 1.0*

See the ULPI specifications at http://www.ulpi.org for:

- *UTMI+ Specification, Revision 1.0*
- *UTMI Low Pin Interface (ULPI) Specification, Revision 1.1*

55.2.1 Overview

USBOTG is a USB 2.0-compliant serial interface engine for implementing a USB interface. The registers and data structures are based on the *Enhanced Host Controller Interface Specification for Universal Serial Bus* (EHCI) from Intel Corporation. This module can act as a host, a device, or an On-The-Go (OTG) negotiable host or device on the USB bus.

The USBOTG controller interfaces with the processor's core. The controller is configured to support host or device operations under firmware control. The ULPI (UTMI Low Pin Interface) interface supports HS, full-speed (FS), and low-speed (LS) applications.

55.2.2 Features

- Complies with USB specification rev 2.0
- USB Host mode
  - Supports EHCI
- USB Device mode
  - Supports FS/HS operation via an external ULPI transceiver
  - Supports one upstream facing port
  - Supports six programmable, bidirectional USB endpoints, including endpoint 0. See endpoint configurations in the following table:

Table 388. Endpoint configurations

Endpoint	Type	FIFO size	Data transfer	Comments
0	Bidirectional	Variable	Control	Mandatory
1-5	IN or OUT	Variable	Ctrl, int, bulk, or ISO	Optional

- Suspend mode/low-power
  - As host, firmware can suspend individual devices or the entire USB and disable chip clocks for low-power operation.
  - Device supports low-power suspend.
  - Remote wake-up is supported for host and device.
- Support for off-chip HS, FS, LS transceiver
  - External ULPI transceiver supports HS, FS, and LS operations in Host mode, and HS and FS operations in Device mode.
  - Interface uses an 8-bit, single-data-rate ULPI data bus.
  - ULPI PHY supplies a 60 MHz USB reference clock input to the USB controller.

55.2.3 Modes of operation

This module has two basic operating modes: Host and Device. Selection of an operating mode is accomplished via the USBMODE[CM] field.

Speed selection is auto-detected at connect time using enumeration procedures in the USB network. This module provides these operation modes:

- USB disabled: In this mode, the USB datapath does not accept transactions received on the USB interface.
- USB enabled: In this mode, the datapath of the USB host is enabled to accept transactions received on the USB interface.
- USB enabled, low-power modes.

55.3 External signal description

This table describes the external signal functionality of the USBOTG module.

Table 389. Signal descriptions

Signal	I/O	Description	
ULPI interface			
ULPI_CLK	I/O	60 MHz clock input from the ULPI transceiver	
ULPI_DIR	O	Direction. ULPI_DIR controls data bus direction. When PHY has data to transfer to USB port, it drives ULPI_DIR high to take ownership of the bus. When the PHY has no data to transfer, it drives ULPI_DIR low and monitors the bus for link activity. The PHY pulls ULPI_DIR high when the interface cannot accept data from the link; for example, when PHY's PLL is not stable.	
		State meaning	Asserted—PHY has data to transfer to the link. Negated—PHY has no data to transfer.
		Timing	Synchronous to ULPI_CLK
ULPI_NXT	I	Next data. PHY asserts ULPI_NXT to throttle data. When the USB port sends data to the PHY, ULPI_NXT indicates when PHY accepts the current byte. The USB port places the next byte on the data bus in the following clock cycle. When the PHY sends data to USB port, ULPI_NXT indicates when a new byte is available for USB port to consume.	
		State meaning	Asserted—PHY is ready to transfer byte. Negated—PHY is not ready.
		Timing	Synchronous to ULPI_CLK
ULPI_STP	I	Stop. ULPI_STP indicates the end of a transfer on the bus.	
		State meaning	Asserted—USB asserts this signal for one clock cycle to stop the data stream currently on the bus. If the USB port sends data to the PHY, ULPI_STP indicates the last data byte was previously on the bus. If the PHY is sending data to the USB port, ULPI_STP forces the PHY to end its transfer, deassert ULPI_DIR, and relinquish control of the data bus to the USB port. Negated—Indicates normal operation.

Table continues on the next page...

Table 389. Signal descriptions (continued)

Signal	I/O	Description	
		Timing	Synchronous to ULPI_CLK
ULPI_DATA[7:0]	I/O	Data bit *n*. ULPI_DATA*n* is bit *n* of the 8-bit, bi-directional data bus used to carry USB, register, and interrupt data between the USB port controller and the PHY.	
		State meaning	Asserted—Data bit *n* is 1. Negated—Data bit *n* is 0.
		Timing	Synchronous to ULPI_CLK

55.4 USBOTG register descriptions

There are two types of registers in the USB module: USB core registers (UOTG) and USB non-core registers (UOTGNC).

The UOTG are used to control USB core functions and are more independent of USB features. Each USB controller core has its own core registers.

The UOTGNC are additional to the UOTG and are more dependent upon USB features.

NOTE

You must preserve a register's reserved field values when writing (that is, read its reset value, then write this value back).

The UOTGNC register name prefix indicates a USB non-core register.

55.4.1 Core register descriptions

55.4.1.1 UOTG memory map

UOTG base address: 4406_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Identification (ID)	32	RO	E4A1_FA05h
4h	General Hardware Parameters (HWGENERAL)	32	RO	0000_0083h
8h	Host Hardware Parameters (HWHOST)	32	RO	1002_0001h
Ch	Device Hardware Parameters (HWDEVICE)	32	RO	0000_000Dh
10h	TX Buffer Hardware Parameters (HWTXBUF)	32	RO	8008_0B04h
14h	RX Buffer Hardware Parameters (HWRXBUF)	32	RO	0000_0804h
80h	General Purpose Timer Load (GPTIMER0LD)	32	RW	0000_0000h
84h	General Purpose Timer Controller (GPTIMER0CTRL)	32	RW	0000_0000h
88h	General Purpose Timer Load (GPTIMER1LD)	32	RW	0000_0000h
8Ch	General Purpose Timer Controller (GPTIMER1CTRL)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
90h	System Bus Configuration (SBUSCFG)	32	RW	0000_01FBh
100h	Capability Register Length (CAPLENGTH)	8	RO	40h
102h	Host Controller Interface Version (HCIVERSION)	16	RO	0100h
104h	Host Controller Structural Parameters (HCSPARAMS)	32	RO	0001_0011h
108h	Host Controller Capability Parameters (HCCPARAMS)	32	RO	0000_0006h
120h	Device Controller Interface Version (DCIVERSION)	16	RO	0001h
124h	Device Controller Capability Parameters (DCCPARAMS)	32	RO	0000_0186h
140h	USB Command (USBCMD)	32	RW	0008_0B00h
144h	USB Status (USBSTS)	32	W1C	0000_0000h
148h	Interrupt Enable (USBINTR)	32	RW	0000_0000h
14Ch	USB Frame Index (FRINDEX)	32	RW	0000_0000h
154h	Frame List Base Address or Device Address (PERIODICLISTBASE_or_DEVICEADDR)	32	RW	0000_0000h
158h	Next Asynchronous Address or Endpoint List Address (ASYNCLISTADDR_or_ENDPOINTLISTADDR)	32	RW	0000_0000h
160h	Programmable Burst Size (BURSTSIZE)	32	RW	0000_1010h
164h	TX FIFO Fill Tuning (TXFILLTUNING)	32	RW	0000_0000h
16Ch	IC_USB Enable and Voltage Negotiation (IC_USB)	32	RW	0000_0000h
170h	ULPI Register Access (ULPI_VIEWPORT)	32	RW	0800_0000h
178h	Endpoint NAK (ENDPTNAK)	32	W1C	0000_0000h
17Ch	Endpoint NAK Enable (ENDPTNAKEN)	32	RW	0000_0000h
180h	Configure Flag (CONFIGFLAG)	32	ROO	0000_0001h
184h	Port Status and Control (PORTSC1)	32	RW	8C00_0004h
1A4h	OTG Status Control (OTGSC)	32	RW	0000_1020h
1A8h	USB Device Mode (USBMODE)	32	RW	0000_5000h
1ACh	Endpoint Setup Status (ENDPTSETUPSTAT)	32	RW	0000_0000h
1B0h	Endpoint Initialization (ENDPTPRIME)	32	RW	0000_0000h
1B4h	Endpoint De-Initialize (ENDPTFLUSH)	32	RW	0000_0000h
1B8h	Endpoint Status (ENDPTSTAT)	32	RO	0000_0000h
1BCh	Endpoint Complete (ENDPTCOMPLETE)	32	RW	0000_0000h
1C0h	Endpoint Control 0 (ENDPTCTRL0)	32	RW	0080_0080h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1C4h - 1D4h	Endpoint Control (ENDPTCTRL1 - ENDPTCTRL5)	32	RW	0000_0000h

55.4.1.2 Identification (ID)

Offset

Register	Offset
ID	0h

Function

Identifies the USB 2.0 OTG HS core and its revision.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								REVISION			0				
W																
Reset	1	1	1	0	0	1	0	0	1	0	1	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		NID						0		ID					
W																
Reset	1	1	1	1	1	0	1	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-25 —	Reserved
24-21 REVISION	Revision Number of the Controller Core
20-16 —	Reserved
15-14	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48ab-b251-590d3ab624c1

Table continued from the previous page...

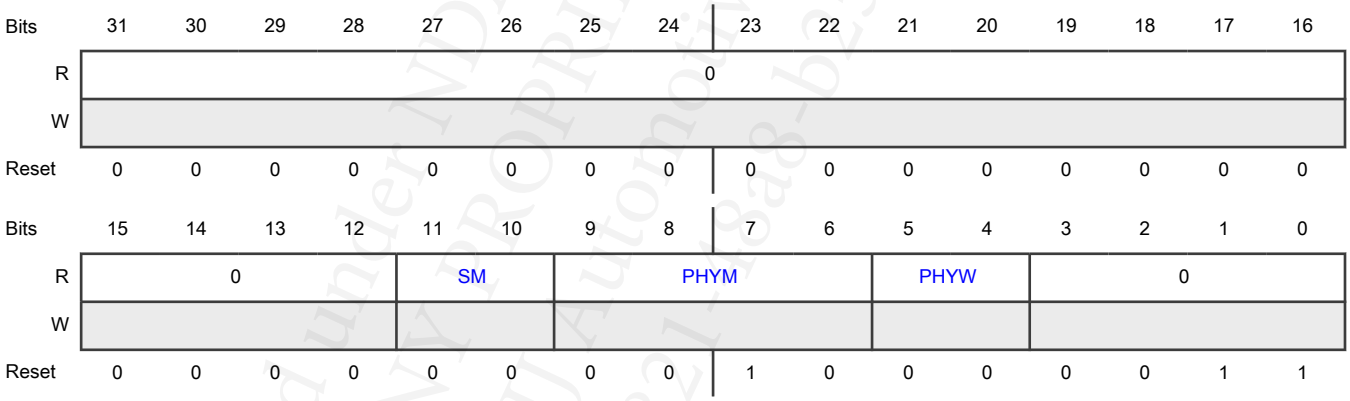
Field	Function
—	
13-8 NID	Complement Version of ID
7-6 —	Reserved
5-0 ID	Configuration Number This number is set to 05h and indicates that the peripheral is USB 2.0 OTG HS core.

55.4.1.3 General Hardware Parameters (HWGENERAL)

Offset

Register	Offset
HWGENERAL	4h

Diagram



Fields

Field	Function
31-12 —	Reserved
11-10 SM	Serial Interface Mode Capability Selects between parallel and serial signaling for LS and FS applications.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - No serial engine; always use parallel signaling 01b - Serial engine present; always use serial signaling for FS or LS 10b - Software-programmable reset to use parallel signaling for FS or LS 11b - Software-programmable reset to use serial signaling for FS or LS
9-6 PHYM	Transceiver Type Selects transceiver type and software-programmable reset. 0000b - UTMI/UMTI+ 0001b - ULPI DDR 0010b - ULPI 0011b - Serial only 0100b - Software-programmable reset to UTMI/UTMI+ 0101b - Software-programmable reset to ULPI DDR 0110b - Software-programmable reset to ULPI 0111b - Software-programmable reset to Serial 1000b - IC-USB 1001b - Software-programmable reset to IC-USB 1010b - HSIC 1011b - Software-programmable reset to HSIC
5-4 PHYW	Data Width of the Transceiver connected to the controller core Selects data bus width of the transceiver connected to the controller core and software-programmable reset. 00b - 8-bit wide data bus [60 MHz clock from the transceiver] 01b - 16-bit wide data bus [30 MHZ clock from the transceiver] 10b - Software-programmable reset to 8-bit width 11b - Software-programmable reset to 16-bit width
3-0 —	Reserved

55.4.1.4 Host Hardware Parameters (HWHOST)

Offset

Register	Offset
HWHOST	8h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46255-4821-4c8a-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												NPORT			HC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31-4 —	Reserved
3-1 NPORT	Number of Ports Contains the number of downstream ports supported by the host controller is NPORT+1. This field is set to 000b for this single-port controller.
0 HC	Host Capable 0b - Host operation mode not supported 1b - Host operation mode supported

55.4.1.5 Device Hardware Parameters (HWDEVICE)

Offset

Register	Offset
HWDEVICE	Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								DEVEP							DC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1

Fields

Field	Function
31-7 —	Reserved
6-1 DEVEP	Device Endpoint Number The OTG controller core has six endpoints.
0 DC	Device Capable 0b - Device operation mode not supported 1b - Device operation mode supported

55.4.1.6 TX Buffer Hardware Parameters (HWTXBUF)

Offset

Register	Offset
HWTXBUF	10h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								TXCHANADD							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXADD								TXBURST							
W																
Reset	0	0	0	0	1	0	1	1	0	0	0	0	0	1	0	0

Fields

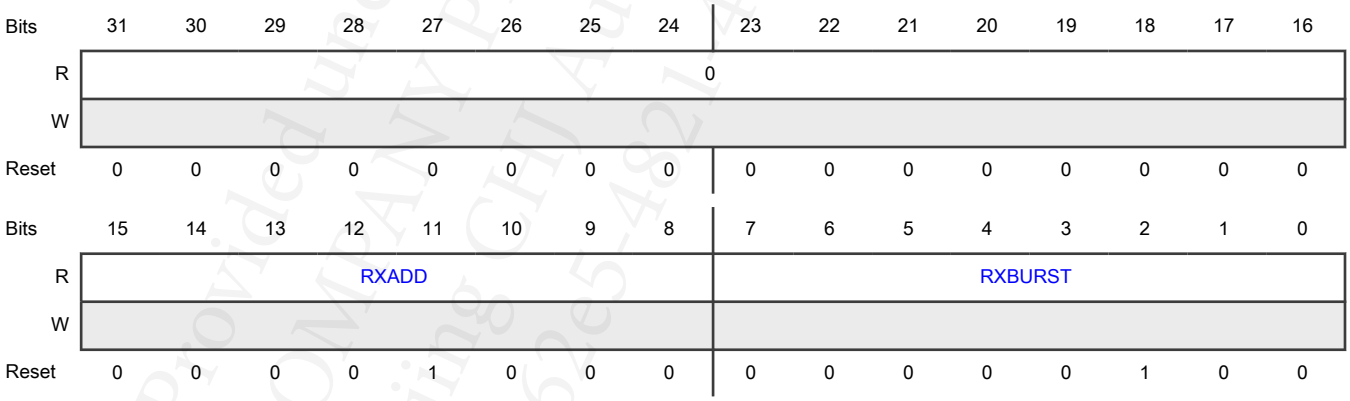
Field	Function
31-24 —	Reserved
23-16 TXCHANADD	TX Channel Address Indicates the number of address bits required for TX channel buffer addressing. The TX buffer size is equal to $2^{TXCHANADD}$×4 bytes. For the OTG controller core, there is one TX buffer for each endpoint, therefore there are six TX buffers in all. The value of this field is configured as 8h, therefore, the buffer size is 256×4 bytes.
15-8 TXADD	TX Address Indicates the number of address bits required for TX buffer addressing. The buffer total size for all transmit endpoints is 2^{RXADD}.
7-0 TXBURST	TX Burst Size Indicates the default burst size for memory to TX buffer transfer. This is the reset value of BURSTSIZE[TXPBURST]. See BURSTSIZE for details.

55.4.1.7 RX Buffer Hardware Parameters (HWRXBUF)

Offset

Register	Offset
HWRXBUF	14h

Diagram



Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c467e54821148a8-b251-390d3ab624c1

Fields

Field	Function
31-16 —	Reserved
15-8 RXADD	RX Address Indicates the number of address bits required for RX buffer addressing. The buffer total size for all receive endpoints is 2^{RXADD}. RX buffer size is equal to 2^{RXADD}×4 bytes. For the OTG controller core, there is one RX buffer shared by six endpoints. The value of this field is set to 08h, therefore, the buffer size is 256×4 bytes.
7-0 RXBURST	RX Burst size Indicates the default burst size for memory to RX buffer transfer. This is the reset value of BURSTSIZE[RXPBURST]. See BURSTSIZE for details.

55.4.1.8 General Purpose Timer Load (GPTIMER0LD - GPTIMER1LD)

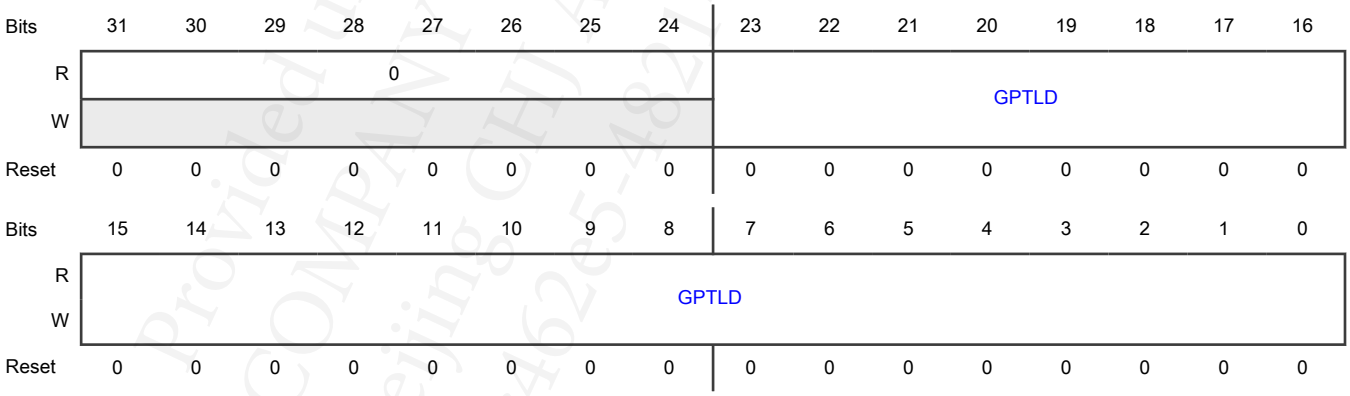
Offset

Register	Offset
GPTIMER0LD	80h
GPTIMER1LD	88h

Function

Controls the load value of the count timer in GPT*n*CL. See GPTIMER0CTRL for details.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-4da8-b251-390d3ab624c1

Fields

Field	Function
31-24 —	Reserved
23-0 GPTLD	General Purpose Timer Load Value These bits are loaded to the GPTCNT bits when the GPTRST field is set to 1. This value represents the time in microseconds minus 1 for the timer duration. For example, a one millisecond timer, load 1000-1=999, or 0003E7h. **NOTE** Maximum value is FFFFFFh, or 16.777215 seconds.

55.4.1.9 General Purpose Timer Controller (GPTIMER0CTRL - GPTIMER1CTRL)

Offset

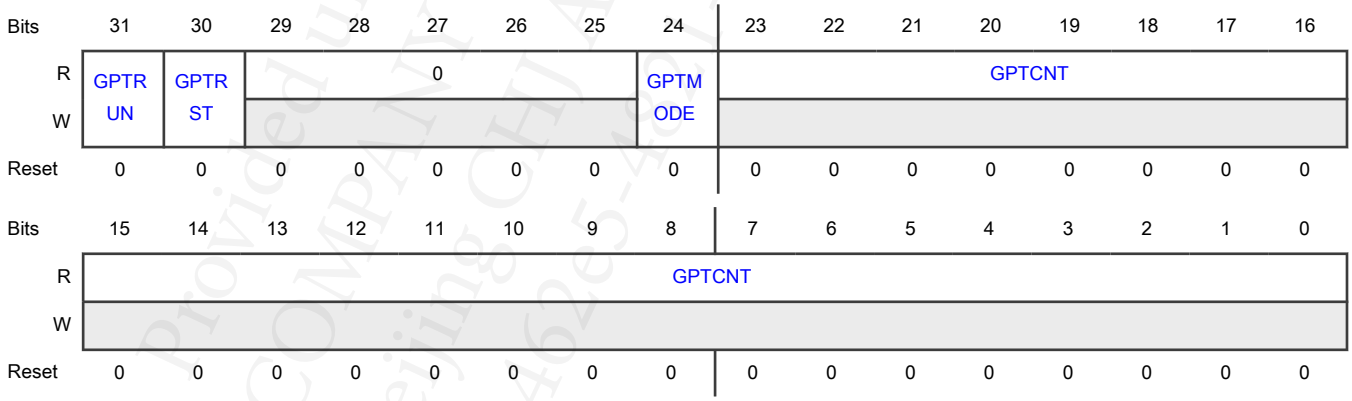
Register	Offset
GPTIMER0CTRL	84h
GPTIMER1CTRL	8Ch

Function

Contains the countdown timer control and a data field can be queried to determine the running count value. This timer has a granularity of 1 us and can be programmed to slightly more than 16 seconds. There are two counter modes that are described in the register table. When the timer counter value transitions to 0, an interrupt could be generated if enabled.

Interrupt status field is USBSTS[TI0]; interrupt enable field is USBINTR[TIE0].

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4e1-48a-b251-390dab624c1

Fields

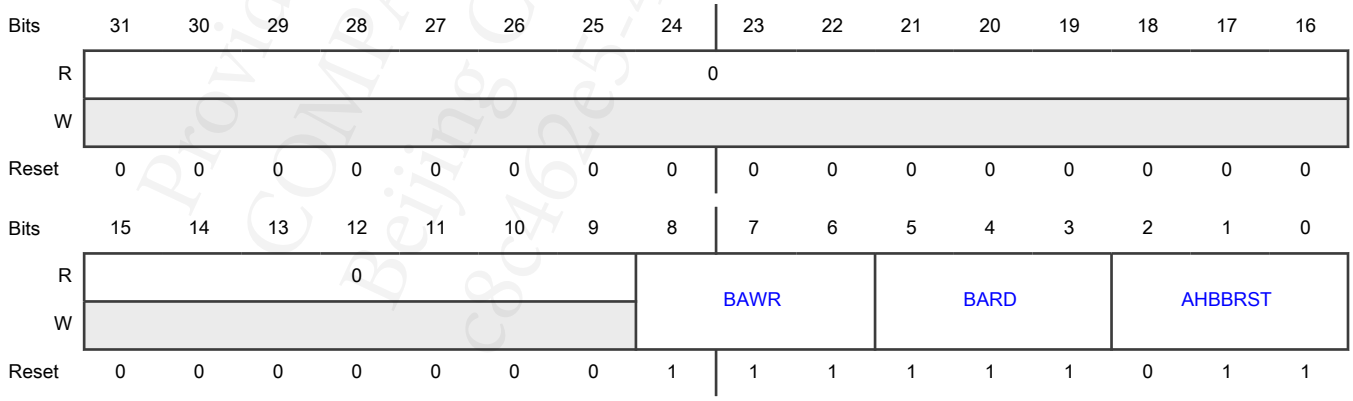
Field	Function
31 GPTRUN	General Purpose Timer Run The GPTCNT bits are not affected when setting or clearing this field. 0b - Stop counting 1b - Run
30 GPTRST	General Purpose Timer Reset 0b - No action 1b - Load the counter value from GPT0LD[GPTLD]
29-25 —	Reserved
24 GPTMODE	General Purpose Timer Mode In One-Shot mode, the timer counts down to 0, generates an interrupt, and stops until you reset the counter. In Repeat mode, the timer counts down to 0, generates an interrupt, and automatically reloads the counter value from the GPTLD bits to start again. 0b - One-Shot mode 1b - Repeat mode
23-0 GPTCNT	General Purpose Timer Counter Contains the count value of the countdown timer.

55.4.1.10 System Bus Configuration (SBUSCFG)

Offset

Register	Offset
SBUSCFG	90h

Diagram



Fields

Field	Function
31-9 —	Reserved
8-6 BAWR	AMBA Burst Alignment for Writes To improve performance on system bus architectures where unaligned bursts are a problem, this mode guarantees that AMBA-AHB burst transfers never cross the burst size address boundary. Burst transfers that start unaligned are broken into smaller aligned bursts until aligned bursts of maximum size can be used. This aligns all packet data AMBA burst transfers to their respective address boundaries: • INCR16–64 byte boundary • INCR8–32 byte boundary • INCR4–16 byte boundary • SINGLE–4 byte boundary This mode only applies to packet data transfers, not data structures. When burst alignment is enabled, you can select a minimum threshold for the packet size that uses aligned bursts when being written to system memory. • 000 - Disable burst alignment • 001 - Align bursts for packets larger than 512 bytes • 010 - Align bursts for packets larger than 256 bytes • 011 - Align bursts for packets larger than 128 bytes • 100 - Align bursts for packets larger than 64 bytes • 101 - Align bursts for packets larger than 32 bytes • 110 - Align bursts for packets larger than 16 bytes • 111 - Align bursts for all packets This field configures the burst alignment only for AMBA writes.
5-3 BARD	AMBA Burst Alignment for Reads Same function as BAWR above, but for AMBA reads.
2-0 AHBBRST	AHB Master Interface Burst Configuration Controls the AHB master transfer type sequence (or priority). **NOTE** Overrides BURSTSIZE when its value is not 0. 000b - Incremental burst of unspecified length only 001b - INCR4 burst, then single transfer 010b - INCR8 burst, INCR4 burst, then single transfer 011b - INCR16 burst, INCR8 burst, INCR4 burst, then single transfer

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - Reserved 101b - INCR4 burst, then incremental burst of unspecified length 110b - INCR8 burst, INCR4 burst, then incremental burst of unspecified length 111b - INCR16 burst, INCR8 burst, INCR4 burst, then incremental burst of unspecified length

55.4.1.11 Capability Register Length (CAPLENGTH)

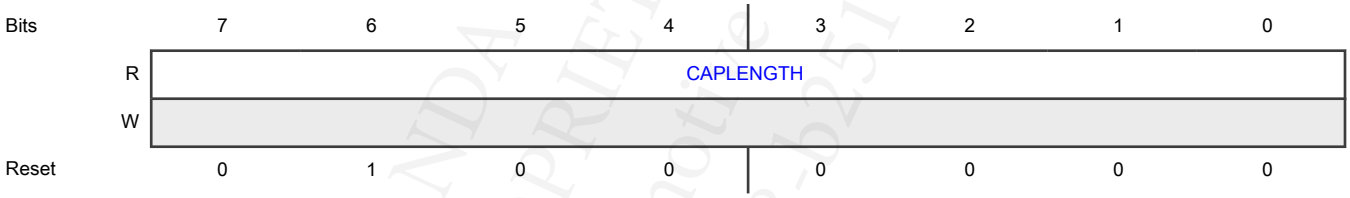
Offset

Register	Offset
CAPLENGTH	100h

Function

Indicates the offset that must be added to the register base address at the beginning of the operational register.

Diagram



Fields

Field	Function
7-0 CAPLENGTH	This field is used as an offset to be added to the register base to find the beginning of the Operational register.

55.4.1.12 Host Controller Interface Version (HCIVERSION)

Offset

Register	Offset
HCIVERSION	102h

Function

Contains a BCD encoding of the EHCI revision number supported by this host controller. The most significant byte of this register represents a major revision and the least significant byte is the minor revision.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HCIVERSION															
W																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 HCIVERSION	Host Controller Interface Version Number For example, the value 10h indicates EHCI revision 1.0

55.4.1.13 Host Controller Structural Parameters (HCSPARAMS)

Offset

Register	Offset
HCSPARAMS	104h

Function

Indicates the port steering logic capabilities of Host Controller Structural Parameters (HCSPARAMS).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				N_TT				N_PTT				0			PI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	N_CC				N_PCC				0			PPC	N_PORTS			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a3-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
27-24 N_TT	Number of Transaction Translators Indicates the number of embedded transaction translators associated with the USB2.0 host controller. 0000b - Single-port host 0001b - Multi-port host
23-20 N_PTT	Number of Ports Per Transaction Translator Indicates the number of ports assigned to each transaction translator within the USB2.0 host controller. This field is set to equal N_PORTS for Multi-Port Host, and 0000b for single-port host.
19-17 —	Reserved
16 PI	Port Indicators Indicates whether the ports support port indicator control. This field is 1b in the controller core. 0b - The ports do not support port indicator control. 1b - The port status and control registers include a read/writeable field for controlling the state of the port indicator.
15-12 N_CC	Number of Companion Controllers Indicates the number of companion controllers associated with this USB2.0 host controller. This field is 0000b in the controller core. 0000b - There is no internal companion controller and port-ownership hand-off is not supported. 0001b - There are internal companion controller(s) and port-ownership hand-offs are supported.
11-8 N_PCC	Number of Ports Per Companion Controller Indicates the number of ports supported per internal companion controller. It is used to indicate the port routing configuration to the system software. For example, if N_PORTS has a value of 6 and N_CC has a value of 2 then N_PCC could have a value of 3. The convention is that the first N_PCC ports are assumed to be routed to companion controller 1, the next N_PCC ports to companion controller 2, and so on. In the previous example, the N_PCC could have been 4, where the first 4 are routed to companion controller 1 and the last two are routed to companion controller 2. The number in this field must be consistent with N_PORTS and N_CC. This field is 0000b in the controller core.
7-5 —	Reserved
4 PPC	Port Power Control

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates whether the host controller implementation includes port power control. The value of this field affects the functionality of the port power field in each port status and control register. 0b - The ports do not have port power switches. 1b - The ports have port power switches.
3-0 N_PORTS	Number of Downstream Ports Specifies the number of physical downstream ports implemented on this host controller. The value of this field determines how many port registers are addressable in the Operational register. Valid values are in the range of 1h to Fh. The value 0h in this field is undefined. This field is always set to 0001b because the controller core is a single-port host.

55.4.1.14 Host Controller Capability Parameters (HCCPARAMS)

Offset

Register	Offset
HCCPARAMS	108h

Function

Identifies multiple mode control (time-base bit functionality) addressing capabilities.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EECP								IST				0	ASP	PFL	ADC
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
31-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 EECP	EHCI Extended Capabilities Pointer Indicates the existence of a capabilities list. A value of 00h indicates that no extended capabilities are implemented. A non-zero value in this register indicates the offset in PCI configuration space of the first EHCI extended capability. If implemented, the pointer value must be 40h or greater to maintain the consistency of the PCI header defined for this class of device. This field has a fixed value of 00h.
7-4 IST	Isochronous Scheduling Threshold Indicates, relative to the current position of the executing host controller, where you can reliably update the isochronous schedule. When bit [7] is 0, the value of the least significant 3 bits indicates the number of microframes for which a host controller can hold a set of isochronous data structures (one or more) before flushing the state. When IST[3] is 1, then the host software assumes that the host controller may cache an isochronous data structure for an entire frame. This field has a fixed value of 0h.
3 —	Reserved
2 ASP	Asynchronous Schedule Park Capability Supports the park feature for HS QHs in the asynchronous schedule. The feature can be disabled or enabled and set to a specific level by using USBCMD[ASPE] and USBCMD[ASP]. This field has a fixed value of 1.
1 PFL	Programmable Frame List Flag If you write 0 to this field, then the system software must use a frame list length of 1024 elements with this host controller. You must write 0 to USBCMD[FS1]. If you write 1 to this field, then the system software can specify and use a smaller frame list and configure the host controller via USBCMD[FS1]. The frame list must always be aligned on a 4K-page boundary. This requirement ensures that the frame list is always physically contiguous. This field has a fixed value of 1.
0 ADC	64-bit Addressing Capability This field has a fixed value of 0 and 64-bit addressing capability is not supported.

55.4.1.15 Device Controller Interface Version (DCIVERSION)

Offset

Register	Offset
DCIVERSION	120h

Function

Indicates the two-byte BCD encoding of the device controller interface version number.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DCIFVER															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
15-0 DCIFVER	Device Controller Interface Version Number For example, 01h means rev0.1.

55.4.1.16 Device Controller Capability Parameters (DCCPARAMS)

Offset

Register	Offset
DCCPARAMS	124h

Function

Describes the overall device capability of the controller.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							HC	DC	0	DEN					
W																
Reset	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHTC Automotive Technology Co., Ltd.
c8c467e5-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-9 —	Reserved
8 HC	Host Capable When the value of this field is 1, this controller is capable of operating as an EHCI compatible USB 2.0 host controller.
7 DC	Device Capable When the value of this field is 1, this controller is capable of operating as a USB 2.0 device.
6 —	Reserved
5-0 DEN	Device Endpoint Number Indicates the number of endpoints built into the device controller. If this controller is not device capable, then the value of this field is 0. Valid values are 0 - 15.

55.4.1.17 USB Command (USBCMD)

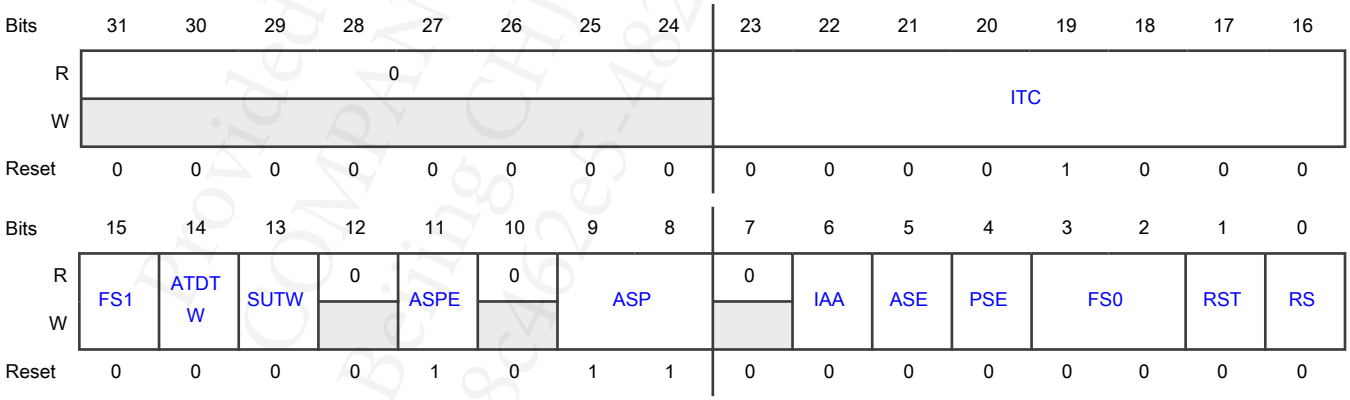
Offset

Register	Offset
USBCMD	140h

Function

Indicates the command to be executed by the serial bus host/device controller. Writing to the register causes a command to be executed.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8a52e5-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-24 —	Reserved
23-16 ITC	Interrupt Threshold Control The system software uses this field to set the maximum rate at which the host/device controller issues interrupts. ITC contains the maximum interrupt interval measured in microframes. Valid values are shown below. Maximum interrupt interval values are these: 00000000b - Immediate (no threshold) 00000001b - 1 microframe 00000010b - 2 microframes 00000100b - 4 microframes 00001000b - 8 microframes 00010000b - 16 microframes 00100000b - 32 microframes 01000000b - 64 microframes
15 FS1	Frame List Size See also FS0. This field is read/write only if the value of Programmable Frame List Flag in the HCCPARAMS registers is 1. Specifies the size of the frame list that controls which fields in USB Frame Index (FRINDEX) must be used for the frame list current index. **NOTE** This field is made up from USBCMD[FS1] and USBCMD[FS0]. 000 - 1024 elements (4096 bytes) 001 - 512 elements (2048 bytes) 010 - 256 elements (1024 bytes) 011 - 128 elements (512 bytes) 110 - 16 elements (64 bytes) 111 - 8 elements (32 bytes)
14 ATDTW	Add device transfer descriptor (dTD) TripWire Used as a semaphore to ensure proper addition of a new dTD to an active (primed) endpoint's linked list. This field is set and cleared by you. This field is cleared by hardware when state machine is hazard region for which adding a dTD to a primed endpoint may go unrecognized.

Table continues on the next page...

Table continued from the previous page...

Field	Function
13 SUTW	Setup TripWire Used as a semaphore to ensure that the setup data payload of eight bytes is extracted from a QH by the device controller driver (DCD) without being corrupted. If the setup lockout mode is off (USBMODE[SLOM] is 1), then there is a hazard when new setup data arrives while the DCD is copying the setup data payload from the QH for a previous setup packet. This field is set and cleared by you. The hardware also writes 0 to this field when a hazard is detected.
12 —	Reserved
11 ASPE	Asynchronous Schedule Park Mode Enable If HCCPARAMS[ASP] is 0, then this field defaults to 1h and is R/W. Otherwise, the field must be 0 and is RO. You use this field to enable or disable Park mode. When this field is 1, Park mode is enabled. When this field is a 0, Park mode is disabled. This field is set to 1h (the default value).
10 —	Reserved
9-8 ASP	Asynchronous Schedule Park Mode Count If HCCPARAMS[ASP] is 1, then this field defaults to 3 and is read/write. Otherwise, it defaults to 0 and is read-only. It contains a count of the number of successive transactions the host controller is allowed to execute from an HS QH on the asynchronous schedule before continuing traversal of the asynchronous schedule. Valid values are 1 to 3. You must not write a 0 to this field when Park Mode Enable is a 1 as this results in undefined behavior. This field is set to 3h (the default value).
7 —	Reserved
6 IAA	Interrupt on Async Advance Doorbell You use this as a doorbell to tell the host controller to issue an interrupt the next time it advances the asynchronous schedule. You must write 1 to this field to ring the doorbell. When the host controller has evicted all appropriate cached schedule states, it sets the Interrupt on the Async Advance status field in USB Status (USBSTS). If the Interrupt on Sync Advance Enable field in Interrupt Enable (USBINTR) is 1, then the host controller asserts an interrupt at the next interrupt threshold. The host controller clears this field to 0 after it has set the Interrupt on Sync Advance status field in USB Status (USBSTS) to 1. You must not write a 1 to this field when the asynchronous schedule is inactive; doing so yields undefined results. This field is only used in host mode. Writing 1 to this field when device mode is selected yields undefined results.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 ASE	Asynchronous Schedule Enable Controls whether the host controller skips processing of the asynchronous schedule. Only the host controller uses this field. 0b - Do not process the asynchronous schedule 1b - Use ASYNCLISTADDR to access the asynchronous schedule
4 PSE	Periodic Schedule Enable Controls whether the host controller skips processing the periodic schedule. Only the host controller uses this field. 0b - Do not process the periodic schedule 1b - Use PERIODICLISTBASE to access the periodic schedule
3-2 FS0	FS0 See description at FS1.
1 RST	Controller Reset You use this field to reset the controller. It is cleared to 0 by the Host/Device controller when the reset process is complete. You cannot terminate the reset process early by writing a 0 to this register. Host operation mode: When you write 1 to this field, the controller resets its internal pipelines, timers, counters, state machines, and so on, to their initial value. Any transaction currently in progress on the USB is immediately terminated. A USB reset is not driven on downstream ports. You must not write 1 to this field when USBSTS[HCH] is 0. Attempting to reset an actively running host controller results in undefined behavior. Device operation mode: When you write 1 to this field, the controller resets its internal pipelines, timers, counters, state machines, and so on, to their initial values. Writing 1 to this field, when the device is in the attached state, is not recommended because the effect on an attached host is undefined. To ensure that the device is not in an attached state before initiating a device controller reset, all primed endpoints must be flushed, and USBCMD[RS] must be written to 0. The reset value of this field is gated with the reset state machine. If the PHY clock is not up and running, the read value of this field is 1.
0 RS	Run/Stop Host Operation mode: When the value of this field is 1, the controller proceeds with the execution of the schedule. The controller continues execution as long as this field is set to 1. When this field is cleared to 0, the host controller completes the current transaction on the USB and then halts. The HC Halted field in the status register indicates when the controller has finished the transaction and has entered the stopped state. You must not write a 1 to this field unless the controller is in the Halted state (that is, USBSTS[HCH] is a 1). Device Operation mode: Writing a 1 to this field causes the controller to enable a pull-up on D+ and initiate an attach event. This control field is not directly connected to the pull-up enable, as the pull-up becomes

Table continues on the next page...

Table continued from the previous page...

Field	Function
	disabled upon transitioning into HS mode. You must use this field to prevent an attach event before the controller has been properly initialized. Writing a 0 to this field causes a detach event. 0b - Stop 1b - Run

55.4.1.18 USB Status (USBSTS)

Offset

Register	Offset
USBSTS	144h

Function

Indicates various states of the host/device controller and any pending interrupts. The register does not indicate status resulting from a transaction on the serial bus.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						TI1	TI0	0				UPI	UAI	0	NAKI
W							W1C	W1C					W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	AS	PS	RCL	HCH	UALTI	ULPII	0	SLI	SRI	URI	AAI	SEI	FRI	PCI	UEI	UI
W					W1C	W1C		W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25 TI1	General Purpose Timer Interrupt 1 This field is set to 1 when the counter in the GPT1CL register transitions to 0.
24	General Purpose Timer Interrupt 0

Table continues on the next page...

Table continued from the previous page...

Field	Function
TI0	This field is set to 1 when the counter in the GPT0CL register transitions to 0.
23-20 —	Reserved
19 UPI	USB Host Periodic Interrupt The host controller writes 1 to this field when the cause of an interrupt is a completion of a USB transaction in which the transfer descriptor (TD) has an interrupt on complete (IOC) field set to 1 and the TD was from the periodic schedule. This field is also set to 1 by the host controller when a short packet is detected and the packet is on the periodic schedule. A short packet occurs when the actual number of bytes received is less than expected. This field is not used by the device controller and is always 0.
18 UAI	USB Host Asynchronous Interrupt The host controller writes 1 to this field when the cause of an interrupt is completion of a USB transaction in which the TD has an IOC field set to 1 and the TD originates from the asynchronous schedule. This field is also set to 1 by the host when a short packet is detected and the packet is on the asynchronous schedule. A short packet occurs when the actual number of bytes received is less than expected. This field is not used by the device controller and is always 0.
17 —	Reserved
16 NAKI	NAK Interrupt This field is set to 1 by hardware when, for a particular endpoint, both the TX/RX Endpoint NAK fields (ENDPTNAK[EPTN] and ENDPTNAK[ERTN]) and the corresponding TX/RX Endpoint NAK Enable fields (ENDPTNAKEN[EPTNE] and ENDPTNAKEN[EPRNE]) are 1. This field is automatically cleared to 0 by hardware when all Enabled TX/RX Endpoint NAK fields are cleared to 0.
15 AS	Asynchronous Schedule Status Reports the current real status of the asynchronous schedule. When cleared to 0, the asynchronous schedule status is disabled; if set to 1, the status is enabled. The host controller is not required to immediately disable or enable the asynchronous schedule when you transition USBCMD[ASE]. When this field and USBCMD[ASE] are the same value, the asynchronous schedule is either enabled (1) or disabled (0). This field is only used in the host operation mode.
14 PS	Periodic Schedule Status Reports the current real status of the periodic schedule. When cleared to 0, the periodic schedule is disabled, and if set to 1 the status is enabled. The host controller is not required to immediately disable or enable the periodic schedule when you transition the Periodic Schedule Enable field in USB Command (USBCMD). When this field and the Periodic Schedule Enable field are the same value, the periodic schedule is either enabled (1) or disabled (0).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is only used in the host operation mode.
13 RCL	Reclamation This status field is used to detect an empty asynchronous schedule. It is only used in the host operation mode.
12 HCH	HC Halted This field is a 0 whenever the Run/Stop field is a 1. The controller sets this field to 1 after it has stopped executing because of the Run/Stop field being cleared to 0, either by you or by the controller hardware (for example, on an internal error). This field is only used in the host operation mode. The OTG core does not operate as a host by default. See USBMODE[CM].
11 UALTI	ULPI alt_int Interrupt This interrupt field is set to 1 when an RXCMD is received through the ULPI interface with bit 7 set (alt_int). The alt_int itself is set when an unmasked event occurs on any bit in the Carkit Interrupt Latch register, in the ULPI PHY. You must read the Carkit Interrupt Latch register (Read to Clear) through the ULPI Viewport to check the source of the interrupt.
10 ULPII	ULPI Interrupt This field is set to 1 by hardware when there is an event completion in the ULPI viewport. This field is usable only if the controller supports ULPI interface mode.
9 —	Reserved
8 SLI	DC Suspend When a controller enters a suspend state from an active state, the value of this field is 1. The device controller clears the field to 0 upon exiting from a suspend state. This field is only used in device operation mode.
7 SRI	SOF Received When the device controller detects a start-of-(micro)frame, this field is set to 1. When a SOF is extremely late, the device controller automatically sets this field to 1 to indicate that an SOF was expected. Therefore, this field is set roughly every 1 ms in device FS mode and every 125 ms in HS mode, and is synchronized to the actual SOF that is received. Because the device controller is initialized to FS before connect, this field is set to 1 at an interval of 1 ms during the prelude to connect and chirp. In host mode, this field is set every 125 µs and can be used by the host controller driver as a time base. You write a 1 to this field to clear it.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 URI	USB Reset Received When the device controller detects a USB Reset and enters the default state, this field is set to 1. You can write a 1 to this field to clear the USB Reset Received status field to 0. This field is only used in device operation mode.
5 AAI	Interrupt on Async Advance You can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing a 1 to USBCMD[IAA]. This status field indicates the assertion of that interrupt source. This field is only used in host operation mode.
4 SEI	System Error This field is set to 1 when an error is seen in response to a read on the system interface.
3 FRI	Frame List Rollover The host controller sets this field to 1 when the Frame List Index rolls over from its maximum value to 0. The exact value at which the rollover occurs depends on the frame list size. For example, if the frame list size (as programmed in USBCMD[FS1]) is 1024, USB Frame Index (FRINDEX) rolls over every time FRINDEX [13] toggles. Similarly, if the size is 512, the host controller sets this field to 1 every time FHINDEX [12] toggles. This field is only used in host operation mode.
2 PCI	Port Change Detect The host controller sets this field to 1 when, on any port, a Connect Status occurs, a Port Enable/Disable Change occurs, or the Force Port Resume field is set as the result of a J-K transition on the suspended port. The device controller sets this field to 1 when the port controller enters the FS or HS operational state. When the port controller exits the FS or HS operation states due to Reset or Suspend events, the notification mechanisms are the USB Reset Received and the DCSuspend fields, respectively.
1 UEI	USB Error Interrupt When completion of a USB transaction results in an error condition, this field is set to 1 by the host/device controller. This field is set to 1, along with the USBSTS[UI] field, if the TD on which the error interrupt occurred also had its IOC field set to 1. The device controller detects resume signaling only.
0 UI	USB Interrupt Set by the host/device controller when the cause of an interrupt is a completion of a USB transaction where the TD has an IOC field set to 1. This field is also set by the host/device controller when a short packet is detected. A short packet is when the actual number of bytes received was less than the expected number of bytes.

55.4.1.19 Interrupt Enable (USBINTR)

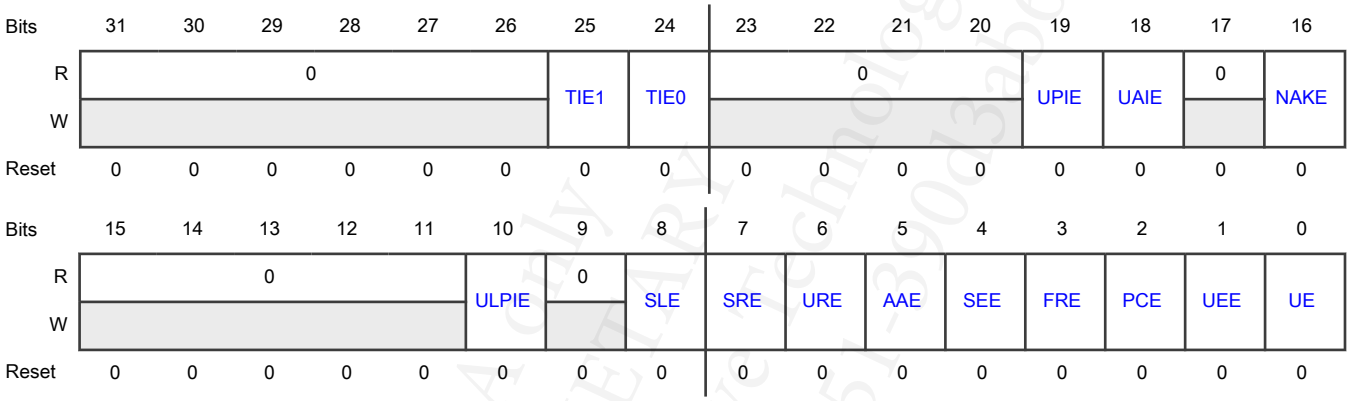
Offset

Register	Offset
USBINTR	148h

Function

Enables interrupts to software. An interrupt is generated when a field is 1 and the corresponding interrupt source is active. USB Status (USBSTS) still shows interrupt sources even if they are disabled by the USBINTR, allowing you to poll interrupt events.

Diagram



Fields

Field	Function
31-26 —	Reserved
25 TIE1	General Purpose Timer 1 Interrupt Enable When this field is 1 and USBSTS[TI1] is 1, the controller issues an interrupt.
24 TIE0	General Purpose Timer 0 Interrupt Enable When this field is 1 and USBSTS[TI0] is 1, the controller issues an interrupt.
23-20 —	Reserved
19 UPIE	USB Host Periodic Interrupt Enable When this field is 1 and USBSTS[UPI] is 1, the host controller issues an interrupt at the next interrupt threshold.
18	USB Host Asynchronous Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
UAIE	When this field is 1 and USBSTS[UAI] is 1, the host controller issues an interrupt at the next interrupt threshold.
17 —	Reserved
16 NAKE	NAK Interrupt Enable When this field is 1 and USBSTS[NAKI] is 1, the controller issues an interrupt.
15-11 —	Reserved
10 ULPIE	ULPI Interrupt Enable When this field is 1 and USBSTS[ULPII] is 1, the controller issues an interrupt. This field is usable only if the controller supports ULPI interface mode.
9 —	Reserved
8 SLE	Sleep Interrupt Enable When this field is 1 and USBSTS[SLI] is 1, the controller issues an interrupt. This field is only used in device operation mode.
7 SRE	SOF Received Interrupt Enable When this field is 1 and USBSTS[SRI] is 1, the controller issues an interrupt.
6 URE	USB Reset Interrupt Enable When this field is 1 and USBSTS[URI] is 1, the controller issues an interrupt. This field is only used in device operation mode.
5 AAE	Async Advance Interrupt Enable When this field is 1 and USBSTS[AAI] is 1, the controller issues an interrupt. This field is only used in host operation mode.
4 SEE	System Error Interrupt Enable When this field is 1 and USBSTS[SEI] is 1, the controller issues an interrupt.
3 FRE	Frame List Rollover Interrupt Enable When this field is 1 and USBSTS[FRI] is 1, the controller issues an interrupt. This field is only used in host operation mode.
2	Port Change Detect Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
PCE	When this field is 1 and USBSTS[PCI] is 1, the controller issues an interrupt.
1 UEE	USB Error Interrupt Enable When this field is 1 and USBSTS[UEI] is 1, the controller issues an interrupt.
0 UE	USB Interrupt Enable When this field is 1 and USBSTS[UI] is 1, the controller issues an interrupt.

55.4.1.20 USB Frame Index (FRINDEX)

Offset

Register	Offset
FRINDEX	14Ch

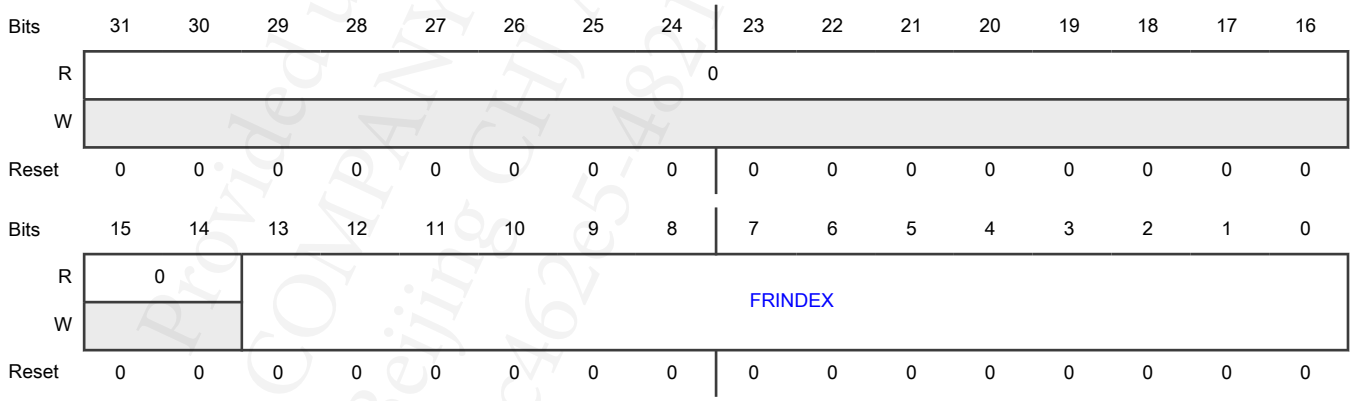
Function

Used by the host controller to index the periodic frame list. The register updates every 125 µs (once each microframe). Bits [N:3] are used to select a particular entry in the periodic frame list during periodic schedule execution. The number of bits used for the index depends on the size of the frame list that you set in USBCMD[FS1].

This register must be written as a DWord. Byte writes produce undefined results. This register cannot be written unless the host controller is in the Halted state as indicated by USBSTS[HCH]. A write to this register while USBCMD[RS] is set to 1 produces undefined results. Writes to this register also affect the SOF value.

In Device mode, this register is read-only, and the device controller updates it using the frame number indicated by the SOF marker. Whenever the USB receives and SOF, FRINDEX [13:3] is checked against the SOF marker. If FRINDEX [13:3] is different from the SOF marker, FRINDEX [13:3] will be set to the SOF value and FRINDEX [2:0] becomes 0 (that is, SOF for a 1 ms frame). If FRINDEX [13:3] is equal to the SOF value, FRINDEX [2:0] is incremented (that is, SOF for a 125 µs microframe.).

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0 FRINDEX	Frame Index In this field, the value increments at the end of each time frame (microframe). Bits [N:3] are used for the Frame List current index. This means that each location of the frame list is accessed eight times (frames or microframes) before moving to the next index. The following illustrates values of N, based on the value of USBCMD[FS1], when used in Host mode. In Device mode, the value is the current frame number of the last frame transmitted. It is not used as an index. In either mode, bits 2:0 indicate the current microframe. USBCMD [Frame List Size] Number Elements N 00000000000000b - (1024) 12 00000000000001b - (512) 11 00000000000010b - (256) 10 00000000000011b - (128) 9 00000000000100b - (64) 8 00000000000101b - (32) 7 00000000000110b - (16) 6 00000000000111b - (8) 5

55.4.1.21 Frame List Base Address or Device Address (PERIODICLISTBASE_or_DEVICEADDR)

Offset

Register	Offset
PERIODICLISTBASE_or_DEVICEADDR	154h

Function

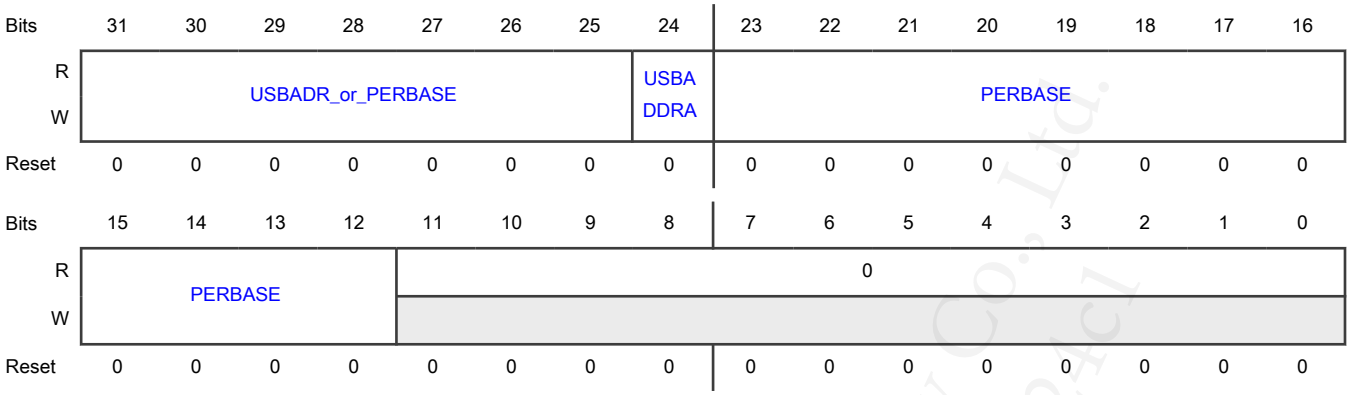
PERIODICLISTBASE - For host controller only:

Contains the beginning address of the periodic frame list in the system memory. The host controller device (HCD) loads this register prior to starting the schedule execution by the host controller. The memory structure referenced by this physical memory pointer is assumed to be 4 KB aligned. The contents of this register are combined with the ones of USB Frame Index (FRINDEX) to enable the host controller to step through the periodic frame list in sequence.

DEVICEADDR - For device controller only:

The upper seven bits of this register represent the device address. After any controller reset or a USB reset, the device address is set to the default address (0). The default address will match all incoming addresses. You must reprogram the address after receiving a SET_ADDRESS descriptor.

Diagram



Fields

Field	Function
31-25 USBADR_or_PERBASE	Device Address (For Device mode) or PERBASE (For Host mode) Device mode: These bits correspond to the USB device address. Host mode: See the PERBASE description.
24 USBADDRA	Device Address Advance (For Device mode) or Reserved (For Host mode) Device mode: When this field is 0, any writes to USBADR are instantaneous. When this field is written to 1 at the same time or before USBADR is written, the write to the USBADR field is staged and held in a hidden register. After an IN occurs on endpoint 0 and generates an ACK response, USBADR is loaded from the holding register. Hardware automatically clears this field to 0 on the following conditions: • IN generates an ACK to endpoint 0 (USBADR is updated from the hidden register). • OUT/SETUP occurs to endpoint 0 (USBADR is not updated). • Device Reset occurs (USBADR is reset to 0). **NOTE** • After the status phase of the SET_ADDRESS descriptor, the DCD has 2 ms to program the USBADR field. This mechanism ensures that this specification is met when the DCD cannot execute a write of the device address within 2 ms from the SET_ADDRESS status phase. If the DCD writes the USBADR with USBADRA=1 after the SET_ADDRESS data phase (before the prime of the status phase), the USBADR is programmed instantly at the correct time and meet the 2 ms USB requirement. • This field is reserved for Host mode operation.
23-12 PERBASE	PERBASE (For Host mode) or Reserved (for Device mode) Periodic List Base Address Corresponds to memory address signals [31:12].

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is only used by the host controller. **NOTE** This field is reserved for Device mode operation.
11-0 —	Reserved

55.4.1.22 Next Asynchronous Address or Endpoint List Address (ASYNCLISTADDR_or_ENDPOINTLISTADDR)

Offset

Register	Offset
ASYNCLISTADDR_or_ENDPOINTLISTADDR	158h

Function

Next asynchronous address - for host controller only:

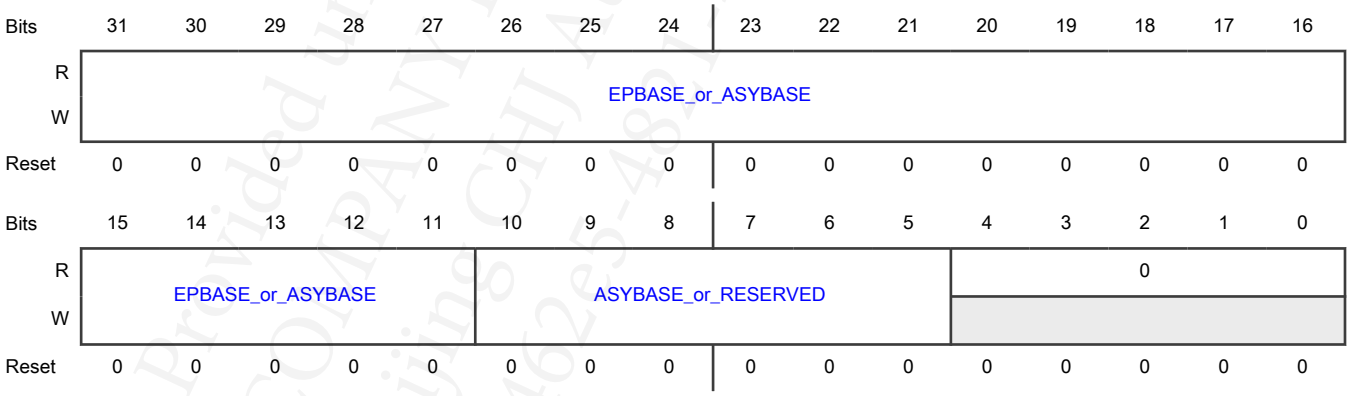
Contains the address of the next asynchronous QH to be executed by the host. You cannot modify bits [4:0] of this register, and when you read them they always return 0s.

Endpoint list address - for device controller only:

In Device mode, this register contains the address of the top of the endpoint list in system memory. You cannot modify bits [10:0] of this register, and when you read them they always return 0s.

The memory structure referenced by this physical memory pointer is assumed to be 64 bytes.

Diagram



Fields

Field	Function
31-11 EPBASE_or_ASYBASE	Endpoint List Pointer (Low) (for Device mode) or Link Pointer Low (LPL) (for Host mode) Device mode: These bits correspond to memory address signals [31:11]. This field references a list of up to 32 QH (that is, one QH per endpoint and direction). Host mode: See the ASYBASE description.
10-5 ASYBASE_or_RESERVED	Link Pointer Low (LPL) (for Host mode) or Reserved (for Device mode) Host mode: These bits correspond to memory address signals [31:5]. This field may only reference a QH. **NOTE** This field is reserved for Device mode operation.
4-0 —	Reserved

55.4.1.23 Programmable Burst Size (BURSTSIZE)

Offset

Register	Offset
BURSTSIZE	160h

Function

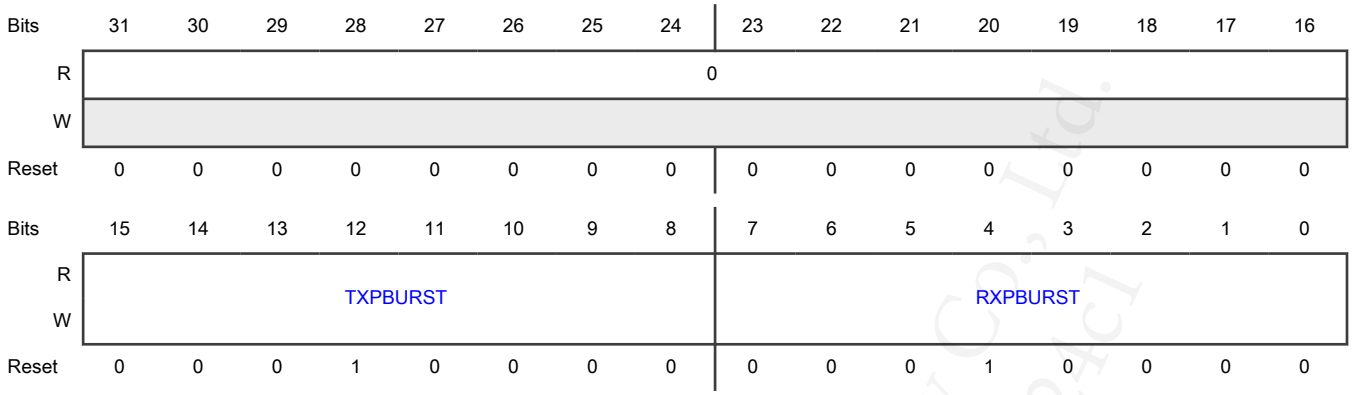
Controls the burst size used during data movement on the AHB master interface. The register is ignored if SBUSCFG[AHBBRST] is a non-zero value.

NOTE

The AHB maximum burst length is 16 beats. Each beat can be 4 bytes; therefore, the maximum burst supported on the AHB is 64 bytes.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4811-48a8-b251-390a3a624c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15-8 TXPBURST	Programmable TX Burst Size HWTXBUF[TXBURST] determines the default value. This field represents the maximum burst length in 32-bit words while moving data from system memory to the USB bus.
7-0 RXPBURST	Programmable RX Burst Size HWRXBUF[RXBURST] determines the default value. This field represents the maximum burst length in 32-bit words while moving data from the USB bus to system memory.

55.4.1.24 TX FIFO Fill Tuning (TXFILLTUNING)

Offset

Register	Offset
TXFILLTUNING	164h

Function

The fields in this register control performance tuning associated with how the host controller posts data to the TX latency FIFO before moving the data onto the USB bus. The specific areas of performance include how much data to post into the FIFO and an estimate of the length the operation should take in the target system.

Definitions:

T_0 = Standard packet overhead

T_1 = Time to send data payload

T_{ff} = Time to fetch packet into TX FIFO up to specified level

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48ac-b251-390d3ab624c1

T_s = Total Packet Flight Time (send-only) packet

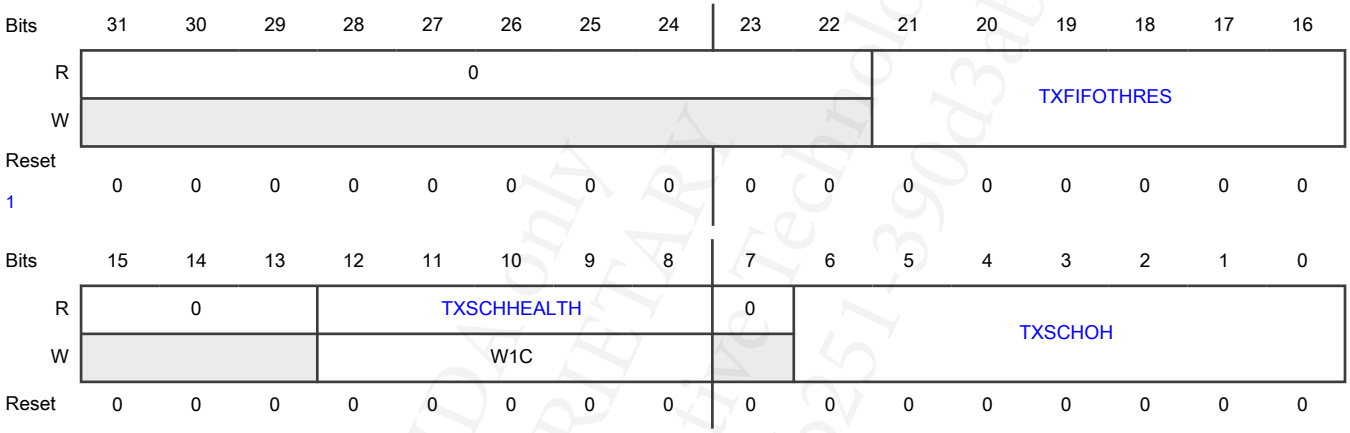
$T_s = T_0 + T_1$

T_p = Total Packet Time (fetch and send) packet

$T_p = T_{ff} + T_0 + T_1$

Upon discovery of a transmit (OUT/SETUP) packet in the data structures, the host controller checks to ensure T_p remains before the end of the (micro)frame. If so, it proceeds to pre-fill the TX FIFO. If at any time during the pre-fill operation the time remaining the [micro]frame is < T_s, then the packet attempt ceases and the packet is tried at a later time. Although this is not an error condition and the host controller eventually recovers, a mark is made to the scheduler health counter to note the occurrence of a "back-off" event. When a back-off event is detected, the partial packet fetched may need to be discarded from the latency buffer to make room for periodic traffic that begins after the next SOF. Too many back-off events can waste bandwidth and power on the system bus and thus must be minimized (not necessarily eliminated). Back-offs can be minimized with use of the TSCHHEALTH (T_{ff}) described below.

Diagram



1. Reset value is implementation-dependent.

Fields

Field	Function
31-22 —	Reserved
21-16 TXFIFOTHRES	FIFO Burst Threshold Controls the number of data bursts that are posted to the TX latency FIFO in host mode before the packet begins on the bus. The minimum value is two, and this value must be as small as possible to maximize USB performance. A higher value can be used in systems with unpredictable latency and/or insufficient bandwidth for which the FIFO may underrun because the data transferred from the latency FIFO to the USB occurs before it can be refreshed from system memory. This value is ignored if USBMODE[SDIS] is 1.
15-13 —	Reserved
12-8	Scheduler Health Counter

Table continues on the next page...

Table continued from the previous page...

Field	Function
TXSCHHEALTH	Increments when the host controller fails to fill the TX latency FIFO to the level programmed by TXFIFOTHRES before running out of time to send the packet before the next Start-Of-Frame. This health counter measures the number of times this occurs to provide feedback to selecting a proper TXSCHOH. The maximum value for this counter is 31, and writing to this register clears the counter.
7 —	Reserved
6-0 TXSCHOH	Scheduler Overhead This field adds a fixed offset to the schedule time estimator described above as T_{ff}. As an approximation, the value chosen for this register must limit the number of back-off events captured in the TXSCHHEALTH to less than 10 per second in a high-usage bus. Choosing a value that is too high for this register is not desired as it can needlessly reduce USB utilization. The time unit represented in this register is 1.267 µs when a device is connected in HS mode. The time unit represented in this register is 6.333 µs when a device is connected in LS/FS mode.

55.4.1.25 IC_USB Enable and Voltage Negotiation (IC_USB)

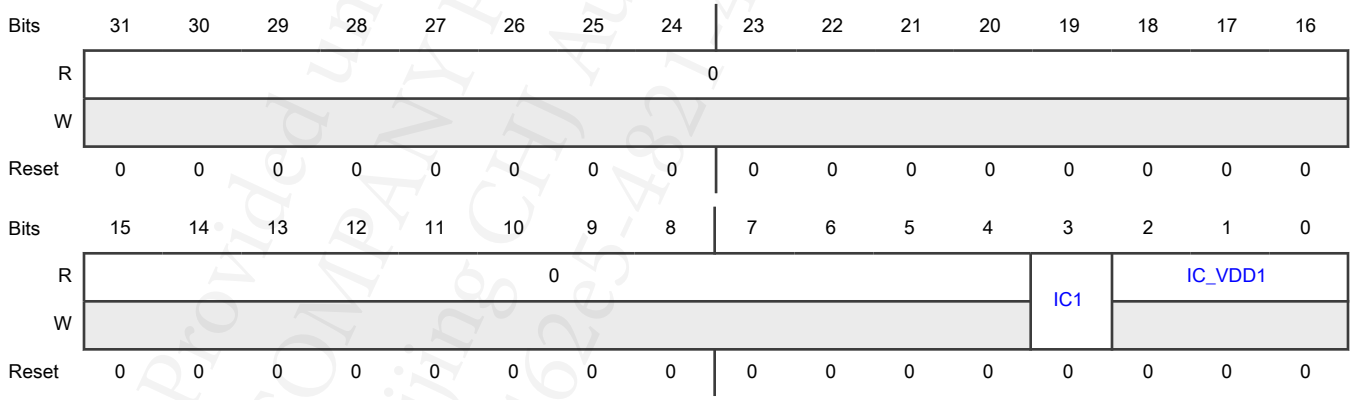
Offset

Register	Offset
IC_USB	16Ch

Function

Enables and controls the IC-USB FS or LS transceiver.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e25-4821-48a8-b251-390d3e24c1

Fields

Field	Function
31-4 —	Reserved
3 IC1	Inter-Chip Transceiver Enable Enables the inter-chip transceiver. To enable the IC-USB interface, the PTS bits in the PORTSC1 register also need to be set correspondingly. See PORTSC1. **NOTE** If IC1 is enabled, IOMUX must set the DP/DM pad into loopback mode.
2-0 IC_VDD1	Indicates which voltage is being supplied to the peripheral. The field is set to 000b in the case of a device controller. 000b - No voltage 001b - 1.0 V 010b - 1.2 V 011b - 1.5 V 100b - 1.8 V 101b - 3.0 V 110b - Reserved 111b - Reserved

55.4.1.26 ULPI Register Access (ULPI_VIEWPORT)

Offset

Register	Offset
ULPI_VIEWPORT	170h

Function

Provides indirect access to the ULPI PHY register set. Although the controller modules perform access to the ULPI PHY register set, there may be circumstances for which you need direct access.

CAUTION

Writes to the ULPI through the ULPI viewport can substantially harm standard USB operations. No usage model has been defined in which you should need to execute writes directly to the ULPI. Executing read operations though the ULPI viewport must have no harmful side effects to standard USB operations. Also, if the ULPI interface is not enabled, this register is always read-cleared.

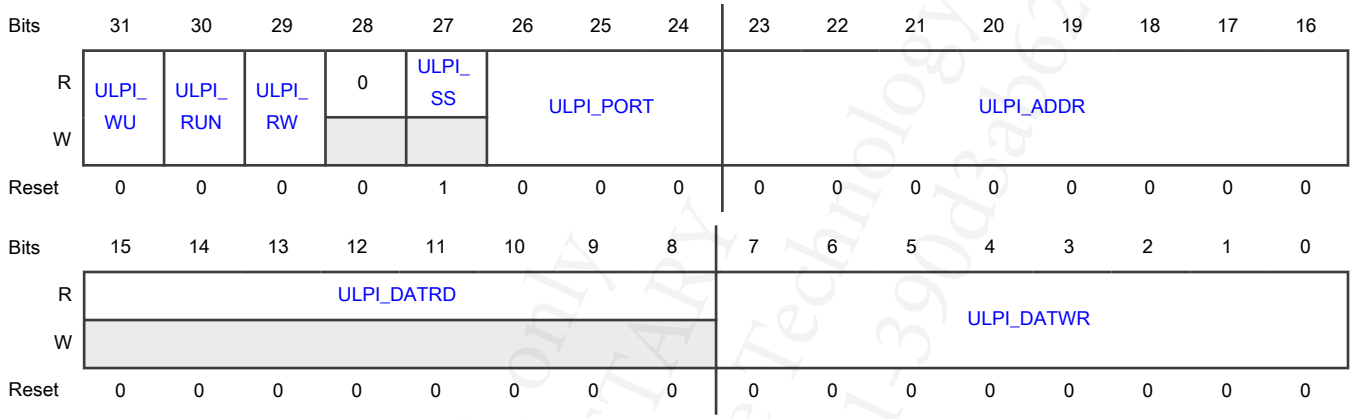
There are two operations that can be performed with the ULPI viewport: wake-up and read/write. The wake-up operation is used to put the ULPI interface into normal operation mode and re-enable the clock if necessary. A wake-up operation is required before accessing the registers when the ULPI interface is operating in low-power mode, serial mode, or carkit mode. The ULPI state can be determined by reading the sync state field (ULPI_VIEWPORT[ULPI_SS]). If this field is 1, then the ULPI

interface is running in normal operating mode and can accept read/write operations. If ULPI_VIEWPORT[ULPI_SS] becomes 0, then read/write operations are not executed. Undefined behavior results if a read or write operation is performed when ULPI_VIEWPORT[ULPI_SS] becomes 0. To execute a wake-up operation, you write all 32-bits of the ULPI VIEWPORT where ULPI_VIEWPORT[ULPI_PORT] is constructed appropriately, the ULPI_WU field is set to 1, and ULPI_VIEWPORT[ULPI_RUN] is cleared to 0. You must poll this register until the ULPI_WU field is cleared to 0 in order for the operation to complete.

To execute a read or write operation, you must write to all 32-bits of this register, where ULPI_DATWR, ULPI_ADDR, ULPI_PORT, and ULPI_RW fields are constructed appropriately and the ULPI_RUN field is 1. Poll the register until ULPI_RUN becomes 0 in order for the operation to complete. For read operations, ULPI_DATRD is valid after ULPI_RUN becomes 0.

The polling method above can be replaced with interrupts using the ULPI interrupt defined in USB Status (USBSTS) and Interrupt Enable (USBINTR). When a wake-up or read/write operation completes, the ULPI interrupt is set.

Diagram



Fields

Field	Function
31 ULPI_WU	ULPI Wake-Up Writing a 1 to this field begins the wake-up operation. This field is automatically cleared to 0 after the wake-up is complete. After this field is set to 1, it cannot be cleared to 0 by you. **NOTE** The driver must never execute a wake-up and a read/write operation at the same time.
30 ULPI_RUN	ULPI Run Writing this field to 1 begins a read/write operation. This field is automatically cleared to 0 after the read/write is complete. After this field is set to 1, it cannot be cleared to 0 by you. **NOTE** The driver must never execute a wake-up and a read/write operation at the same time.
29 ULPI_RW	ULPI Read/Write Selects between running a read or write operation to the ULPI. 0b - Read 1b - Write

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 —	Reserved
27 ULPI_SS	ULPI Sync State Represents the state of the ULPI interface. Before reading this field, the ULPI_PORT field must be set accordingly if used with the multi-port host. Otherwise, this field must always remain 0. 0b - Any other state (that is, carkit, serial, low-power) 1b - Normal sync state
26-24 ULPI_PORT	ULPI Port Number For wake-up or read/write operations, this value selects the port number to which the ULPI PHY is attached. Valid values are 0 and 1.
23-16 ULPI_ADDR	ULPI Data Address When a read or write operation is commanded, the address of the operation is written to this field.
15-8 ULPI_DATRD	ULPI Data Read After a read operation completes, the result is placed in this field.
7-0 ULPI_DATWR	ULPI Data Write When a write operation is commanded, the data to be sent is written to this field.

55.4.1.27 Endpoint NAK (ENDPTNAK)

Offset

Register	Offset
ENDPTNAK	178h

Function

This status register indicates whether the device has sent NAK on the corresponding endpoint.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab6240f

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0										EPTN					
W											W1C					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										EPRN					
W											W1C					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-22 —	Reserved
21-16 EPTN	TX Endpoint NAK Each TX endpoint has 1 in this field. The field is 1 when the device sends a NAK handshake on a received IN token for the corresponding endpoint. Bit [N] - Endpoint #[N], where N is 0 to 5
15-6 —	Reserved
5-0 EPRN	RX Endpoint NAK Each RX endpoint has 1 in this field. The field is 1 when the device sends a NAK handshake on a received OUT or PING token for the corresponding endpoint. Bit [N] - Endpoint #[N], where N is 0 to 5

55.4.1.28 Endpoint NAK Enable (ENDPTNAKEN)

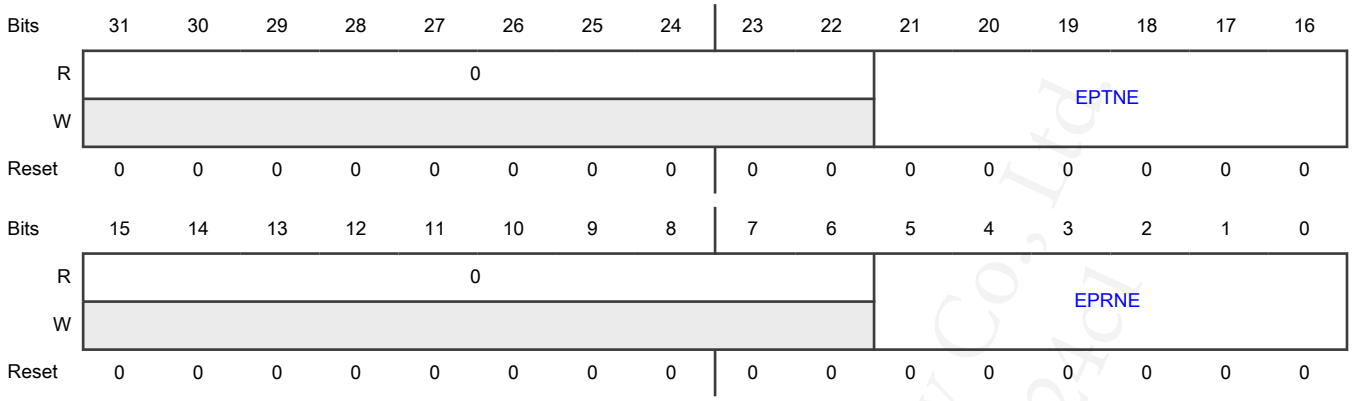
Offset

Register	Offset
ENDPTNAKEN	17Ch

Function

This configuration register is used to enable interrupts corresponding to NAK received on a particular endpoint.

Diagram



Fields

Field	Function
31-22 —	Reserved
21-16 EPTNE	TX Endpoint NAK Enable Each field is an enable field for the corresponding TX Endpoint NAK field. If this field is set to 1 and the corresponding TX Endpoint NAK field is set to 1, then the NAK Interrupt field is set to 1. Bit [N] - Endpoint #[N], where N is 0 to 5
15-6 —	Reserved
5-0 EPRNE	RX Endpoint NAK Enable Each field is an enable field for the corresponding RX Endpoint NAK field. If this field is set to 1 and the corresponding RX Endpoint NAK field is set to 1, then the NAK Interrupt field is set to 1. Bit [N] - Endpoint #[N], where N is 0 to 5

55.4.1.29 Configure Flag (CONFIGFLAG)

Offset

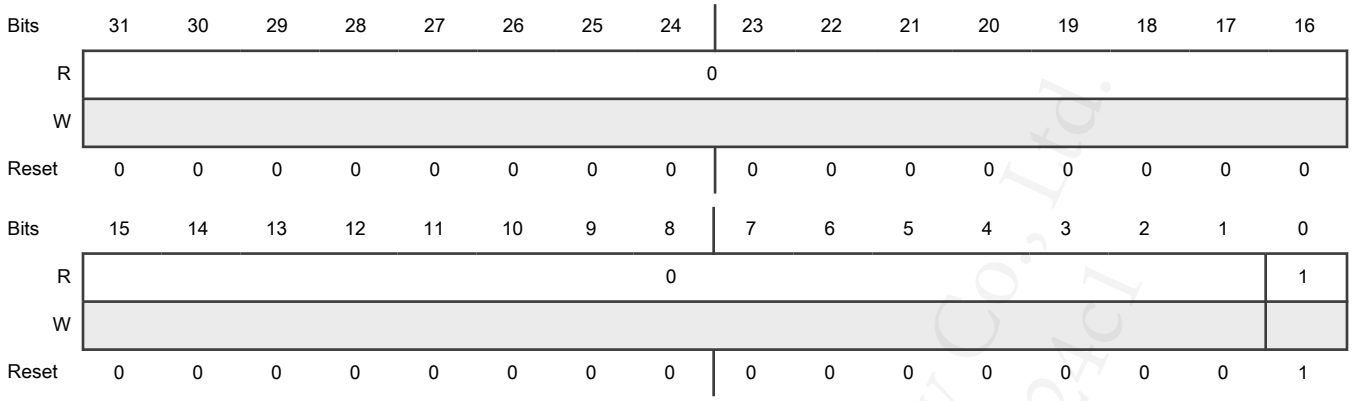
Register	Offset
CONFIGFLAG	180h

Function

This EHCI register is not used in this implementation. A read from this register returns a constant of 0000_0001h to indicate that all port routings default to this host controller.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-4c8-b251-390d8ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved (0000_0001h; all port routings default to this host)
0 —	Reserved (0000_0001h; all port routings default to this host)

55.4.1.30 Port Status and Control (PORTSC1)

Offset

Register	Offset
PORTSC1	184h

Function

Host controller

A host controller could implement one to eight of these port status and control registers. The number is determined by the N_PORTs bits in the HCSPARAMS register (see HCSPARAMS). You can read this parameter register to determine how many ports need service.

Both of the controller cores are single-port hosts, so there is only one port status and control register for each controller core.

This register is only reset by power-on reset or controller core reset. The initial conditions of a port are:

- No device connected
- Port disabled

If the port supports power control, the device controller state remains until port power is supplied (by you).

A device controller has only port register one (PORTSC1) and it does not support power control. Port control in Device mode is only used for status port reset, suspend, and current connect status. It is also used to initiate Test mode or force signaling, and allows you to put the PHY into low-power Suspend mode and disable the PHY clock.

Provided under NDA
COMPANY PROPRIETARY
Beijing CH Automotive Technology Co., Ltd.
c9462e5-4d1-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PTS1		STS	PTW	PSPD		PTS0	PFSC	PHCD	WKOC	WKDS	WKCN	PTC			
W																
Reset	1	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PIC		PO	PP	LS		HSP	PR	SUSP	FPR	OCC	OCA	PEC	PE	CSC	CCS
W											W1C		W1C		W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Fields

Field	Function
31-30 PTS1	Parallel Transceiver Select This field consists of PTS1 and PTS0. **NOTE** All USB port interface modes are listed in this field description, but not all are supported. For the feature details of each controller core, see Features. The behavior is unknown when an unsupported interface mode is selected. 000 UTMI/UTMI+ 001 Reserved 010 ULPI 011 Serial/USB 1.1 PHY/IC-USB (FS only) 100 HSIC 101 Reserved 110 Reserved 111 Reserved
29 STS	Serial Transceiver Select The serial interface engine can be used in combination with the UTMI+/ULPI physical interface to provide FS or LS signaling instead of the parallel interface signals. When this field is 1, the serial interface engine is used instead of the parallel interface signals. This field has no effect unless the PTS field is configured to select the UTMI+/ULPI interface. The Serial/USB1.1 PHY/IC-USB uses the serial interface engine for FS or LS signaling regardless of this field's value. 0b - Parallel interface signals selected 1b - Serial interface engine selected
28	Parallel Transceiver Width

Table continues on the next page...

Table continued from the previous page...

Field	Function
PTW	This field has no effect if serial interface engine is used. These register bits are implementation-dependent. 0b - Select the 8-bit UTMI interface [60MHz] 1b - Select the 16-bit UTMI interface [30MHz]
27-26 PSPD	Port Speed Indicates the speed at which the port is operating. 00b - FS 01b - LS 10b - HS 11b - Undefined
25 PTS0	PTS0 See the description in PTS1.
24 PFSC	Port Force FS Connect When this field is set to 1, the port is forced to only connect at FS, It disables the chirp sequence that allows the port to identify itself as FS. 0b - Normal operation 1b - Forced to FS
23 PHCD	PHY Low Power Suspend - Clock Disable When this field is 1, the PHY clock is disabled. Reading this field indicates the status of the PHY clock. **NOTE** The PHY clock cannot be disabled if it is being used as the system clock. In Device mode, the PHY can be put into low-power Suspend mode when the device is not running (USBCMD Run/Stop=0) or the host has signaled suspend (PORTSC1 SUSPEND=1). PHY Low Power Suspend is cleared automatically when the host initials resume. Before forcing a resume from the device, the DCD must clear this field. In Host mode, the PHY can be put into low-power Suspend mode when the downstream device has been put into Suspend mode or when no downstream device is connected. Low-power Suspend mode is completely under your control. 0b - Enable PHY clock 1b - Disable PHY clock
22 WKOC	Wake on Over-current Enable Writing this field to 1 enables the port to be sensitive to over-current conditions as wake-up events. This field is 0 if PORTSC1 is 0.

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 WKDS	Wake on Disconnect Enable Writing this field to 1 enables the port to be sensitive to device disconnects as wake-up events. This field is 0 if PORTSC1 is 0 or is in Device mode.
20 WKCN	Wake on Connect Enable Writing this field to a 1 enables the port to be sensitive to device connects as wake-up events. This field is 0 if PORTSC1 is 0 or is in Device mode.
19-16 PTC	Port Test Control The FORCE_ENABLE_FS and FORCE ENABLE_LS are extensions to the test mode support specified in the EHCI specification. Writing the PTC field to any of the FORCE_ENABLE_{HS, FS, or LS} values forces the port into the connected and enabled state at the selected speed. Writing the PTC field back to TEST_MODE_DISABLE allows the port state machines to progress normally from that point. **NOTE** LS operations are not supported as a peripheral device. Any value other than 0h indicates that the port is operating in Test mode. 0000b - TEST_MODE_DISABLE 0001b - J_STATE 0010b - K_STATE 0011b - SE0 (host) / NAK (device) 0100b - Packet 0101b - FORCE_ENABLE_HS 0110b - FORCE_ENABLE_FS 0111b - FORCE_ENABLE_LS 1000-1111b - Reserved
15-14 PIC	Port Indicator Control Writing to this field has no effect if HCSPARAMS[PI] is 0. See "USB Specification Revision 2.0" for a description on how these fields are to be used. This field is 0 if port power is 0. Field-value encoding: 00b - Port indicators are off 01b - Amber 10b - Green 11b - Undefined
13	Port Owner

Table continues on the next page...

Table continued from the previous page...

Field	Function
PO	This field unconditionally goes to 0 when the configured field in Configure Flag (CONFIGFLAG) makes a 0 to 1 transition. This field unconditionally goes to 1 whenever the Configured field is 0. You use this field to release ownership of the port to a selected host controller (if the attached device is not an HS device). You write 1 to this field when the attached device is not an HS device. A 1 in this field means that an internal companion controller owns and controls the port. Port owner hand-off is not supported, therefore this field is always 0.
12 PP	Port Power The function of this field depends on the value of HCSPARAMS[PPC]. The behavior is as follows: PPC PP Operation 0 1b - Read Only. The host controller does not have port power control switches. Each port is hard-wired to power. 1 1b/0b - Read/Write. Host/OTG controller requires port power control switches. This field represents the current setting of the switch (0=off, 1=on). When power is not available on a port (that is, PP equals 0), the port is non-functional and does not report attaches, detaches, and so on. When an over-current condition is detected on a powered port and PPC is 1, the PP field in each affected port may be transitioned by the host controller driver from 1 to 0 (removing power from the port).
11-10 LS	Line Status Reflects the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines. In host mode, the use of linestate by the host controller driver is not necessary (unlike EHCI), because the port controller state machine and the port routing manage the connection of LS and FS. In device mode, the use of linestate by the DCD is not necessary. The encoding of the bits [11:10] are: 00b - SE0 01b - K-state 10b - J-state 11b - Undefined
9 HSP	HS Port When the field is 1, the host/device connected to the port is in HS mode; when this field is 0, the host/device connected to the port is not in HS mode. **NOTE** HSP is redundant with PSPD (bit 27, 26) but is retained for compatibility.
8 PR	Port Reset In Host mode, this field is R/W. When you write 1 to this field, the bus reset sequence begins as defined in USB Specification Revision 2.0. This field automatically changes to 0 after the reset sequence is completed. This behavior is different from

Table continues on the next page...

Table continued from the previous page...

Field	Function
	the EHCI in which the host controller driver is required to clear this field to 0 after the reset duration is timed in the driver. In Device mode, this field is read-only. Device reset from the USB bus is also indicated in USB Status (USBSTS). This field is 0 if PORTSC1[PP] is 0. 0b - Port is not in Reset 1b - Port is in Reset
7 SUSP	Suspend 1 - Port is in suspend state 0 - Port is not in suspend state In Host mode: Read/Write PORTSC1[PE] and PORTSC1[SUSP] of this register define the port states as follows: Bits [Port Enabled, Suspend] Port State 0x - Disabled 10 - Enabled 11 - Suspended When in the Suspend state, the downstream propagation of data is blocked on this port except for port reset. The blocking occurs at the end of the current transaction if a transaction was in progress when this field was written to 1. In the Suspend state, the port is sensitive to resume detection. The field status does not change until the port is suspended, and there may be a delay in suspending a port if there is a transaction currently in progress. The host controller unconditionally clears this field to 0 when you set the Force Port Resume field to 0. The host controller ignores a write of 0 to this field. If the host software sets this field to 1 when the port is not enabled (that is, the Port enabled field is 0), the results are undefined. This field is 0 if PORTSC1[PP] is 0 in Host mode. In Device mode: Read Only In Device mode, this field is a read-only status field.
6 FPR	Force Port Resume In Host mode: You set this field to 1 to drive resume signaling. The host controller sets this field to 1 if a J-to-K transition is detected while the port is in the Suspend state. When this field transitions to a 1 because a J-to-K transition is detected, the Port Change Detect field in the USBSTS register is also set to 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE This field automatically changes to 0 after the resume sequence is completed. This behavior is different from EHCI, in which the host controller driver is required to set this field to 0 after the resume duration is timed in the driver. When the Host controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (FS, 'K') is driven on the port as long as this field remains 1. This field remains 1 until the port has switched to HS idle. Writing 0 has no effect because the port controller times the resume operation to clear the field when the port control state switches to HS or FS idle. This field is 0 if PORTSC1[PP] is 0 in Host mode. This field is not EHCI-compatible. In Device mode: After the device has been in the Suspend state for 5 ms or more, you must set this field to 1 to drive resume signaling before clearing. The Device Controller sets this field to 1 if a J-to-K transition is detected while the port is in the Suspend state. The field is cleared to 0 when the device returns to normal operation. Also, when this field transitions to a 1 because a J-to-K transition detected, USBSTS[PCI] is also set to 1. Field values: 0b - No resume (K-state) detected/driven on port 1b - Resume detected/driven on port
5 OCC	Over-Current Change This field is set to 1 by hardware when there is a change to Over-current Active. You can clear this field to 0 by writing 1 to it.
4 OCA	Over-Current Active Automatically transitions from 1 to 0 when the over-current condition is removed. 0b - This port does not have an over-current condition 1b - This port presently has an over-current condition
3 PEC	Port Enable/Disable Change In Host mode: For the root hub, this field is set to 1 only when a port is disabled due to disconnect on the port or due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the USB Specification). You clear this field to 0 by writing 1 to it. This field is 0 if PORTSC1[PP] is 0. In Device mode: The device port is always enabled, so this field is always 0. 0b - No change 1b - Port enabled/disabled status has changed

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 PE	Port Enabled/Disabled In Host mode: Ports can only be enabled by the host controller as a part of the reset and enable. You cannot enable a port by writing 1 to this field. Ports can be disabled by a disconnect event or other fault condition, or by the host software. The field status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events. When the port is disabled (0), downstream propagation of data is blocked except for reset. This field is 0 if PORTSC1[PP] is 0 in host mode. In Device mode: The device port is always enabled, so this field is always 1. 0b - Disabled 1b - Enabled
1 CSC	Connect Status Change In Host mode: Indicates that a change has occurred in the port's Current Connect Status. The host/device controller sets this field for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, if the insertion status changes twice before the system software has cleared the changed condition, the hub hardware is setting an already-set field (that is, the field remains set). You clear this field to 0 by writing a 1 to it. This field is 0 if PORTSC1[PP] is 0 in Host mode. In Device mode: This field is undefined in device controller mode.
0 CCS	Current Connect Status In Host mode: 0 = No device is present 1 = Device is present on port This value reflects the current state of the port, and may not correspond directly to the event that caused the Connect Status Change field (bit 1) to be set to 1. This field is 0 if PORTSC1[PP] is 0 in Host mode. In Device mode: A value of 1 in this field indicates that the device successfully attached and is operating in either HS or FS as indicated by the HS Port field in this register. A 0 indicates that the device did not attach successfully or you forcibly disconnected it by writing a 0 to USBCMD[RS]. It does not identify the device being disconnected or suspended.

55.4.1.31 OTG Status Control (OTGSC)

Offset

Register	Offset
OTGSC	1A4h

Function

This register has four sections:

- OTG Interrupt enables (Read/Write)
- OTG Interrupt status (Read/Write to Clear)
- OTG Status inputs (Read Only)
- OTG Controls (Read/Write)

The status inputs are debounced using a 1 ms time constant. Values on the status inputs that do not persist for more than 1 ms do not cause an update of the status inputs (bits 14:0 of this register) or cause an OTG interrupt.

Also see USBMODE.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	DPIE	TIMIE	BSEIE	BSVIE	ASVIE	AVVIE	IDIE	0	DPIS	TIMIS	BSEIS	BSVIS	ASVIS	AVVIS	IDIS
W										W1C	W1C	W1C	W1C	W1C	W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	DPS	TIM_TOG	BSE	BSV	ASV	AVV	ID	HABA	HADP	IDPU	DP	OT	HAAR	VC	VD
W																
Reset	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 DPIE	Data Pulse Interrupt Enable
29 TIMIE	One-Millisecond Timer Interrupt Enable (1MsE)

Table continues on the next page...

Table continued from the previous page...

Field	Function
28 BSEIE	B Session End Interrupt Enable Writing 1 to this field enables the B session end interrupt.
27 BSVIE	B Session Valid Interrupt Enable Writing 1 to this field enables the B session valid interrupt.
26 ASVIE	A Session Valid Interrupt Enable Writing 1 to this field enables the A session valid interrupt.
25 AVVIE	A VBus Valid Interrupt Enable Writing 1 to this field enables the A VBus valid interrupt.
24 IDIE	USB ID Interrupt Enable Writing 1 to this field enables the USB ID interrupt.
23 —	Reserved
22 DPIS	Data Pulse Interrupt Status This field is set to 1 when data bus pulsing occurs on DP or DM. Data bus pulsing is only detected when USBMODE[CM] = 3 (host) and PORTSC1[PP] = 0. You must write a 1 to this field to clear it to 0.
21 TIMIS	One-Millisecond Timer Interrupt Status (1MsS) This field is set once every millisecond. You must write a 1 to this field to clear it to 0.
20 BSEIS	B Session End Interrupt Status This field is set to 1 when VBus has fallen below the B session end threshold. You must write a 1 to this field to clear it to 0.
19 BSVIS	B Session Valid Interrupt Status This field is set to 1 when VBus has either risen above or fallen below the B session valid threshold (0.8 VDC). You must write a 1 to this field to clear it to 0.
18 ASVIS	A Session Valid Interrupt Status This field is set to 1 when VBus has either risen above or fallen below the A session valid threshold (0.8 VDC). You must write a 1 to this field to clear it to 0.
17	A VBus Valid Interrupt Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
AVVIS	This field is set to 1 when VBus has either risen above or fallen below the VBus valid threshold (4.4 VDC) on an A device. You must write a 1 to this field to clear it to 0.
16 IDIS	USB ID Interrupt Status This field is set to 1 when a change on the ID input has been detected. You must write a 1 to this field to clear it to 0.
15 —	Reserved
14 DPS	Data Bus Pulsing Status 0b - Data bus pulsing is not being detected on the port 1b - Data bus pulsing is being detected on the port
13 TIM_TOG	One-Millisecond Timer Toggle This field toggles once per millisecond.
12 BSE	B Session End Indicates VBus is below the B session end threshold.
11 BSV	B Session Valid Indicates VBus is above the B session valid threshold.
10 ASV	A Session Valid Indicates VBus is above the A session valid threshold.
9 AVV	A VBus Valid Indicates VBus is above the A VBus valid threshold.
8 ID	USB ID 0b - A device 1b - B device
7 HABA	Hardware Assist B-Disconnect to A-connect. 0b - Disabled 1b - Enable automatic B-disconnect to A-connect sequence
6 HADP	HADP Hardware Assist Data. If set to 1, the hardware assist data pulsing sequence starts.
5	ID Pullup

Table continues on the next page...

Table continued from the previous page...

Field	Function
IDPU	Provides control over the ID pull-up resistor. When this field is 0, the ID input is not sampled. 0b - Pull-up resistor is off 1b - Pull-up resistor is on (default)
4 DP	Data Pulsing Writing 1 to this field causes the pullup on DP to be asserted for data pulsing during SRP.
3 OT	OTG Termination This field must be set when the OTG device is in device mode; this controls the pulldown on DM.
2 HAAR	Hardware Assist Auto-Reset 0b - Disabled 1b - Enable automatic reset after connect on host port
1 VC	VBUS Charge Writing 1 to this field causes the VBus line to be charged. This is used for VBus pulsing during SRP.
0 VD	VBUS Discharge Writing 1 to this field causes VBus to discharge through a resistor.

55.4.1.32 USB Device Mode (USBMODE)

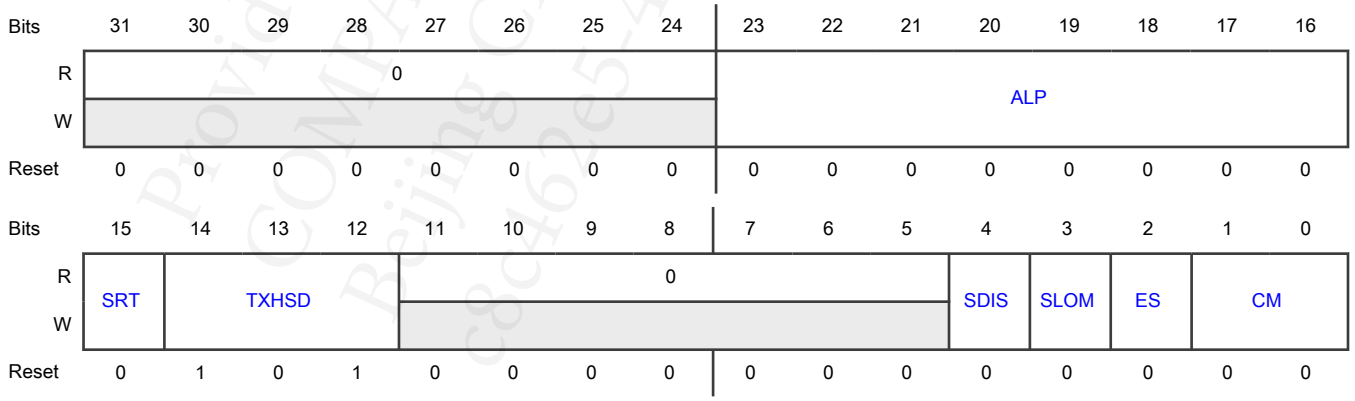
Offset

Register	Offset
USBMODE	1A8h

Function

Configures various features of the USB controller as given in the field descriptions below.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca246e54-821-48a8-b251-320d3ab6241

Fields

Field	Function
31-24 —	Reserved
23-16 ALP	Auto Low Power Includes one bit per port. To enable or disable this automation for a port, you must write the corresponding bit: 16 + port number - 1. When you enable this field, the PHY is put into low-power when in Suspend (PORTSCx[SUSP]), even if PORTSCx[PHCD] is not set to 1.
15 SRT	Shorten Reset Time When the controller is in Host mode, this field enables a bypass of the Chirp J/K reset handshake, saving 6-7 ms in simulation time for each reset sequence. This field should only be used for initial system integration simulations, and must always be cleared for normal operation.
14-12 TXHSD	Tx to Tx HS Delay
11-5 —	Reserved
4 SDIS	Stream Disable Mode Device mode: Writing 1 to this field disables double priming on both RX and TX for low bandwidth systems. This mode ensures that when the RX and TX buffers are sufficient to contain an entire packet that the standard double-buffering scheme is disabled to prevent overruns/underruns in bandwidth-limited systems. Note: In HS mode, all packets received are responded to with a NYET handshake when stream disable is active. Host mode: Writing 1 to this field ensures that overruns/underruns of the latency FIFO are eliminated for low bandwidth systems where the RX and TX buffers are sufficient to contain the entire packet. Enabling stream disable also has the effect of ensuring the TX latency is filled to capacity before the packet is launched onto the USB. **NOTE** Time duration to pre-fill the FIFO becomes significant when stream disable is active. See TXFILLTUNING to characterize the adjustments needed for the scheduler when using this feature. **NOTE** The use of this feature substantially limits overall USB performance.
3 SLOM	Setup Lockout Mode In Device mode, controls the behavior of the setup lock mechanism. 0b - Setup lockouts on (default) 1b - Setup lockouts off (DCD requires use of Setup Data Buffer Tripwire in USBCMD)

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 ES	Endian Select Controls the byte alignment of the transfer buffers to match the host microprocessor. The fields in the microprocessor interface and the data structures are unaffected by the value of this field because they are based on the 32-bit word. Bit Meaning 0b - Little Endian 1b - Big Endian
1-0 CM	Controller Mode Controller mode is defaulted to the proper mode for host-only and device-only implementations. For those designs that contain both host and device capability, the controller defaults to an idle state and must be initialized to the desired operating mode after reset. For combination host/device controllers, this register can only be written once after reset. If it is necessary to switch modes, you must reset the controller by writing to USBCMD[RST] before reprogramming this register. 00b - Idle 01b - Reserved 10b - Device Controller 11b - Host Controller

55.4.1.33 Endpoint Setup Status (ENDPTSETUPSTAT)

Offset

Register	Offset
ENDPTSETUPSTAT	1ACh

Function

For every setup transaction that is received, a corresponding bit in this register is set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ENDPTSETUPSTAT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e54-821-48a8-b251-390a5b624c1

Fields

Field	Function
31-16 —	Reserved
15-0 ENDPTSETUPSTAT	Setup Endpoint Status You must clear or acknowledge the setup transfer by writing a 1 to a respective bit after it has read the setup data from QH. The response to a setup packet with regard to order of operations and total response time is crucial to limiting bus timeouts while the setup lockout mechanism is engaged. See the Device Operational Model for details. This register is only used in Device mode.

55.4.1.34 Endpoint Initialization (ENDPTPRIME)

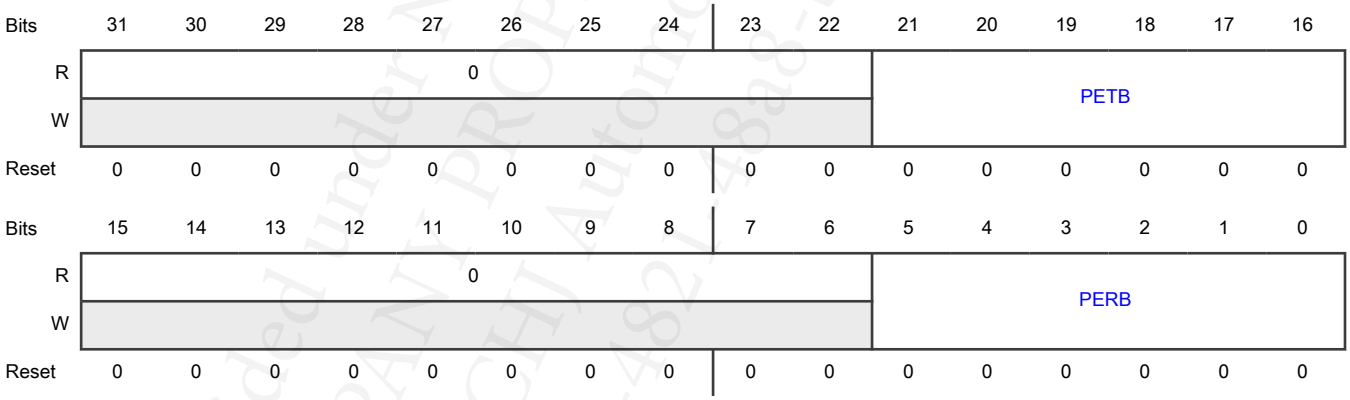
Offset

Register	Offset
ENDPTPRIME	1B0h

Function

Only used in Device mode. Used to prime endpoints corresponding to transmit and receive.

Diagram



Fields

Field	Function
31-22 —	Reserved
21-16	Prime Endpoint Transmit Buffer

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4821-48a-b251-390d3ab624

Table continued from the previous page...

Field	Function
PETB	For each endpoint, a corresponding bit is used to request that a buffer prepare for a transmit operation in order to respond to a USB IN/INTERRUPT transaction. You must set the corresponding bit to 1 when posting a new TD to an endpoint. The hardware automatically uses this bit to begin parsing for a new TD from the QH and prepare a transmit buffer. The hardware clears this bit to 0 when the associated endpoint(s) is (are) successfully primed. **NOTE** These bits are momentarily set to 1 by the hardware during hardware re-priming operations when a dTD is retired, and the device queue head (dQH) is updated. PETB[N] - Endpoint #N, N is in 0..5
15-6 —	Reserved
5-0 PERB	Prime Endpoint Receive Buffer For each endpoint, a corresponding bit is used to request that a buffer prepare for a receive operation that occurs when a USB host initiates a USB OUT transaction. You must set the corresponding bit to 1 whenever posting a new TD to an endpoint. The hardware automatically uses this bit to begin parsing for a new TD from the QH and prepare a receive buffer. The hardware clears this bit to 0 when the associated endpoint(s) is (are) successfully primed. **NOTE** These bits are momentarily set to 1 by the hardware during hardware re-priming operations when a dTD is retired, and the dQH is updated. PERB[N] - Endpoint #N, N is in 0..5

55.4.1.35 Endpoint De-Initialize (ENDPTFLUSH)

Offset

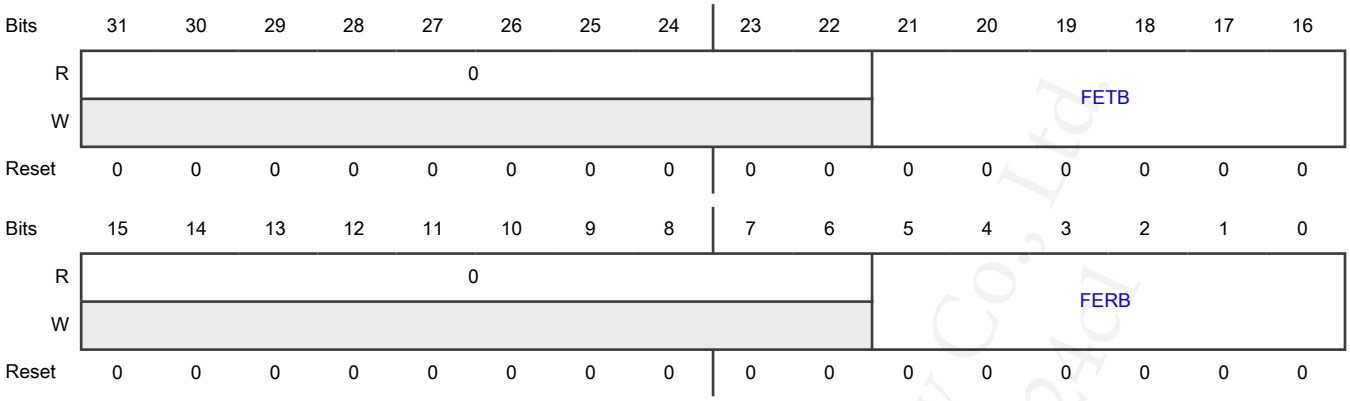
Register	Offset
ENDPTFLUSH	1B4h

Function

Only used in Device mode. Writing 1 to a field(s) in this register causes the associated endpoint(s) to clear any primed buffers.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-fdd3ab624c1

Diagram



Fields

Field	Function
31-22 —	Reserved
21-16 FETB	Flush Endpoint Transmit Buffer Writing 1 to a field(s) in this register causes the associated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer continues until completion. The hardware clears this register to 0 after the endpoint flush operation is successful. FETB[N] - Endpoint #N, N is in 0..5
15-6 —	Reserved
5-0 FERB	Flush Endpoint Receive Buffer Writing 1 to a field(s) in this register causes the associated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer continues until completion. The hardware clears this register to 0 after the endpoint flush operation is successful. FERB[N] - Endpoint #N, N is in 0..5

55.4.1.36 Endpoint Status (ENDPTSTAT)

Offset

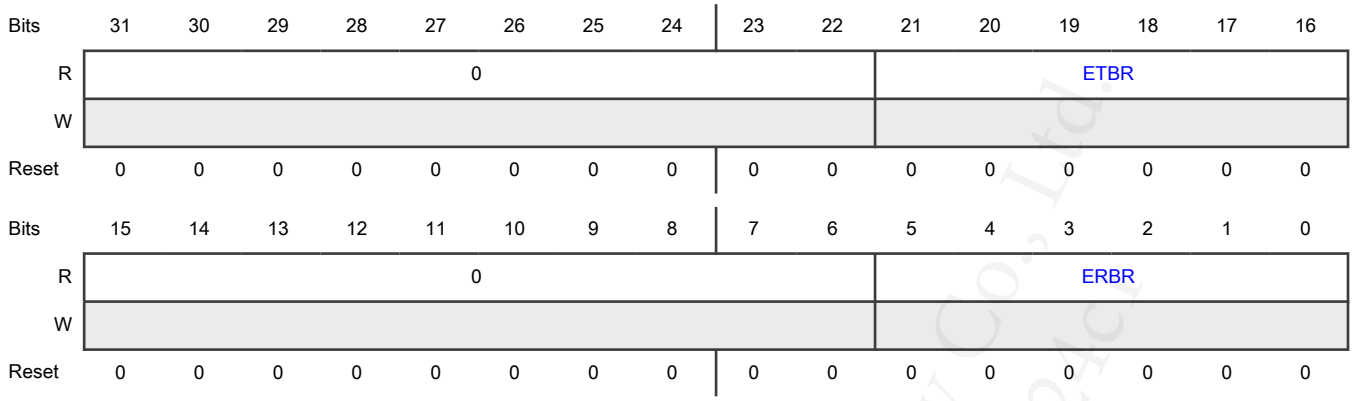
Register	Offset
ENDPTSTAT	1B8h

Function

Only used in Device mode. One bit for each endpoint indicates the readiness status of the respective endpoint buffer.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-8a8-b25-390d8ab624c1

Diagram



Fields

Field	Function
31-22 —	Reserved
21-16 ETBR	Endpoint Transmit Buffer Ready One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set to 1 by the hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. There is always a delay between setting a bit to 1 in the ENDPTPRIME register and endpoint indicating ready. This delay time varies based on the current USB traffic and the number of bits set in the ENDPTPRIME register. Buffer ready is cleared by USB reset, by the USB DMA system, or through the ENDPTPRIME register. **NOTE** These bits are momentarily cleared by hardware during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated. ETBR[N] - Endpoint #N, N is in 0..5
15-6 —	Reserved
5-0 ERBR	Endpoint Receive Buffer Ready One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set to 1 by the hardware as a response to receiving a command from a corresponding bit in the ENDPRIME register. There is always a delay between setting a bit to 1 in the ENDPRIME register and endpoint indicating ready. This delay time varies based on the current USB traffic and the number of bits set to 1 in the ENDPRIME register. Buffer ready is cleared to 0 by USB reset, by the USB DMA system, or through the ENDPDINT register. **NOTE** These bits are momentarily cleared to 0 by hardware during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated. ERBR[N] - Endpoint #N, N is in 0..5

55.4.1.37 Endpoint Complete (ENDPTCOMPLETE)

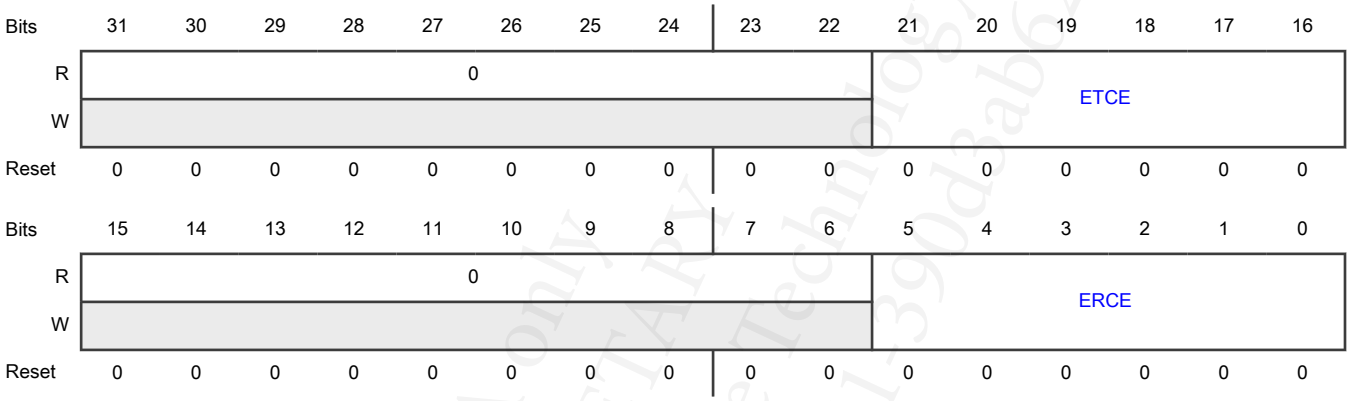
Offset

Register	Offset
ENDPTCOMPLETE	1BCh

Function

Only used in Device mode. Each bit indicates a transmit (IN/INTERRUPT) or receive (OUT/SETUP) event occurred.

Diagram



Fields

Field	Function
31-22 —	Reserved
21-16 ETCE	Endpoint Transmit Complete Event Each bit indicates a transmit event (IN/INTERRUPT) occurred and you must read the corresponding endpoint queue to determine the endpoint status. If the corresponding IOC field is set in the TD, then this bit is set simultaneously with the USBSTS[UI]. Writing a 1 clears the corresponding bit in this register to 0. ETCE[N] - Endpoint #N, N is in 0..5
15-6 —	Reserved
5-0 ERCE	Endpoint Receive Complete Event Each bit indicates a received event (OUT/SETUP) occurred and you must read the corresponding endpoint queue to determine the transfer status. If the corresponding IOC field is set in the TD, then this bit is set simultaneously with the USBSTS[UI]. Writing a 1 clears the corresponding bit in this register to 0. ERCE[N] - Endpoint #N, N is in 0..5

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c12e5-482-48a8-b251390d3ab6240

55.4.1.38 Endpoint Control 0 (ENDPTCTRL0)

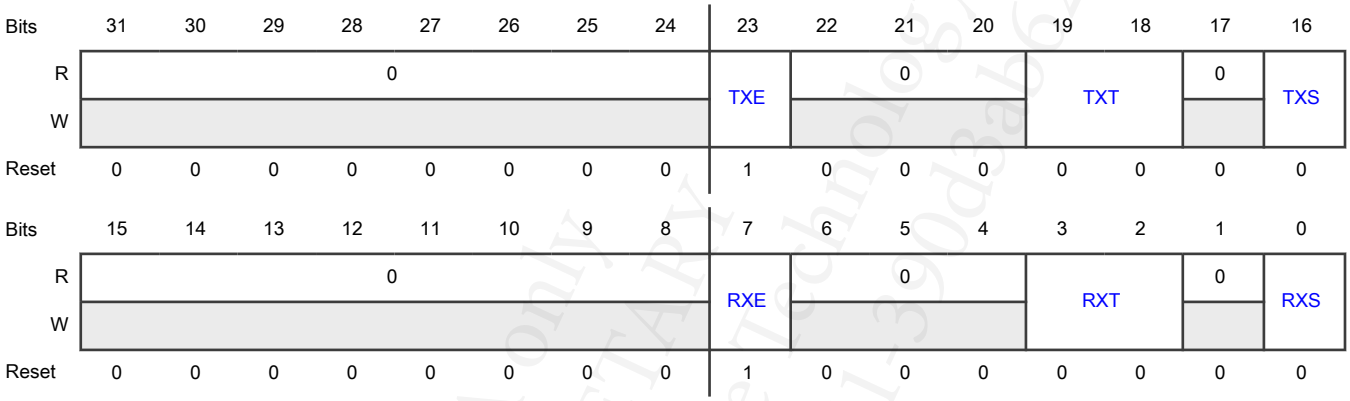
Offset

Register	Offset
ENDPTCTRL0	1C0h

Function

Every chip implements Endpoint 0 as a control endpoint. This register contains endpoint enable, type, and stall control fields.

Diagram



Fields

Field	Function
31-24 —	Reserved
23 TXE	TX Endpoint Enable 1b - Enabled (Endpoint 0 is always enabled)
22-20 —	Reserved
19-18 TXT	TX Endpoint Type Endpoint 0 is fixed as a Control End Point. 00b - Control
17 —	Reserved
16 TXS	TX Endpoint Stall

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e-482f-48a8-b251-390d3ab6240d

Table continued from the previous page...

Field	Function
	You can write a 1 to this field to force the endpoint to return a STALL handshake to the host. It continues returning STALL until you clear the field to 0 or it is automatically cleared to 0 upon receipt of a new SETUP request. After receiving a SETUP request, this field continues to be cleared by hardware until the associated Endpoint Setup Status (ENDPTSETUPSTAT) bit is cleared to 0. Note: There is a slight delay (50 clocks maximum) between the Endpoint Setup Status (ENDPTSETUPSTAT) bit being cleared to 0 and hardware continuing to clear this field. In most systems, it is unlikely the DCD software will observe this delay. However, should the DCD observe that the STALL field is not set after writing a 1 to it, then follow this procedure: Continually write this STALL field until it is set or until a new SETUP has been received by checking the associated Endpoint Setup Status (ENDPTSETUPSTAT) bit. 0b - Endpoint okay 1b - Endpoint stalled
15-8 —	Reserved
7 RXE	RX Endpoint Enable 1b - Enabled (Endpoint 0 is always enabled)
6-4 —	Reserved
3-2 RXT	RX Endpoint Type Endpoint 0 is fixed as a Control endpoint. 00b - Control
1 —	Reserved
0 RXS	RX Endpoint Stall You can write a 1 to this field to force the endpoint to return a STALL handshake to the host. It continues returning STALL until you clear the field to 0 or it is automatically cleared to 0 upon receipt of a new SETUP request. 0b - Endpoint is okay 1b - Endpoint is stalled

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHN Automotive Technology Co., Ltd.
c8c462e5-4871-48a8-b251-390d3ab62...

55.4.1.39 Endpoint Control (ENDPTCTRL1 - ENDPTCTRL5)

Offset

Register	Offset
ENDPTCTRL1	1C4h
ENDPTCTRL2	1C8h
ENDPTCTRL3	1CCh
ENDPTCTRL4	1D0h
ENDPTCTRL5	1D4h

Function

There is an ENDPTCTRLn register for each endpoint in a device.

NOTE

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled, then the unused direction type must be changed from the default control type to any other type (for example, bulk type). Leaving an unconfigured endpoint control causes undefined behavior for the data PID tracking on the active endpoint/direction.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								TXE	TXR	TXI	0	TXT		TXD	TXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RXE	RXR	RXI	0	RXT		RXD	RXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23 TXE	TX Endpoint Enable An endpoint must only be enabled after it has been configured. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 TXR	TX Data Toggle Reset Write 1 - Reset PID Sequence Whenever a configuration event is received for this endpoint, you must write a 1 to this field in order to synchronize the data PIDs between the host and the device.
21 TXI	TX Data Toggle Inhibit Only used for test and must always be written as 0. Writing a 1 to this field causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet. 0b - PID sequencing enabled 1b - PID sequencing disabled
20 —	Reserved
19-18 TXT	TX Endpoint Type 00b - Control 01b - Isochronous 10b - Bulk 11b - Interrupt
17 TXD	TX Endpoint Data Source 0b - 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] (must always be written as 0)
16 TXS	TX Endpoint Stall This field is automatically set to 1 upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared to 0 automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint. You can write a 1 to this field to force the endpoint to return a STALL handshake to the Host. It continues to return STALL until you either clear this field to 0 or it is automatically cleared to 0 as above. **NOTE** For a CONTROL type endpoint, there is a slight delay (50 clocks maximum) between the Endpoint Setup Status (ENDPTSETUPSTAT) bit being cleared to 0 and the hardware continuing to clear this field to 0. In most systems, it is unlikely the DCD software will observe this delay. Take care that the STALL field is not set to 1 immediately after writing a 1 to it. Follow this procedure: Continually write this STALL field until it is set to 1 or until a new SETUP has been received by checking the associated Endpoint Setup Status (ENDPTSETUPSTAT) bit. 0b - Endpoint is okay 1b - Endpoint is stalled
15-8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7 RXE	RX Endpoint Enable An Endpoint must only be enabled after it has been configured. 0b - Disabled 1b - Enabled
6 RXR	RX Data Toggle Reset Whenever a configuration event is received for this Endpoint, you must write a 1 to this field in order to synchronize the data PIDs between the host and the device. 1b - Reset PID Sequence
5 RXI	RX Data Toggle Inhibit Only used for test and must always be written as 0. Writing a 1 to this field causes this endpoint to ignore the data toggle sequence and always accept data packets regardless of their data PID. 0b - Disabled 1b - Enabled
4 —	Reserved
3-2 RXT	RX Endpoint Type 00b - Control 01b - Isochronous 10b - Bulk 11b - Reserved
1 RXD	RX Endpoint Data Sink This field must always be written as 0. 0b - Dual Port Memory Buffer/DMA Engine
0 RXS	RX Endpoint Stall This field is set to 1 automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared to 0 automatically upon receipt a SETUP request if this Endpoint is configured as a Control Endpoint, You can write a 1 to this field to force the endpoint to return a STALL handshake to the host. It continues to return STALL until you either clear this field to 0 or it is automatically cleared to 0 as above. After receiving a SETUP request, this field continues to be cleared to 0 by hardware until the associated Endpoint Setup Status (ENDPTSETUPSTAT) bit is cleared to 0. Note: There is a slight delay (50 clocks maximum) between the Endpoint Setup Status (ENDPTSETUPSTAT) bit being cleared to 0 and hardware continuing to clear this bit to 0. In most systems, it is unlikely the DCD software will observe this delay.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	However, should the DCD observe that the STALL field is not set after writing a 1 to it, then follow this procedure: Continually write this STALL field until it is set to 1 or until a new SETUP has been received by checking the associated Endpoint Setup Status (ENDPTSETUPSTAT) bit. 0b - Endpoint is okay 1b - Endpoint is stalled

55.4.2 Non-core register descriptions

55.4.2.1 UOTGNC memory map

UOTGNC base address: 4406_4200h

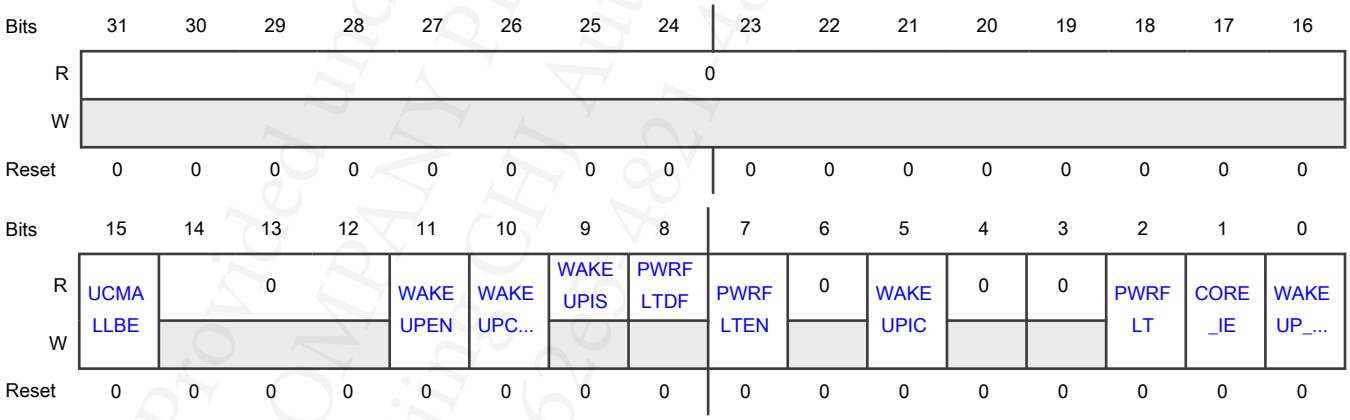
Offset	Register	Width (In bits)	Access	Reset value
0h	Control register (CR)	32	RW	0000_0000h

55.4.2.2 Control register (CR)

Offset

Register	Offset
CR	0h

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e6e-4821-48a8-b251-39...ab624c1

Fields

Field	Function
31-16 —	Reserved
15 UCMALLBE	USB Core Master All Byte Enable This field enables assertion of all bytes in all transactions, that is, no transaction is corrected. 0b - No change on line levels. 1b - Assert all byte enables.
14-12 —	Reserved
11 WAKEUPEN	User Wake-Up Enable This field enables or disables user wake-up. 0b - Disable 1b - Enable
10 WAKEUPCTRL	User Wake-Up Control You request wake-up using this field. 0b - No wake-up request 1b - Wake-up request is received
9 WAKEUPIS	USB Wake-Up Interrupt Status This field indicates reception of a wake-up interrupt on the USB port. You clear this field by writing 1 to the CR[WAKEUPIC] field. 0b - Wake-up interrupt received 1b - Wake-up interrupt is not received
8 PWRFLTDF	Vbus Power Fault Status This field indicates the status of Power fault. Default behavior of this field (that is, when CR[PWRFLT]=0): 0b - No power fault 1b - Power fault detected When polarity of Power fault is changed (that is, when CR[PWRFLT]=1), behavior of this field changes to: 0b - Power fault detected 1b - No power fault
7 PWRFLTEN	Enable Vbus Power Fault Detection

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field enables detection of Vbus power fault (CR[PWRFLT]). 0b - Disable 1b - Enable
6 —	Reserved
5 WAKEUPIC	Clear USB Wake-Up Interrupt 0b - Clears this field (CR[WAKEUPIC]) 1b - Clears the USB Wake-Up Interrupt Status field (CR[WAKEUPIS])
4 —	Reserved
3 —	Reserved
2 PWRFLT	Vbus Power Fault Polarity Control This field controls the polarity of power fault. Write to this field to reverse the polarity. See CR[PWRFLTDF] to learn impact of writing to this field. 1b - Reverse the polarity of Power fault
1 CORE_IE	Enable Interrupt from Core This field enables or disables interrupts from the core. 0b - Disable 1b - Enable
0 WAKEUP_IE	USB Wake-Up Interrupt Enable This field enables or disables the wake-up interrupt. You must disable this field after receiving a wake-up interrupt and enable it again prior to going into Suspend mode. 0b - Disable 1b - Enable

55.5 Functional description

These sections describe the functionality of the various building blocks of the USB.

55.5.1 USB dual role device/host controller

The OTG controller is an instantiation of EHCI-compatible USB core that supports HS, FS, and LS operations.

The USBOTG core USB signals do not connect directly to the OTG I/O pins. Instead, the signals pass through the Serial and ULPI Mux to allow for additional functionality on the OTG port. The OTG controller is configured for operation with a parallel

ULPI transceiver (HS, FS, or LS). The selection between transceiver types is software programmable. The block defaults to Serial mode.

The USB controller core supports HS, FS, and LS operations in Host mode, and HS and FS operations in Device mode.

55.5.1.1 Host mode

The controller supports the direct connection of a HS, FS, or LS device (without an external hub). Although there is no separate transaction translator block in the system, the transaction translator function, normally associated with a USB 2.0 HS hub, is implemented within the DMA and protocol engine blocks to support connection to FS and LS devices.

55.5.1.2 Peripheral (Device) mode

- Up to six bidirectional endpoints
- HS or FS operation
- Remote wake-up capable

55.5.2 System interface

The system interface block contains all the control and status registers to allow a core to interface with the module. These registers allow the processor to control the configuration and ascertain the capabilities of the module, and they control the module's operation.

55.5.3 DMA engine

This module contains a local DMA engine. It is responsible for moving all of the data transferred over the USB between the module and system memory.

The DMA controllers must access control information and packet data from system memory. Control information is contained in linked list-based queue structures. The DMA controllers have state machines that are able to parse all of the data structures defined in the EHCI specification. In Host mode, the data structures are EHCI compliant and represent queues of transfers performed by the host controller, including the split-transaction requests that allow an EHCI controller to direct packets to FS and LS devices.

55.5.4 FIFO RAM controller

The FIFO RAM controller is used for context information and to control FIFOs between the protocol engine and the DMA controller. These FIFOs decouple the system processor/memory bus requests from the extremely tight timing required by USB.

In Host mode, a single data channel is maintained in each direction through the buffer memory, and the USB host modules use 1024-byte transmit buffers and 1024-byte receive buffers.

55.5.5 USB power control block

The USB controller supports suspend and wake-up functionality. The power control block allows for placing the transceiver in low-power mode when USB bus is IDLE, and supports local and remote wake-up to bring the transceiver out of low-power mode when needed. Additionally, the power control block can wake-up the platform from Sleep mode by generating an interrupt.

55.5.5.1 Entering low-power Suspend mode

This mode is always entered when the controlling driver software sets the appropriate field in the USB core register, Port Status and Control (PORTSC1). After the controller is suspended, the clocks to the USB can be stopped.

55.6 Wake-up events

The power control block monitors the USB bus when the USB core is in the suspend state. Depending on whether the core is in Host or Device mode, a number of wake-up conditions are monitored. Upon detection of a wake-up condition, an interrupt

(asynchronous) can be generated to the platform if the value of the related wake-up interrupt enable field is 1. This interrupt also reactivates the platform clocks if they were stopped during the suspend.

55.6.1 Host mode events

The host controller wakes up on the following events:

- Remote wake-up request

A peripheral can request the host to reactivate the bus by driving wake-up signaling on the DM/DP lines. The power control block sends a wake-up request to the USB core when J-K transition on DM/DP line is detected.

- Wake-up on over-current

If wake-up on over-current is enabled (the value of PORTSC1[WKOC] is 1), the power control block sends a wake-up request to the USB core when an over-current event is detected.

- Wake-up on disconnect

If wake-up on disconnection is enabled (the value of PORTSC1[WKDS] is 1), the power control block sends a wake-up request to the USB core when a disconnection event is detected (J-SE0/K-SE0 transition on DM/DP line).

- Wake-up on connect

If wake-up on connection is enabled (the value of PORTSC1[WKCN] is 1), the power control sub-block sends a wake-up request to the USB core when a connection event is detected (SE0-J/SE0-K transition on DM/DP line).

55.6.2 Device mode events

When the OTG controller is configured for peripheral operation, the power control block sends a wake-up request to the USB core when any non-idle state is detected on the USB bus.

55.7 Interrupts

55.7.1 USB core interrupts

Each USB core uses one dedicated vector in the interrupt table. The vector number associated with each core can be found in the interrupt section.

With the exception of the wake-up interrupts, all of the interrupt sources are controlled in the USB cores.

55.7.2 USB wake-up interrupts

Each USB core has an associated wake-up interrupt. The wake-up interrupts are generated outside of the USB controller cores, but each one uses the same vector as the corresponding USB controller core's interrupt. These interrupts are generated by the power control blocks, which run on the 32 KHz standby clock. The wake-up interrupt is designed to work even when the USB and platform clocks are disabled, such that a wake-up condition on the USB bus can reactivate the platform clocks.

Because the wake-up interrupt is generated and cleared on a 32 KHz clock, this interrupt request responds very slowly to clear actions. For this reason, you must disable the wake-up interrupt to clear the request flag. Disabling the interrupt masks the request instantaneously as this is clocked by the platform clock. You must then wait for at least three 32 KHz clock cycles before re-enabling this interrupt to allow sufficient time for the request flag to clear. Because this interrupt is only used during low-power modes of the USB, it is sufficient to enable the wake-up interrupt just prior to entering the USB Suspend mode.

55.8 Data structures

55.8.1 Device data structures

This section defines the interface data structures used to communicate control, status, and data between DCD software and the device controller. The interface consists of multiple dQH and TD instances.

NOTE

You must ensure that data structures do not span a 4 KB page boundary.

The USB OTG uses an array of device endpoint QHs to organize device transfers. As shown in the following figure, there are two endpoint QHs in the array for each device endpoint—one for IN and the other for OUT. ENDPOINTLISTADDR provides a pointer to the first entry in the array.

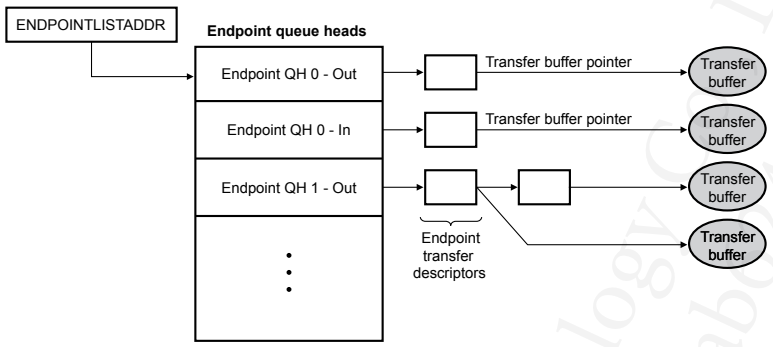


Figure 455. End point QH head organization

55.8.1.1 Endpoint dQH

All transfers are managed in the endpoint dQH. The dQH is a 48-byte data structure, but must align on 64-byte boundaries. During priming of an endpoint, the dTD copies into the overlay area of the dQH, which starts at the NextTD pointer and continues through the end of the buffer pointers entries. After a transfer is complete, the dTD status updates to the CurrentTD pointer. When a packet is in progress, the overlay area of the dQH acts as a staging area for the dTD, so the device controller can access needed information with minimal latency.

The following table shows the endpoint QH structure.

Table 390. Endpoint QH layout

31–30	29	28–27	26–16	15	14–10	9–0	offset
Mult	ZLT	0	Maximum Packet Length	IOS	0	0	00h
Current dTD Pointer						0	04h
Next dTD Pointer						T	08h[1]
0	0	Total Bytes		IOC	0	Status	0Ch[1]
Buffer Pointer (Page 0)						Current Offset	10h[1]
Buffer Pointer (Page 1)						Reserved	14h[1]
Buffer Pointer (Page 2)						Reserved	18h[1]
Buffer Pointer (Page 3)						Reserved	1Ch[1]

Table continues on the next page...

Table 390. Endpoint QH layout (continued)

	Buffer Pointer (Page 4)	Reserved	20h[1]
	Reserved		24h
	Setup buffer bytes 3-0		28h
	Setup buffer bytes 7-4		2Ch

Device controller read/write; all others read-only

1. Offsets 08h through 20h contain the transfer overlay.

55.8.1.1.1 Endpoint capabilities and characteristics (offset = 0h)

This entry specifies static information about the endpoint. In other words, this information does not change over the lifetime of the endpoint. DCD software must not attempt to modify this information when the corresponding endpoint is enabled.

Table 391. Endpoint capabilities and characteristics

Field	Description
31-30 Mult	Multiple packet select This field indicates the number of packets executed per transaction description: 00 Execute N transactions as demonstrated by the USB variable length packet protocol, where N is computed using the Maximum Packet Length field (dQH) and the Total Bytes field (dTD) 01 Execute 1 transaction 10 Execute 2 transactions 11 Execute 3 transactions
	NOTE Non-ISO endpoints must set Mult equal to 00. ISO endpoints must set Mult equal to 01, 10, or 11 as needed.
29 ZLT	Zero length termination select. This field is ignored in isochronous transfers. Clearing this bit enables the hardware to automatically append a zero-length packet when the following conditions are true: • The packet transmitted equals the value of the Maximum Packet Length field • The dTD has exhausted the Total Bytes field After this, the dTD retires. When the device is receiving packets, if the last packet length received equals the maximum packet length, and the value of the Total Bytes field is zero, it waits for a zero-length packet from the host to retire the current dTD. Setting this field disables the zero-length packet. When the device is transmitting, the hardware does not append any zero-length packet. When receiving, it does not require a zero-length packet to retire

Table continues on the next page...

Table 391. Endpoint capabilities and characteristics (continued)

Field	Description
	a dTD, the last packet of which was equal to the maximum packet length. The dTD is retired as soon as the value of the Total Bytes field becomes zero, or a short packet is received. 0 Enable zero-length packet (default) 1 Disable the zero-length packet
	NOTE Each transfer is defined by one dTD, so the zero-length termination is for each dTD. In some software application cases, the logic transfer does not fit into only one dTD, so it does not make sense to add a zero-length termination packet each time a dTD is consumed. For those cases, it is recommended that the ZLT feature be disabled and the software used to generate the zero-length termination.
28-27	Reserved
26-16 Maximum Packet Length	This field directly corresponds to the maximum packet size of the associated endpoint (wMaxPacketSize). The maximum value for this field is 400h (1024).
15 IOS	Interrupt on setup. This field, used on control type endpoints, indicates if USBSTS[UI] is set in response to a setup being received.
14-0	Reserved

55.8.1.1.2 Transfer overlay (offset = 8h-20h)

The seven entries in the overlay area represent a transaction working space for the device controller. The general operational model is that the device controller can detect whether the overlay area contains a description of an active transfer. If it does not contain an active transfer, then the device controller does not read the associated endpoint.

After an endpoint is configured, the dTD is copied into this QH overlay area by the device controller. Until a transfer expires, you must not write the QH overlay area or the associated TD. When the transfer is complete, the device controller writes the results back to the original TD and advances the queue.

See Endpoint dTD for a description of the overlay fields.

55.8.1.1.3 Current dTD Pointer (offset = 0h)

The current dTD pointer is used by the controller to locate the transfer in progress. This entry is for device controller (hardware) use only and must not be modified by DCD software.

Table 392. Current dTD pointer

Field	Description
31-5 Current dTD Pointer	This field is a pointer to the dTD that is represented in the transfer overlay area, and is updated by the device controller to the next dTD pointer during endpoint priming or queue advance.
4-0	Reserved

55.8.1.1.4 Setup buffer (offset = 28h-2Ch)

The setup buffer is a dedicated storage for the 8-byte data that follows a setup PID. See Control endpoint operation for information on the procedure for reading the setup buffer.

NOTE

Each endpoint has a TX and an RX dQH associated with it, and only the RX QH receives setup data packets.

Table 393. Multiple mode control

Offset	Field	Description
28h	31-0 Setup buffer 0	Setup buffer 0. This buffer contains bytes 3 to 0 of an incoming setup buffer packet and is written by the device controller software reads.
2Ch	31-0 Setup buffer 1	Setup buffer 1. This buffer contains bytes 7 to 4 of an incoming setup buffer packet and is written by the device controller software reads.

55.8.1.2 Endpoint dTD

The dTD describes to the device controller the location and quantity of data for a given transfer.

NOTE

The DCD software must not attempt to modify any field in an active dTD except the next dTD pointer.

Table 394. Endpoint dTD

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	offset
Next dTD Pointer																						0	0	0	0	T						00h
0	Total Bytes														IOC	0	0	0	Mult O			0	0	Status									04h
Buffer Pointer (Page 0)																						Current Offset										08h
Buffer Pointer (Page 1)																						0	Frame Number									0Ch
Buffer Pointer (Page 2)																						0	0	0	0	0	0	0	0	0	0	10h
Buffer Pointer (Page 3)																						0	0	0	0	0	0	0	0	0	0	14h

Table continues on the next page...

Table 394. Endpoint dTD (continued)

Buffer Pointer (Page 4)	0	0	0	0	0	0	0	0	0	0	0	0	0	18h

Device controller read/write; all others read-only.

55.8.1.2.1 Next dTD pointer (offset = 0h)

The Next dTD pointer is used to point the device controller to the next dTD in the linked list.

Table 395. Next dTD pointer

Field	Description
31-5 Next dTD Pointer	This field contains the physical memory address of the next dTD to be processed. The field corresponds to memory address signals [31:5], respectively.
4-1	Reserved
0 T	Terminate. This field indicates to the device controller that no more valid entries exist in the queue. 0 - Pointer is valid (points to a valid transfer element descriptor) 1 - Pointer is invalid

55.8.1.2.2 dTD token (offset = 4h)

The dTD token is used to specify attributes for the transfer including the number of bytes to read or write and the status of the transaction.

Table 396. dTD token

Field	Description
31	Reserved
30-16 Total Bytes	This field specifies the total number of bytes moved with this TD. This field decrements by the number of bytes actually moved during the transaction and only on the successful completion of the transaction. The maximum value that you may store in the field is 5*4 K(5000h). This is the maximum number of bytes that five-page pointers can access. Although possible to create a transfer up to 20 KB, this assumes the first offset into the first page is 0. When the offset cannot be predetermined, crossing past the fifth page can be guaranteed by limiting the total bytes to 16 K**. Therefore, the maximum recommended transfer is 16 KB (4000h). **NOTE** Larger transfer sizes can be supported, but require disabling ZLT and using multiple dTDs. If the value of the field is 0 when the host controller fetches this TD (and the active bit is 1), the device controller executes a zero-length transaction and retires the TD.

Table continues on the next page...

Table 396. dTD token (continued)

Field	Description
	For IN transfers, Total Bytes to Transfer need not be an even multiple of the maximum packet length. If you build such a TD for an IN transfer, the last transaction is always less than the maximum packet length. For OUT transfers, total bytes must be evenly divisible by the maximum packet length.
15 IOC	Indicates whether USBSTS[UI] is set in response to the device controller being finished with this dTD.
14-12	Reserved
11-10 MultO	Multiplier Override. This field can possibly transmit ISOs (ISO-IN) to override the multiplier in the QH. This field must be 0 for all packet types that are not transmit-ISO. For example, if QH.MULT equals 3, maximum packet size equals 8, total bytes equals 15, and MultO equals 0 [default], then three packets are sent: {Data2(8); Data1(7); Data0(0)}. If QH.MULT equals 3, maximum packet size equals 8, total bytes equals 15, and MultO equals 2, then two packets are sent: {Data1(8); Data0(7)}. For maximum efficiency, you must compute MultO equals greatest integer of (total bytes / maximum packet size) except for the case when total bytes equals 0; then MultO must be 1. **NOTE** Non-ISO and Non-TX endpoints must set MultO to 0.
9-8	Reserved
7-0 Status	The device controller communicates individual command execution states back to the DCD software. This field contains the status of the last transaction performed on this dTD. The bit encodings are as follows.

Bit	Status field description
7	Active. Set by you to enable the execution of transactions by the device controller.
6	Halted. Set by the device controller during status updates to indicate that a serious error has occurred at the device/endpoint addressed by this dTD. Any time a transaction results in the halted bit being set, the active bit is also cleared.
5	Data Buffer Error. Set by the device controller during status update to indicate that the device controller is unable to maintain the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (underrun).
4	Reserved

Table continues on the next page...

Table 396. dTD token (continued)

Field	Description	
	3	Transaction Error. Set by the device controller during status update in case the device did not receive a valid response from the host (time-out, CRC, bad PID).
	2-0	Reserved

55.8.1.2.3 dTD buffer page pointer list (offset = 8h-18h)

The buffer page pointer list (belonging to a device element TD) is an array of physical memory address pointers. These pointers reference the individual pages of a data buffer.

Table 397. Buffer page pointer list

Field	Description
31-12 Buffer Pointer	Selects the page offset in memory for the packet buffer. Non-virtual memory systems typically set the buffer pointers to a series of incrementing integers.
0;11-0 Current Offset	Offset into the 4 KB buffer where the packet begins.
1;10-0 Frame Number	Written by the device controller to indicate the frame number in which a packet finishes. Typically correlates relative completion times of packets on an ISO endpoint.

55.8.2 Host data structures

This section defines the interface data structures used to communicate control, status, and data between HCD (software) and the enhanced host controller (hardware).

The data structure definitions in this chapter support a 32-bit memory buffer address space. The interface consists of the following:

- Periodic schedule
- Periodic frame list
- Asynchronous schedule
- iTDs
- siTDs
- Queue heads
- Queue element transfer descriptors

The following table shows the functions of data structures.

Table 398. Data structure functions

Data structure	Function
Periodic frame list	Root of all periodic (isochronous and interrupt transfer type) transfers for the host controller

Table continues on the next page...

Table 398. Data structure functions (continued)

Data structure	Function
Asynchronous list	Root for all the bulk and control transfers
iTDs	Manage isochronous data streams
siTDs	Manage isochronous split-transaction data streams
Queue heads, queue element transfer descriptors	Manage all interrupt, control, and bulk data streams

These data structures are optimized to reduce the total memory footprint of the schedule and to reduce the average number of memory accesses needed to execute a USB transaction.

Note that you must ensure that no interface data structure reachable by the EHCI host controller spans a 4 KB page boundary.

The data structures defined in this section are (from the host controller's perspective) a mix of read-only and read/writeable fields. The host controller must preserve the read-only fields on all data structure writes.

55.8.2.1 Periodic frame list

This schedule is for all periodic transfers (isochronous and interrupt). The periodic schedule is referenced from the operational registers space using the PERIODICLISTBASE address register and USB Frame Index (FRINDEX) register. The periodic schedule is based on an array of pointers called the periodic frame list. The PERIODICLISTBASE address register is combined with the USB Frame Index (FRINDEX) register to produce a memory pointer into the frame list. The periodic frame list implements a sliding window of work over time.

The following figure shows the organization of periodic schedule.

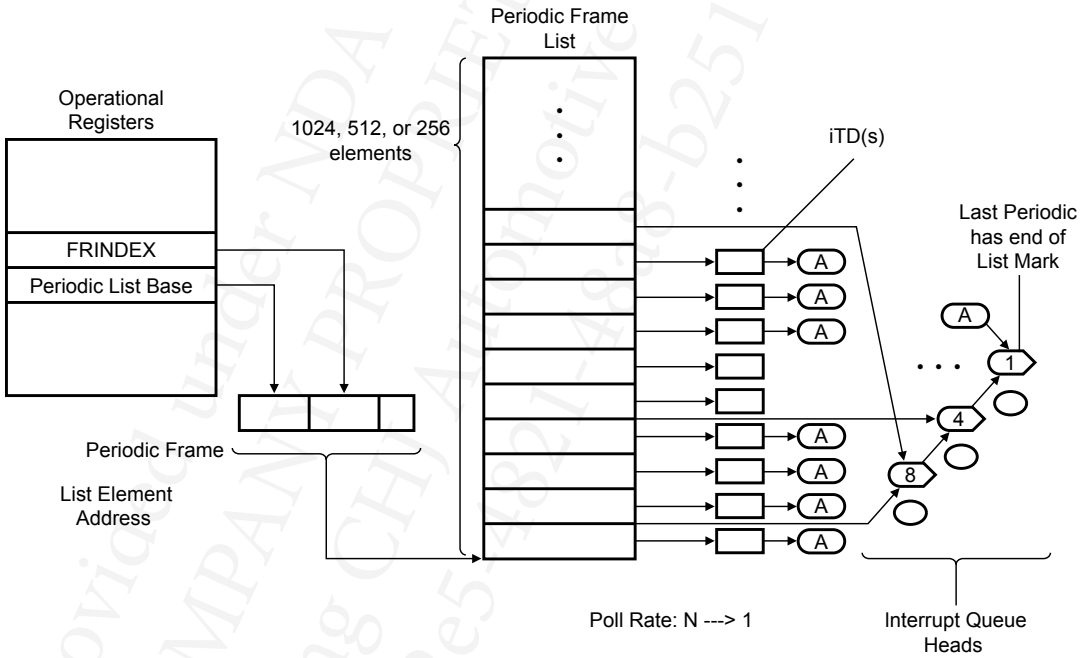


Figure 456. Periodic schedule organization

Split-transaction interrupt, bulk, and control are also managed using QHs and queue element transfer descriptors.

The Periodic frame list is a 4 KB page-aligned array of frame list link pointers. The length of the frame list may be programmable. The programmability of the periodic frame list is exported to the system software via the Host Controller Capability Parameters (HCCPARAMS) register.

Table 399. Frame list length

Programmability	Number of elements
No	1024
Yes	Selectable by system software to be 256, 512, or 1024 elements

An implementation must support all three of the above sizes. Programming the size (that is, the number of elements) is accomplished by system software writing the appropriate value into USBCMD[FS1].

Frame List Link pointers direct the host controller to the first work item in the frame's periodic schedule for the current microframe. The link pointers are aligned on DWord boundaries within the Frame List.

The table below illustrates the format of the frame list element pointer.

Table 400. Format of frame list element pointer

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Frame List Link pointer																											0		Typ			03-00 h

Frame List Link pointers always reference memory objects that are 32-byte aligned. The referenced objects may be isochronous transfer descriptors for HS devices, siTDs (for FS isochronous endpoints), or a QH (used to support HS, FS, and LS interrupts). System software must not place non-periodic schedule items into the periodic schedule. The least significant bits in a frame list pointer are used to indicate to the host controller the type of object the pointer is referencing.

The least significant bit is the T-Bit (bit 0). When the value of this field is 1, the host controller never uses the value of the frame list pointer as a physical memory pointer. The Typ field is used to indicate the exact type of data structure being referenced by this pointer; the value encodings are shown below.

Table 401. Typ field value definitions

Value	Meaning
00b	Isochronous transfer descriptor (iTD)
01b	QH
10b	Split transaction isochronous transfer descriptor (siTD)
11b	Frame span traversal node (FSTN)

55.8.2.2 Asynchronous list QH pointer

The asynchronous transfer list (based at the ASYNCLISTADDR register) is where all of the control and bulk transfers are managed. Host controllers use the list based at ASYNCLISTADDR only when the periodic list is empty or disabled.

Preliminary Information
COMPANY CONFIDENTIAL

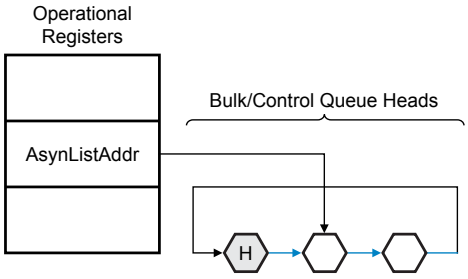


Figure 457. Asynchronous schedule organization

The asynchronous list is a simple circular list of QHs. ASYNCLISTADDR is a pointer to the next QH. This implements a round-robin service for all QHs linked into the asynchronous list.

55.8.2.3 iTD

The format of an iTD is shown in the table below.

This structure is used only for HS isochronous endpoints. All other transfer types must use queue structures. Isochronous TDs must be aligned on a 32-byte boundary.

Table 402. iTD

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Next Link Pointer																											0		Typ		T	03-00h
Status				Transaction 0 length												IOC	PG*					Transaction 0 offset*										07-04h
Status				Transaction 1 length												IOC	PG*					Transaction 1 offset*										0B-08h
Status				Transaction 2 length												IOC	PG*					Transaction 2 offset*										0F-0Ch
Status				Transaction 3 length												IOC	PG*					Transaction 3 offset*										13-10h
Status				Transaction 4 length												IOC	PG*					Transaction 4 offset*										17-14h

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cs462e5-4e2-48a-b251-900a8ab621c1

Table 402. iTD (continued)

Status	Transaction 5 length		I O C	PG*	Transaction 5 offset*				1B-18h
Status	Transaction 6 length		I O C	PG*	Transaction 6 offset*				1F-1Ch
Status	Transaction 7 length		I O C	PG*	Transaction 7 offset*				23-20h
Buffer Pointer (Page 0)					EndPt	R	Device address		27-24h
Buffer Pointer (Page 1)					I/O	Maximum Packet Size			2B-28h
Buffer Pointer (Page 2)					-			Mult	2F-2Ch
Buffer Pointer (Page 3)					-				33-30h
Buffer Pointer (Page 4)					-				37-34h
Buffer Pointer (Page 5)					-				3B-38h
Buffer Pointer (Page 6)					-				3F-3Ch

Host controller read/write Host controller read-only

These fields may be modified by the host controller if the I/O field indicates an OUT.

55.8.2.3.1 Next link pointer

The first DWord of an iTD is a pointer to the next schedule data structure. The following table describes the Next Schedule Element pointer field.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390da4b624c1

Table 403. Next schedule element pointer

Bit	Description
31-5 Link Pointer (LP)	These bits correspond to memory address signals [31:5], respectively. This field points to another iTD, siTD, or QH.
4-3 Reserved	This field is reserved and its value has no effect on operation. You must initialize this field to zero.
2-1 QH/(s)iTD Select (Typ)	This field indicates to the host controller whether the item referenced is an iTD, siTD, or a QH. This allows the host controller to perform appropriate processing on the item after it is fetched. Value encodings are given below. 00b - iTD 01b - QH 10b - siTD 11b - Frame span traversal node (FSTN)
0 Terminate (T)	1b - Link Pointer field is invalid. 0b - Link Pointer field is valid.

55.8.2.3.2 iTD transaction status and control list

DWords 1 through 8 are eight slots of transaction control and status.

Each transaction description includes:

- Status results field
- Transaction length (bytes to send for OUT transactions and bytes received for IN transactions)
- Buffer offset. The PG and Transaction X Offset fields are used with the buffer pointer list to construct the starting buffer address for the transaction.

The host controller uses the information in each transaction description plus the endpoint information contained in the first three DWords of the Buffer Page Pointer list to execute a transaction on the USB.

The following table describes the iTD Transaction Status and Control fields.

Table 404. iTD transaction status and control

Bit	Description	
31-28 Status	Records the status of the transaction executed by the host controller for this slot. This field is a bit vector with the following encoding:	
	Bit	Definition
	31	Active. Set to 1 by you to enable the execution of an isochronous transaction by the host controller. When the transaction associated with this descriptor is completed, the host controller sets this bit to 0 to indicate that a transaction for this element should not be executed when it is next encountered in the schedule.

Table continues on the next page...

Table 404. iTD transaction status and control (continued)

Bit	Description	
	30	Data Buffer Error. Set to 1 by the host controller during status update to indicate that the host controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). If an overrun condition occurs, no action is necessary.
	29	Babble Detected. Set to 1 by the host controller during status update when "babble" is detected during the transaction generated by this descriptor.
	28	Transaction Error (XactErr). Set to 1 by the host controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, and so on). This bit may only be set for isochronous IN transactions.
27-16 Transaction X Length	For an OUT transaction, this field is the number of data bytes that the host controller sends during the transaction. The host controller is not required to update this field to reflect the actual number of bytes transferred during the transfer. For an IN transaction, this is the initial value of the endpoint to deliver. During status update, the host controller writes back the field is the number of bytes the host expects the number of bytes successfully received. The value in this register is the actual byte count (0-zero length data, 1-one byte, 2-two bytes, and so on). The maximum value this field may contain is C00h (3072).	
15 IOC	If the value of this field is 1, it specifies that when this transaction completes, the host controller should issue an interrupt at the next interrupt threshold.	
14-12 Page Select (PG)	You configure the value of this field to indicate which of the buffer page pointers in the offset field in this slot must be concatenated to produce the starting memory address for this transaction. The valid range of values for this field is 0 to 6.	
11-0 Transaction X Offset	This field is a value that is an offset, expressed in bytes, from the beginning of a buffer. The field is concatenated onto the buffer page pointer indicated in the PG field to produce the starting buffer address for this transaction.	

55.8.2.3.3 iTD buffer page pointer list (plus)

DWords 9-15 of an isochronous transaction descriptor are nominally page pointers (4 KB aligned) to the data buffer for this TD. This data structure requires the associated data buffer to be contiguous (relative to virtual memory), but allows the physical memory pages to be non-contiguous.

Seven page pointers are provided to support the expression of eight isochronous transfers. The seven pointers allow for 3 (transactions) * 1024 (maximum packet size) * 8 (transaction records) = 24576 bytes to be moved with this data structure, regardless of the alignment offset of the first page.

Because each pointer is a 4 KB aligned page pointer, the least significant 12 bits in several of the page pointers are used for other purposes.

The tables below illustrate the field descriptions.

Table 405. iTD buffer pointer page 0 (plus)

Field	Description
31-12	This is a 4 KB aligned pointer to physical memory that corresponds to memory address bits [31:12].

Table continues on the next page...

Table 405. iTD buffer pointer page 0 (plus) (continued)

Field	Description
Buffer Pointer (page 0)	
11-8 Endpoint Number (Endpt)	This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.
7 Reserved	This field is reserved for future use and you must initialize it to 0.
6-0 Device Address	This field selects the specific device serving as the data source or sink.

Table 406. iTD buffer pointer page 1 (plus)

Field	Description
31-12 Buffer Pointer (page 1)	This is a 4 KB aligned pointer to physical memory that corresponds to memory address bits [31:12].
11 Direction (I/O)	This field encodes whether the HS transaction should use an IN or OUT PID. 0 - OUT 1 - IN
10-0 Maximum Packet Size	This directly corresponds to the maximum packet size of the associated endpoint (wMaxPacketSize). This field is used for high-bandwidth endpoints where more than one transaction is issued per transaction description (per microframe). The field is used with the Multi field to support high-bandwidth pipes. This field is also used for all IN transfers to detect packet babble. You must not set a value larger than 1024 (400h). Any larger value yields undefined results.

Table 407. iTD buffer pointer page 2 (plus)

Field	Description
31-12 Buffer Pointer	This is a 4 KB aligned pointer to physical memory that corresponds to memory address bits [31:12].
11-2 Reserved	This field is reserved for future use and must be set to 0.
1-0 Multi	This field is used to indicate to the host controller the number of transactions that are executed per transaction description (per microframe). The valid values are: 00b - Reserved—must always be zero 01b - One transaction to be issued for this endpoint per microframe 10b - Two transactions to be issued for this endpoint per microframe 11b - Three transactions to be issued for this endpoint per microframe

Table 408. iTD buffer pointer pages 3-6

Field	Description
31-12 Buffer Pointer	This is a 4 KB aligned pointer to physical memory that corresponds to memory address bits [31:12].
11-0 Reserved	This field is reserved for future use and must be set to 0.

55.8.2.4 siTD

All FS isochronous transfers through the internal transaction translator are managed using the siTD data structure, which satisfies the operational requirements for managing the split transaction protocol.

The following table shows the siTD.

Table 409. siTD

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Next Link Pointer																											0		Typ		T	03-00h
I/O	Port Number				-	Hub Addr								-				EndPt		-	Device Address											07-04h
Reserved																μFrame C-mask				μFrame S-mask												0B-08h
IOC	P	-						Total Bytes to Transfer								μFrame C-prog-mask				Status												0F-0Ch
Buffer Pointer (Page 0)																						Current Offset										13-10h
Buffer Pointer (Page 1)																						-					TP		T-Count			17-14h
Back Pointer																											0				T	1B-18h

	Host controller read/write		Host controller read only

55.8.2.4.1 Next link pointer

DWord0 of a siTD is a pointer to the next schedule data structure.

The following table describes the next link pointer fields.

Table 410. Next link pointer

Field	Description
31-5	Next Link Pointer (LP). This field contains the address of the next data object to be processed in the periodic list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved. This field must be written to zeros.
2-1	QH/(s)iTD Select (Typ). This field indicates to the host controller whether the item referenced is an iTD/siTD or a QH. This allows the host controller to perform the proper type of processing on the item after it is fetched. Value encodings are: 00b - iTD 01b - QH 10b - siTD 11b - FSTN
0	Terminate (T) 1b - Link pointer field not valid 0b - Link pointer valid

55.8.2.4.2 siTD endpoint capabilities/characteristics

DWords 1 and 2 specify static information about the FS endpoint, the addressing of the parent companion controller, and microframe scheduling control.

The tables below describe the endpoint and transaction translator characteristics, and microframe schedule control fields.

Table 411. Endpoint and transaction translator characteristics

Field	Description
31	Direction (I/O). This field encodes whether the FS transaction should be an IN or OUT. 0 - OUT 1 - IN
30-24	Port number. This field is the port number of the recipient transaction translator.
23	This field is reserved and must be set to 0.
22-16	Hub address. This field holds the device address of the companion controller's hub.
15-12	This field is reserved and must be set to 0.
11-8	Endpoint number (Endpt). This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.
7	This field is reserved for future use and must be set to 0.
6-0	Device address. This field selects the specific device serving as the data source or sink.

Table 412. Microframe schedule control

Field	Description
31-16	This field is reserved for future use and must be set to 0.
15-8	Split Completion Mask (mFrame C-Mask). This field (along with the Active and SplitX state fields in the Status byte) is used to determine during which microframes the host controller must execute complete-split transactions. When the criteria for using this field is met, an all-zeros value has undefined behavior. The host controller uses the value of the three low-order bits of the FRINDEX register to index into this bit field. If the FRINDEX register value indexes to a position for which the mFrame C-Mask field is a 1, then this siTD is a candidate for transaction execution. There may be more than one bit set in this mask.
7-0	Split Start Mask (mFrame S-mask). This field (along with the Active and SplitX state fields in the Status byte) is used to determine during which microframes the host controller must execute start-split transactions. The host controller uses the value of the three low-order bits of the FRINDEX register to index into this bit field. If the FRINDEX register value indexes to a position for which the mFrame S-mask field is a 1, then this siTD is a candidate for transaction execution. An all zeros value in this field, in combination with existing periodic frame list, has undefined results.

55.8.2.4.3 siTD transfer state

DWords 3-6 are used to manage the state of the transfer.

The following table describes siTD transfer state fields.

Table 413. siTD transfer status and control

Bit	Description
31	IOC When the host controller determines that the split transaction is complete, it asserts a hardware interrupt at the next interrupt threshold. 0 - Do not interrupt when transaction is complete. 1 - Do interrupt when transaction is complete.
30	Page Select (P). Used to indicate which data page pointer must be concatenated with the CurrentOffset field to construct a data buffer pointer. 0 - Selects Page 0 pointer 1 - Selects Page 1 pointer The host controller is not required to write this field back when the siTD is retired (Active bit transitioned from a one to a zero).
29-26	This field is reserved for future use and must be set to zero.
25-16	Total Bytes to Transfer. You initialize this field to the total number of bytes expected in this transfer. Maximum value is 1023 (3FFh).
15-8	µFrame Complete-Split Progress Mask (C-prog-Mask). This field is used by the host controller to record which split-completes has been executed.

Table continues on the next page...

Table 413. siTD transfer status and control (continued)

Bit	Description	
7-0	Status. This field records the status of the transaction executed by the host controller for this slot. This field is a bit vector with the following encoding:	
	Bit number	Definition
	7	Active. You set this bit to 1 to enable the execution of an isochronous split transaction by the host controller.
	6	ERR. The host controller writes a 1 to this bit when an ERR response is received from the companion controller.
	5	Data Buffer Error. The host controller writes a 1 to this bit during status update to indicate that the host controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). In the case of an under run, the host controller transmits an incorrect CRC (thus invalidating the data at the endpoint). If an overrun condition occurs, no action is necessary.
	4	Babble Detected. The host controller writes a 1 to this bit during status update when "babble" is detected during the transaction generated by this descriptor.
	3	Transaction Error (XactErr). The host controller writes a 1 to this bit during status update in the case for which the host did not receive a valid response from the device (Timeout, CRC, Bad PID, and so on). This bit is set only for IN transactions.
	2	Missed Microframe. The host controller detected that a host-induced hold-off caused the host controller to miss a required complete-split transaction.
	1	Split Transaction State (SplitXstate). The bit encodings are: 00b - Do Start Split. This value directs the host controller to issue a Start Split transaction to the endpoint when a match is encountered in the S-mask. 01b - Do Complete Split. This value directs the host controller to issue a Complete Split transaction to the endpoint when a match is encountered in the C-mask.
	0	Reserved—must always be zero.

55.8.2.4.4 siTD buffer pointer list (plus)

DWords 4 and 5 are the data buffer page pointers for the transfer. This structure supports one physical page cross. The most significant 20 bits of each DWord in this section are the 4 KB (page) aligned buffer pointers.

The least significant 12 bits of each DWord are used as additional transfer state. The following table describes the siTD buffer pointer fields.

Provided by IHS Markit
COMPANY CONFIDENTIAL
Beijing CHX Xinyuan Tech Co., Ltd.
c8c462e6-4821-4a3-b23-3900d6b6041

Table 414. Buffer page pointer list (plus)

Field	Description	
31-12	Buffer Pointer List. Bits [31:12] of DWords 4 and 5 are 4 KB page-aligned, physical memory addresses. These bits correspond to physical address bits [31:12] respectively. The lower 12 bits in each pointer are defined and used as specified below. This field P specifies the current active pointer.	
11-0	Page 0: Current Offset. The 12 least significant bits of the Page 0 pointer is the current byte offset for the current page pointer (as selected with the page indicator bit (P field)). The host controller is not required to write this field back when the siTD is retired (Active bit transitioned from a 1 to a 0). The least significant bits of Page 1 pointer are split into three sub-fields. Page 1:	
	Field	Description
	11-5	Reserved
	4-3	Transaction Position (TP). This field is used with T-Count to determine whether to send All, Begin, Mid, or End with each outbound transaction payload. System software must initialize this field with the appropriate starting value. The host controller must correctly manage this state during the lifetime of the transfer. The bit encodings are: 00b - All. The entire FS transaction data payload is in this transaction (that is, less than or equal to 188 bytes). 01b - Begin. This is the first data payload for a FS transaction that is greater than 188 bytes. 10B - Mid. This is the middle payload for a FS OUT transaction that is larger than 188 bytes. 11b - End. This is the last payload for a FS OUT transaction that was larger than 188 bytes.
	2-0	Transaction Count (T-Count). You initialize this field with the number of OUT start-splits this transfer requires. Any value larger than 6 is undefined.

55.8.2.4.5 siTD back link pointer

DWord 6 of a siTD is simply another schedule link pointer. This pointer is always zero, or references a siTD, and it cannot reference any other schedule data structure.

The following table describes the siTD back link pointer fields.

.

Table 415. siTD back link pointer

Field	Description
31-5	siTD Back Pointer. This field is a physical memory pointer to an siTD.
4-1	Reserved—must always be zero."
0	Terminate (T) 1 - siTD Back Pointer field is invalid 0 - siTD Back Pointer field is valid

55.8.2.5 Queue element transfer descriptor (qTD)

This data structure is only used with a QH. It describes one or more USB transactions to transfer up to 20480 (5*4096) bytes. The structure contains two structure pointers used for queue advancement, a DWord of transfer state, and a five-element array of data buffer pointers. It is 32 bytes and must be physically contiguous.

The buffer associated with this transfer must be virtually contiguous. The buffer may start on any byte boundary; however, for optimal usage of on-chip buses, it is recommended to align the buffers on a 32-byte boundary. A separate buffer pointer list element must be used for each physical page in the buffer, regardless of whether the buffer is physically contiguous.

Host controller updates (host controller writes) to stand-alone qTDs only occur during transfer retirement. References in the following bit field definitions of updates to the qTD are to the qTD portion of a QH.

The following table shows the QH data structure.

Table 416. QH data structure

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Next qTD Pointer																											0				T	03-00h
Alternate Next qTD Pointer																											0				T	07-04h
DT	Total Bytes to Transfer															IOC	C_Page			Cerr		PID code		Status								0B-08h
Buffer Pointer (Page 0)																				Current Offset												0F-0Ch
Buffer Pointer (Page 0)																				-												13-10h
Buffer Pointer (Page 0)																				-												17-14h

Table continues on the next page...

Table 416. QH data structure (continued)

		4h
Buffer Pointer (Page 0)	-	1B-18h
Buffer Pointer (Page 0)	-	1F-1Ch

	Host controller read/write		Host controller read only.

qTDs must be aligned on 32-byte boundaries.

55.8.2.5.1 Next qTD pointer

The first DWord of an element TD is a pointer to another transfer element descriptor. The following table describes Next qTD pointer fields.

Table 417. qTD next element transfer pointer (DWord 0)

Field	Description
31-5	Next Transfer Element Pointer. This field contains the physical memory address of the next qTD to be processed. The field corresponds to memory address signals [31:5], respectively.
4-1	Reserved
0	Terminate (T) 1 - Pointer is invalid 0 - Pointer is valid (points to a valid transfer element descriptor) This bit indicates to the host controller that there are no more valid entries in the queue.

55.8.2.5.2 Alternate next qTD pointer

The second DWord of a qTD is used to support hardware-only advance of the data stream to the next TD on short packet. To be more explicit, the host controller always uses this pointer when the current qTD is retired because of a short packet. The following table describes the TD Alternate Next Element Transfer Pointer field descriptions.

Freescale Semiconductor Inc. NDA only
COMPANY PROPRIETARY
Beijing CAT Automotive Technology Co., Ltd.
c8c40e5-4921-483-b251-39003ab6246d

Table 418. TD alternate next element transfer pointer (DWord 1)

Field	Description
31-5	Alternate Next Transfer Element Pointer. This field contains the physical memory address of the next qTD to be processed in the event that the current qTD execution encounters a short packet (for an IN transaction). The field corresponds to memory address signals [31:5], respectively.
4-1	Reserved
0	Terminate (T). This bit indicates to the host controller that there are no more valid entries in the queue. 1 - Pointer is invalid 0 - Pointer is valid (points to a valid transfer element descriptor)

55.8.2.5.3 qTD token

The third DWord of a qTD contains most of the information that the host controller requires to execute a USB transaction (the remaining endpoint-addressing information is specified in the QH).

NOTE

The field descriptions forward reference fields defined in the QH. Where necessary, these forward references are preceded with a QH notation.

The following table describes the TD token fields.

Table 419. TD token (DWord 2)

Field	Description
31 Data Toggle	This is the data toggle sequence field. The use of this field depends on the setting of the Data Toggle Control field in the QH.
30-16 Total Bytes to Transfer	This field specifies the total number of bytes to be moved with this TD. The field is decremented by the number of bytes actually moved during the transaction only on the successful completion of the transaction. The maximum value that you may store in this field is 5 * 4 KB (5000h). This is the maximum number of bytes five page pointers can access. If the value of this field is 0 when the host controller fetches this TD (and the value of the Active field is 1), the host controller executes a zero-length transaction and retires the TD. It is not a requirement for OUT transfers that this field be an even multiple of QH.Maximum Packet Length. If you build such a TD for an OUT transfer, the last transaction is always less than QH.Maximum Packet Length. Although it is possible to create a transfer up to 20 KB, it is required that the first offset into the first page is 0. When the offset cannot be predetermined, crossing past the 5th page can be guaranteed by limiting the total bytes to 16 KB**. Therefore, the maximum recommended transfer is 16 KB (4000h).
15 IOC	If the value of this bit is 1, it specifies that when this qTD is completed, the host controller must issue an interrupt at the next interrupt threshold.
14-12 Current Page (C_Page)	This field is used as an index into the qTD buffer pointer list. Valid values are in the range 0h to 4h. The host controller is not required to write this field back when the qTD is retired.

Table continues on the next page...

Table 419. TD token (DWord 2) (continued)

Field	Description
11-10 Error Counter (CERR)	This field is a 2-bit down counter that keeps track of the number of consecutive errors detected when executing this qTD. If this field is programmed with a non-zero value during set-up, the host controller decrements the count and writes it back to the qTD if the transaction fails. If the counter counts from 1 to 0, the host controller: • Marks the qTD inactive • Writes the Halted bit to 1 • Writes the error status bit for the error that caused this field to decrement to 0 An interrupt is generated if the the value of the USB Error Interrupt Enable field in the USBINTR register is 1. If HCD writes a 0 to this field during set-up, the host controller does not count errors for this qTD and there is no limit on the retries of this qTD. Note that write-backs of intermediate execution state are to the QH overlay area, not to the qTD. Transaction Error - Decrement Data Buffer Error - No decrement[3] Stalled - No decrement[1] Babble Detected - No decrement[1] No Error - No decrement[2]

Error	Decrement counter
1	Detection of Babble or Stall automatically halts the QH. Thus, count is not decremented.
2	If the EPS field indicates an HS device or the QH is in the asynchronous schedule (and PIDCode indicates an IN or OUT) and a bus transaction completes and the host controller does not detect a transaction error, then the host controller must reset CERR to extend the total number of errors for this transaction. For example, CERR must be reset with maximum value (3) on each successful completion of a transaction. The host controller must never reset this field if the value at the start of the transaction is 00b. See Split transaction interrupt for CERR adjustment rules when the EPS field indicates a FS or LS device and the QH is in the periodic schedule. See Asynchronous—do complete split for CERR adjustment rules when the EPS field indicates a FS or LS device, the QH is in the asynchronous schedule and the PIDCode indicates a SETUP.
3	Data buffer errors are host problems; these do not count against the device's retries.

NOTE

You must not write 0 to CERR when the EPS field is programmed with a value indicating an FS or LS device. This combination could result in an undefined behavior.

Table continues on the next page...

Table 419. TD token (DWord 2) (continued)

Field	Description	
9-8 PID Code	This field is an encoding of the token, which must be used for transactions associated with this TD. The encodings are described as follows.	
	00b	OUT token generates token (E1H)
	01b	IN token generates token (69H)
	10b	SETUP token generates token (2DH) (undefined if endpoint is an interrupt, the QH is non-zero) transfer type
	11b	Reserved
7-0 Status	This field is used by the host controller to communicate individual command execution states back to HCD. This field contains the status of the last transaction performed on this qTD. The bit encodings are described as follows:	
	Bit	**Status field description**
	7	Active. You write 1 to this bit to enable the execution of transactions by the host controller.
	6	Halted. The host controller writes a 1 to this field during status updates to indicate that a serious error has occurred at the device/endpoint addressed by this qTD. This can be caused by babble, the error counter counting down to zero, or reception of the STALL handshake from the device during a transaction. Any time a transaction results in the Halted field being set to a 1, the Active field is also set to 0.
	5	Data Buffer Error. The host controller writes a 1 to this field during status updates to indicate that the host controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). If an overrun condition occurs, the host controller forces a timeout condition on the USB, invalidating the transaction at the source. If the host controller writes this field to a 1, then it remains a 1 for the duration of the transfer.
	4	Babble Detected. The host controller writes a 1 to this field during status update when "babble" is detected during the transaction. In addition to writing a 1 to this field, the host controller writes a 1 to the Halted field. Because "babble" is considered a fatal error for the transfer, writing a 1 to the Halted field ensures that no more transactions occur because of this descriptor.
	3	Transaction Error (XactErr). The host controller writes a 1 to this field during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, and so on). If the host controller writes a 1 to this field, then it remains a 1 for the duration of the transfer.
	2	Missed Microframe. This field is ignored unless the QH.EPS field indicates an FS or LS endpoint and the QH is in the periodic list. The host controller writes a 1 to this field when it detects that a host-induced hold-off caused the host controller to miss a required complete-split transaction. If the host controller writes a 1 to this field, then it remains a 1 for the duration of the transfer.

Table continues on the next page...

Table 419. TD token (DWord 2) (continued)

Field	Description	
	1	Split Transaction State (SplitXstate). The host controller ignores this field unless the QH.EPS field indicates a FS or LS endpoint. With an FS or LS device, the host controller uses this field to track the state of the split-transaction. The functional requirements of the host controller for managing this state field and the split transaction protocol depend on whether the endpoint is in the periodic or asynchronous schedule. Following are the bit encodings: 0b - Do Start Split. This value directs the host controller to issue a start split transaction to the endpoint. 1b - Do Complete Split. This value directs the host controller to issue a complete split transaction to the endpoint.
	0	Ping State (P)/ERR. If the QH.EPS field indicates an HS device and the PID_Code indicates an OUT endpoint, then this is the state field for the Ping protocol. The bit encodings are: 0b - Do OUT. This value directs the host controller to issue an OUT PID to the endpoint. 1b - Do Ping. This value directs the host controller to issue a PING PID to the endpoint. If the QH.EPS field does not indicate an HS device, then this field is used as an error indicator field. The host controller writes this field to a 1 whenever a periodic split-transaction receives an ERR handshake.

55.8.2.5.4 qTD buffer page pointer list

The last five DWords of a qTD form an array of physical memory address pointers. These pointers reference the individual pages of a data buffer.

System software initializes the Current Offset field to the starting offset into the current page, where this page is selected through the value in the C_Page field.

The following table describes the qTD buffer pointer(s) (DWords 3-7) fields.

Table 420. qTD buffer pointer(s) (DWords 3-7)

Field	Description
31-12	Buffer Pointer List. Each element in the list is a 4 KB page-aligned physical memory address. The lower 12 bits in each pointer are reserved (except for the first one), as each memory pointer must reference the start of a 4 KB page. The field C_Page specifies the current active pointer. When the transfer element descriptor is fetched, the starting buffer address is selected using C_Page (similar to an array index to select an array element). If a transaction spans a 4 KB buffer boundary, the host controller must detect the page-span boundary in the data stream, increment C_Page, advance to the next buffer pointer in the list, and conclude the transaction through the new buffer pointer.
11-0	Current Offset (Reserved). This field is reserved in all pointers except the first one (for example, page 0). The host controller must ignore all of the reserved bits. For the page 0 current offset interpretation, this field is the byte offset into the current page (as selected by C_Page). The host controller is not required to write this field back when the qTD is retired. You must ensure the reserved fields are initialized to 0.

55.8.2.6 QH

The following table shows the QH structure layout.

Table 421. QH structure layout

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
QH Horizontal Link Pointer (QHLP)																											0		Typ		T	03-00h
RL			C	Maximum Packet Length												H	DTC	EP	EndPt			I	Device Address									07-04h
Mult		Port Number*						Hub Addr*								μFrame C-mask*						μFrame S-mask*										0B-08h
Current qTD Pointer																													0			0F-0Ch
Next qTD Pointer																													0		T	13-10h
Alternate Next qTD Pointer																													NakCnt		T	17-14h
DT	Total Bytes to Transfer															IOC	C_Page		Cerr		PID Code		Status									1B-18h
Buffer Pointer (Page 0)																				Current Offset												1F-1Ch
Buffer Pointer (Page 1)																				Reserved		C-prog-mask*										23-20h
Buffer Pointer (Page 2)																				S-bytes*									FrameTag*			27-24h
Buffer Pointer (Page 3)																				-												2B-28h
Buffer Pointer (Page 4)																				-												2F-2Ch

	Host controller read/write		Host controller read only

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHI Automotive Technology Co., Ltd.
c8462e54821-4848-1251-390d3a0624d1

55.8.2.6.1 QHLP

The first DWord of a QH contains a link pointer to the next data object to be processed after any required processing in this queue has been completed, as well as the control bits defined below.

This pointer may reference a QH or one of the isochronous transfer descriptors. It must not reference a qTD.

The following table describes the QH DWord 0 fields.

Table 422. QH DWord 0

Field	Description
31-5	QHLP. This field contains the address of the next data object to be processed in the horizontal list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	QH/(s)iTD Select (Typ). This field indicates to the hardware whether the item referenced by the link pointer is an iTD, siTD, or a QH. This allows the host controller to perform the proper type of processing on the item after it is fetched. Value encodings are: 00b - iTD 01b - QH 10b - siTD 11b - FSTN
0	Terminate (T). If the QH is in the context of the periodic list, a one bit in this field indicates to the host controller that this is the end of the periodic list. This bit is ignored by the host controller when the QH is in the asynchronous schedule. You must ensure that QHs reachable by the host controller always have valid horizontal link pointers. 1 - Last QH (pointer is invalid). 0 - Pointer is valid.

55.8.2.6.2 QH endpoint capabilities/characteristics

The second and third DWords of a QH specify static information about the endpoint. This information does not change over the lifetime of the endpoint. There are three types of information in this region:

- Endpoint characteristics: These are the USB endpoint characteristics including addressing, maximum packet size, and endpoint speed.
- Endpoint capabilities: These are adjustable parameters of the endpoint. They affect how the endpoint data stream is managed by the host controller.
- Split transaction characteristics: This data structure is used to manage FS and LS data streams for bulk, control, and interrupt via split transactions to USB2.0 hub transaction translator. There are additional fields used for addressing the hub and scheduling the protocol transactions (for periodic).

The host controller must not modify the bits in this region.

The following table describes the Endpoint characteristics: QH DWord 1 fields.

Beijing Lomotive Technology Co., Ltd.
c8c46e2e-4821-4a8-b21-390d3a16241
Product under NDA only
COMPANY PROPRIETARY

Table 423. Endpoint characteristics: QH DWord 1

Field	Description	
31-28	Nak Count Reload (RL). This field contains a value that is used by the host controller to reload the Nak Counter field.	
27	Control Endpoint Flag (C). If the QH.EPS field indicates the endpoint is not a HS device, and the endpoint is a control endpoint, then you must set this bit to 1. Otherwise, you must always set this bit to 0.	
26-16	Maximum Packet Length. This directly corresponds to the maximum packet size of the associated endpoint (wMaxPacketSize). The maximum value this field may contain is 400h (1024).	
15	Head of Reclamation List Flag (H). This bit is set by system software to mark a QH as being the head of the reclamation list.	
14	Data Toggle Control (DTC). This bit specifies where the host controller must obtain the initial data toggle on an overlay transition. 0b - Ignore DT bit from incoming qTD. The host controller preserves the DT bit in the QH. 1b - Initial data toggle from the incoming qTD DT bit. Host controller replaces the DT bit in the QH from the DT bit in the qTD.	
13-12	Endpoint Speed (EPS). This is the speed of the associated endpoint. Bit combinations are:	
	Value	**Meaning**
	00b	FS (12 Mbits/sec)
	01b	LS (1.5 Mbits/sec)
	10b	HS (480 Mbits/sec)
	11b	Reserved
	This field must not be modified by the host controller.	
11-8	Endpoint Number (Endpt). This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.	
7	Inactivate on Next Transaction (I). This bit is used by the system software to request that the host controller set the Active bit to 0. See Rebalancing the periodic schedule for full operational details. This field is only valid when the QH is in the periodic schedule and the EPS field indicates a FS and LS endpoint. Writing 1 to this field when the QH is in the asynchronous schedule or the EPS field indicates a HS device and yields undefined results.	
6-0	Device Address. This field selects the specific device serving as the data source or sink.	

The table below describes the endpoint capabilities: QH DWord 2 field descriptions.

Preliminary Information
COMPANY CONFIDENTIAL
Beijing C.A.R Automotive Technology Co., Ltd.
c8c462e2-2201-4ba8-b251-390da1624c

Table 424. Endpoint capabilities: QH DWord 2

Field	Description
31-30	High-Bandwidth Pipe Multiplier (Mult). This field is a multiplier used to key the host controller as the number of successive packets the host controller may submit to the endpoint in the current execution. The host controller makes the simplifying assumption that you properly initialize this field (regardless of the location of the QH in the schedules or other run-time parameters). The valid values are: 00b - Reserved. A 0 in this field yields undefined results. 01b - One transaction to be issued for this endpoint per microframe 10b - Two transactions to be issued for this endpoint per microframe 11b - Three transactions to be issued for this endpoint per microframe
29-23	Port Number. This field is ignored by the host controller unless the EPS field indicates a FS or LS device. The value is the port number identifier on the USB 2.0 Hub (for hub at device address Hub Addr below), below which the FS or LS device associated with this endpoint is attached. This information is used in the split-transaction protocol.
22-16	Hub Addr. This field is ignored by the host controller unless the EPS field indicates a FS or LS device. The value is the USB device address of the USB 2.0 Hub below which the FS or LS device associated with this endpoint is attached. This field is used in the split-transaction protocol.
15-8	Split Completion Mask (μFrame C-Mask). This field is ignored by the host controller unless the EPS field indicates this device is a LS or FS device and this QH is in the periodic list. This field (along with the Active and SplitX-state fields) is used to determine during which microframes the host controller must execute a complete-split transaction. When the criteria for using this field are met, the value 00h in this field has undefined behavior. This field is used by the host controller to match against the three low-order bits of USB Frame Index (FRINDEX). If the USB Frame Index (FRINDEX) bits decode to a position where the value of μFrame C-Mask field is 1, then this QH is a candidate for transaction execution. There may be more than one bit in this mask set.
7-0	Interrupt Schedule Mask (μFrame S-mask). This field is used for all endpoint speeds. You must set this field to 00h when the QH is on the asynchronous schedule. A non-zero value in this field indicates an interrupt endpoint. The host controller uses the value of the three low-order bits of USB Frame Index (FRINDEX) as an index into a bit position in this bit vector. If the μFrame S-mask field has a 1 at the indexed bit position then this QH is a candidate for transaction execution. If the EPS field indicates the endpoint is a HS endpoint, then the transaction executed is determined by the PID_Code field contained in the execution area. This field is also used to support split transaction types: Interrupt (IN/OUT). This condition is true when this field is non-zero and the EPS field indicates this is either a FS or LS device. A value of 00h in this field, in combination with existing in the periodic frame list, has undefined results.

55.8.2.6.3 Transfer overlay-QH

The nine DWords in this area represent a transaction working space for the host controller. The general operational model is that the host controller can detect whether the overlay area contains a description of an active transfer. If it does not contain an active transfer, then it follows the QHLP the next QH. The host controller never follows the Next Transfer Queue Element or Alternate Queue Element pointers unless it is actively attempting to advance the queue. For the duration of the transfer, the host controller keeps the incremental status of the transfer in the overlay area. When the transfer is complete, the results are written back to the original queue element.

The DWord3 of a QH contains a pointer to the source qTD currently associated with the overlay. The host controller uses this pointer to write back the overlay area into the source qTD after the transfer is complete.

The following table describes the Current qTD Link Pointer field descriptions.

Table 425. Current qTD link pointer

Field	Description
31-5	Current Element Transaction Descriptor Link Pointer. Contains the address of the current transaction being processed in this queue and corresponds to memory address signals [31:5], respectively.
4-0	Reserved (R). The host controller ignores these bits when using the value as an address to write data. The actual value may vary depending on the usage.

The DWords 4-11 of a QH are the transaction overlay area. This has the same base structure as a qTD. The QH uses the reserved fields of the page pointers to implement tracking of the state of split transactions.

This area is characterized as an overlay because when the queue is advanced to the next queue element, the source queue element is merged onto this area. This area serves as execution cache for the transfer.

The table below describes the host-controller rules for bits in overlay.

Table 426. Host-controller rules for bits in overlay (DWords 5, 6, 8 and 9)

DWord	Field	Description
5	4-1	Nak Counter (NakCnt)µRW. The host controller decrements this value whenever a transaction for the endpoint associated with this QH results in a Nak or Nyet response. This counter is reloaded from RL before a transaction is executed during the first pass of the reclamation list (relative to an Asynchronous List Restart condition). It is also loaded from RL during an overlay.
6	31	Data Toggle. Controls whether the host controller preserves this bit when an overlay operation is performed.
6	15	IOC. Is always inherited from the source qTD when the overlay operation is performed.
6	11-10	Error Counter (C_ERR). Is copied from the qTD during the overlay and written back during queue advancement.
6	0	Ping State (P)/ERR. If the EPS field indicates a HS endpoint, then you must preserve this field during the overlay operation.
8	7-0	Split-transaction Complete-split Progress (C-prog-mask). Is initialized to zero during any overlay. This field is used to track the progress of an interrupt split-transaction.
9	4-0	Split-transaction Frame Tag (Frame Tag). Is initialized to zero during any overlay. This field is used to track the progress of an interrupt split-transaction.
9	11-5	S-bytes. You must ensure that the S-bytes field in a qTD is zero before activating the qTD. This field is used to keep track of the number of bytes sent or received during an IN or OUT split transaction.

55.8.2.7 Periodic FSTN

This data structure is to be used only for managing FS and LS transactions that span a host-frame boundary.

See Host controller operational model for FSTNs for full operational details. You must not use an FSTN in the asynchronous schedule. An FSTN in the asynchronous schedule results in undefined behavior. You must not use the FSTN feature with a host controller whose USB_HCIVERSION register indicates a revision implementation below 0096h. FSTNs are not defined for implementations before 0.96 and their use yields undefined results.

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
	Normal Path Link Pointer																											0		Typ		T	03-00h
	Back Path Link Pointer																											0		Typ		T	07-04h

Table 427. FSTN structure layout

	Host controller read/write		Host controller read only

55.8.2.7.1 FSTN normal path pointer

The first DWord of an FSTN contains a link pointer to the next schedule object. This object can be of any valid periodic schedule data type.

The following table describes the FSTN normal path pointer fields.

Table 428. FSTN normal path pointer field descriptions

Field	Description
31-5	Normal Path Link Pointer (NPLP). Contains the address of the next data object to be processed in the periodic list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	QH/(s)iTD/FSTN Select (Typ). Indicates to the host controller whether the item referenced is an iTD/siTD, a QH, or an FSTN. This allows the host controller to perform the proper type of processing on the item after it is fetched. Value encodings are: 00b - iTD 01b - QH 10b - siTD 11b - FSTN
0	Terminate (T) 1 - Link Pointer field is not valid. 0 - Link Pointer is valid.

Processed under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48c2-b251-3903ab624c1

55.8.2.7.2 FSTN back path link pointer

The second DWord of an FTSN node contains a Link Pointer to a QH. If the T-bit in this pointer is zero, then this FSTN is a Save-Place indicator. You must set its Typ field to indicate the target data structure is a QH. If the T-bit in this pointer is set to one, then this FSTN is the Restore Indicator. When the T-bit is one, the host controller ignores the Typ field.

The following table describes the FSTN back path link pointer fields.

Table 429. FSTN back path link pointer field descriptions

Field	Description
31-5	Back Path Link Pointer (BPLP). Contains the address of a QH. This field corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	Typ. You must ensure this field is set to indicate the target data structure is a QH. Any other value in this field yields undefined results.
0	Terminate (T). 1 - Link Pointer field is not valid; that is, the host controller must not use bits [31:5] as a valid memory address. This value also indicates that this FSTN is a Restore Indicator. 0 - Link Pointer is valid; that is, the host controller may use bits [31:5] as a valid memory address. This value also indicates that this FSTN is a Save-Place Indicator.

55.9 Device mode operation

The function of the device operation is to transfer a request to and from the Universal Serial Bus (USB). The device controller performs data transfers using a set of linked list transfer descriptors pointed to by a QH. The next sections explain the use of the device controller from the DCD point-of-view and further describes how specific USB events relate to status changes in the device controller programmer's interface.

55.9.1 Port state and control

From a chip or system reset, the USB OTG module enters the Powered state. A transition from the Powered state to the Attach state occurs when USBCMD[RS] is 1. After receiving a reset on the bus, the port enters the DefaultFS or DefaultHS state in accordance with the protocol reset described in Appendix C.2 of the *Universal Serial Bus Specification, Revision 2.0*. The next figure depicts the states of a USB 2.0 device.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a8-b51-3503ab6240d

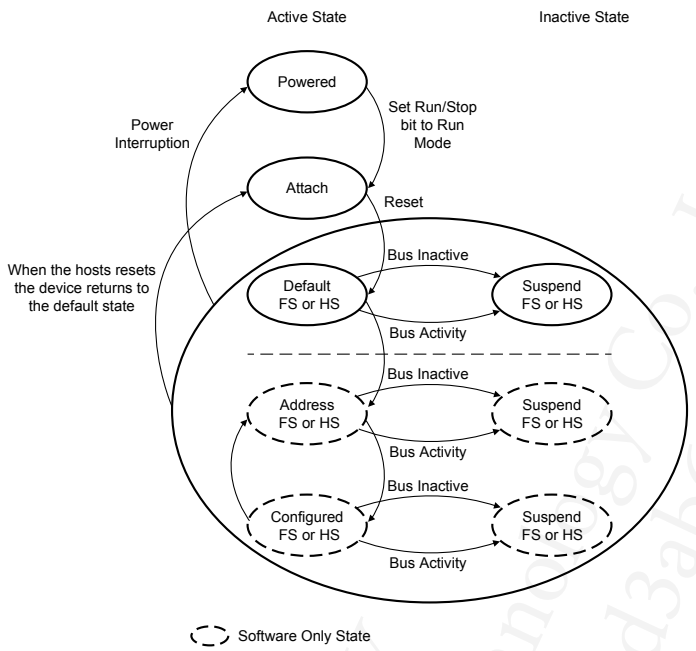


Figure 458. USB 2.0 device states

States Powered, Attach, Default FS or HS, and Suspend FS or HS are implemented in the USB OTG, and they are communicated to the DCD using the following status fields:

Table 430. Device controller state information fields

Field	Register
DC Suspend (SLI)	USBSTS
USB Reset Received (URI)	USBSTS
Port Change Detect (PCI)	USBSTS
HS Port (PSPD)	PORTSCn

The DCD software must maintain a state variable to differentiate between the Default FS or HS state and the Address/Configured states. Change of state from Default to the Address and Configured states is part of the enumeration process described in the Device Framework section of the USB 2.0 specification.

As a result of entering the Address state, the DCD must program the Device Address Register (DEVICEADDR).

Entry into the configured state indicates that all endpoints to be used in the operation of the device have been properly initialized by programming the ENDPTCTRL*n* registers and initializing the associated queue heads.

55.9.2 Bus reset

The host uses a bus reset to initialize downstream devices. When a bus reset is detected, the USB OTG controller does the following:

1. Renegotiates its attachment speed.
2. Resets the device address to 0.

3. Notifies the DCD by interrupt (assuming the USB reset interrupt enable (USBINTR[URE]) is 1).

After a reset is received, all endpoints (except endpoint 0) are disabled and the device controller cancels any primed transactions. The concept of priming is clarified below, but when a reset is received, the DCD must perform the following:

1. Clear all setup token semaphores by reading Endpoint Setup Status (ENDPTSETUPSTAT) and writing the same value back to this register.
2. Clear all the endpoint complete status bits by reading Endpoint Complete (ENDPTCOMPLETE) and writing the same value back to this register.
3. Cancel all primed status by waiting until all bits in Endpoint Initialization (ENDPTPRIME) are 0 and then writing 000F_000Fh to Endpoint De-Initialize (ENDPTFLUSH).
4. Read PORTSCn[PR] and make sure it remains active. A USB reset occurs for a minimum of 3 ms and the DCD must reach this point in the reset clean-up before the end of the reset occurs; otherwise, a hardware reset of the device controller is recommended (rare).
    a. Writing a 1 to USBCMD[RST] can perform a hardware reset.

NOTE

A hardware reset causes the device to detach from the bus by writing 0 to USBCMD[RS]. Therefore, the DCD must completely re-initialize the USB OTG after a hardware reset.

5. Free all allocated dTDs because the device controller no longer executes them. If this is the first time the DCD processes a USB reset event, it is likely no dTDs have been allocated.
6. At this time, the DCD may release control back to the OS because no further changes to the device controller are permitted until a port change detect is indicated.
7. After a port change detect, the device has reached the default state and the DCD can read PORTSCn to determine if the device operates in FS or HS mode. At this time, the device controller has reached normal operating mode and DCD can begin enumeration (see the "Device Framework" chapter of the USB specification).

55.9.3 Suspend/resume

To conserve power, the USB OTG module automatically enters the Suspended state when no bus traffic is observed for a specified period. When suspended, the module maintains any internal status, including its address and configuration. Attached devices must be prepared to suspend any time they are powered, regardless of whether they are assigned a non-default address, are configured, or neither. Bus activity may cease because the host enters a Suspend mode of its own. In addition, a USB device also enters the Suspended state when the hub port it is attached to is disabled.

The USB OTG module exits Suspend mode when there is bus activity. It may also request the host to exit Suspend mode or selectively suspend by using electrical signaling to indicate remote wake-up. The ability of a device to signal remote wake-up is optional. The USB OTG is capable of remote wake-up signaling. When the USB OTG is reset, remote wake-up signaling must be disabled.

55.9.3.1 Suspend operational model

The USB OTG moves into the Suspend state when suspend signaling is detected or activity is missing on the upstream port for more than a specific period. After the device controller enters the Suspend state, an interrupt notifies the DCD (assuming that the device controller suspend interrupt is enabled, by writing 1 to USBINTR[SLE]). When you write 1 to PORTSCn[SUSP], the device controller is suspended.

DCD response when the device controller is suspended is application-specific and may involve switching to a low-power operation. See the USB 2.0 specification for information on the bus power limits in the Suspend state.

55.9.3.2 Resume

If the USB OTG is suspended, its operation resumes when any non-idle signaling is received on its upstream-facing port. In addition, the USB OTG can signal the system to resume operation by forcing resume signaling to the upstream port. Writing 1 to PORTSCn[FPR] when the device is in the Suspend state sends resume signaling upstream. Sending a resume signal to an

upstream port should cause the host to issue resume signaling and bring the suspended bus segment (one or more devices) back to the active condition.

NOTE

Before the use of resume signaling, the host must enable it by using the set feature command defined in the "Device Framework" chapter of the USB 2.0 specification.

55.9.4 Managing endpoints

The USB 2.0 specification defines an endpoint (also called a device endpoint or an address endpoint) as a uniquely addressable portion of a USB device that can source or sink data in a communications channel between the host and the device. The combination of the endpoint number and the endpoint direction specifies endpoint address.

The channel between the host and an endpoint at a specific device represents a data pipe. Endpoint 0 for a device is always a control type data channel used for device discovery and enumeration. Other types of endpoints that are supported by USB include bulk, interrupt, and isochronous. Each endpoint type has specific behavior related to packet response and error managing. See the USB 2.0 specification for more details.

The USB OTG supports up to six endpoint-specified numbers. The DCD can enable, disable, and configure each endpoint except endpoint 0.

Each endpoint direction is essentially independent and can have differing behavior in each direction. For example, the DCD can configure endpoint 1-IN to be a bulk endpoint and endpoint 1-OUT to be an isochronous endpoint. This helps to conserve the total number of endpoints required for device operation. The only exception is that control endpoints must use both directions on a single endpoint number to function as a control endpoint. Endpoint 0, for example, is always a control endpoint and uses both directions.

Each endpoint direction requires a QH allocated in memory. If the maximum is six endpoint numbers (one for each endpoint direction used by the device controller), 12 QHs are required. The operation of an endpoint and use of QHs are described later in this document.

55.9.4.1 Endpoint initialization

After a hardware reset, all endpoints except endpoint 0 are uninitialized and disabled. The DCD must configure and enable each endpoint by writing to the appropriate ENDPTCTRL*n* register. Each register is split into an upper and lower half. The lower half of the register configures the receive or OUT endpoint, and the upper half configures the corresponding transmit or IN endpoint. Control endpoints must have the same configuration in the upper and lower halves of the register; otherwise, behavior is undefined. The next table shows how to construct a configuration word for endpoint initialization.

Table 431. Device controller endpoint initialization

Field	Value
Data Toggle Reset (TXR, RXR)	1 - Synchronize the data PIDs
Data Toggle Inhibit (TXI, RXI)	0 - PID sequencing disabled
Endpoint Type (TXT, RXT)	00 - Control 01 - Isochronous 10 - Bulk 11 - Interrupt
Endpoint Stall (TXS, RXS)	0 - Not stalled

55.9.4.2 Stalling

There are two occasions for which the USB OTG may need to return a STALL response to the host:

- The functional stall is a condition set by the DCD as described in the USB 2.0 Device Framework chapter. A functional stall is used only on non-control endpoints and can be enabled in the device controller by setting the endpoint stall bit in the ENDPTCTRL*n* register associated with the given endpoint and the given direction. In a functional stall condition, the device controller continues to return STALL responses to all transactions occurring on the respective endpoint and direction until the endpoint stall bit is cleared by the DCD.
- A protocol stall, unlike a functional stall, is used on control endpoints and automatically cleared by the device controller at the start of a new control transaction (setup phase). When enabling a protocol stall, the DCD must enable the stall bits as a pair (TXS and RXS bits). A single write to the ENDPTCTRL*n* register can ensure both stall bits are set at the same instant.

NOTE

Any write to the ENDPTCTRL*n* register during operational mode must preserve the endpoint type field (perform a read-modify-write).

Table 432. Device controller stall response matrix

USB packet	Endpoint stall bit	Effect on stall bit	USB response
SETUP packet received by a non-control endpoint	N/A	None	STALL
IN/OUT/PING packet received by a non-control endpoint	1	None	STALL
IN/OUT/PING packet received by a non-control endpoint	0	None	ACK/NAK/NYET
SETUP packet received by a control endpoint	N/A	Cleared	ACK
IN/OUT/PING packet received by a control endpoint	1	None	STALL
IN/OUT/PING packet received by a control endpoint	0	None	ACK/NAK/NYET

55.9.4.3 Data toggle

Data toggle maintains data coherency between host and device for any given data pipe. See the USB 2.0 specification for more information on data toggle.

55.9.4.3.1 Data toggle reset

The DCD may reset the Data Toggle state bit and cause the data toggle sequence to reset in the device controller by setting the Data Toggle reset field in the ENDPTCTRL*n* register. This must only happen when configuring or initializing an endpoint or returning from a STALL condition.

55.9.4.3.2 Data toggle inhibit

This feature is for test purposes only and must never be used during normal device controller operation.

Setting the Data Toggle Inhibit bit causes the USB OTG module to ignore the data toggle pattern normally sent and accepts all incoming data packets regardless of the data toggle state.

In normal operation, the USB OTG checks the DATA0/DATA1 bit against the Data Toggle to determine if the packet is valid. If the data PID does not match the Data Toggle state bit maintained by the device controller for that endpoint, the data toggle is considered not valid. If the data toggle is not valid, the device controller assumes the packet was already received and discards the packet (not reporting it to the DCD). To prevent the USB OTG from re-sending the same packet, the device controller responds to the error packet by acknowledging it with an ACK or NYET response.

55.9.5 Packet transfers

The host initiates all transactions on the USB bus, and in turn the device must respond to any request from the host within the turnaround time stated in the USB 2.0 specification.

A USB host sends requests to the USB OTG in an order that cannot be precisely predicted as a single pipeline, so it is not possible to prepare a single packet for the device controller to execute. However, the order of packet requests is predictable when the endpoint number and direction is considered. For example, if endpoint 3 (transmit direction) is configured as a bulk pipe, expect the host to send IN requests to that endpoint. This USB OTG module prepares packets for each endpoint/direction in anticipation of the host request. The process of preparing the device controller to send or receive data in response to host-initiated transactions on the bus is referred to as priming the endpoint. This term appears throughout the documentation to describe the USB OTG operation so the DCD is built properly. In addition, the term flushing describes the action of clearing a packet queued for execution.

55.9.5.1 Priming transmit endpoints

Priming a transmit endpoint causes the device controller to fetch the dTD for the transaction pointed to by the dQH. After the dTD is fetched, it is stored in the dQH until the device controller completes the transfer described by the dTD. Storing the dTD in the dQH allows the device controller to fetch the operating context needed to manage a request from the host without the need to follow the linked list, starting at the dQH when the host request is received.

After the device has loaded the dTD, the leading data in the packet is stored in a FIFO in the device controller. This FIFO splits into virtual channels so the leading data can be stored for any endpoint up to the maximum number of endpoints configured at device synthesis time.

After a priming request is complete, an endpoint state of "primed" is indicated in Endpoint Status (ENDPTSTAT). For a primed transmit endpoint, the device controller can respond to an IN request from the host and meet the stringent bus turnaround time of HS USB.

55.9.5.2 Priming receive endpoints

Priming receives endpoints identical to priming of transmit endpoints from the point of view of the DCD. The major difference in the operational model at the device controller is no data movement of the leading packet data because the data is to be received from the host.

As part of the architecture, the FIFO for the receive endpoints is not partitioned into multiple channels like the transmit FIFO. Thus, the size of the RX FIFO does not scale with the number of endpoints.

55.9.5.3 Interrupt/bulk endpoint operation

The behaviors of the device controller for interrupt and bulk endpoints are identical. All valid IN and OUT transactions to bulk pipes handshake with a NAK unless the endpoint is primed. After the endpoint is primed, data delivery commences.

A dTD is retired by the device controller when the packets described in the TD are completed. Each dTD describes N packets to transfer according to the USB variable length transfer protocol. The formula below and the next table describe how the device controller computes the number and length of the packets sent/received by the USB, which varies according to the total number of bytes and maximum packet length. See Endpoint capabilities and characteristics (offset = 0h) for details on the ZLT bit.

With zero-length termination (ZLT) cleared:

N = INT(number of bytes/maximum packet length) + 1

With zero-length termination (ZLT) set:

N = MAXINT(number of bytes/maximum packet length)

Table 433. Variable length transfer protocol example (ZLT=0)

Bytes (dTD)	Maximum packet length (dQH)	N	P1	P2	P3
511	256	2	256	255	-
512	256	3	256	256	0
512	512	2	512	0	-

Table 434. Variable length transfer protocol example (ZLT=1)

Bytes (dTD)	Maximum packet length (dQH)	N	P1	P2	P3
511	256	2	256	255	-
512	256	2	256	256	-
512	512	1	512	-	-

TX-dTD is complete when:

- All packets described in the dTD successfully transmit. Total bytes in dTD equal 0 when this occurs.

RX-dTD is complete when:

- All packets described in the dTD are successfully received. Total bytes in dTD equal 0 when this occurs.
- A short packet (number of bytes < maximum packet length) was received.

  This is a successful transfer completion; DCD must check the total bytes field in the dTD to determine the number of bytes remaining. From the total bytes remaining in the dTD, the DCD can compute the actual bytes received.
- A long packet was received (number of bytes > maximum packet size) or (total bytes received > total bytes specified).

  This is an error condition. The device controller discards the remaining packets and sets the buffer error bit in the dTD. In addition, the endpoint flushes and the USBERR interrupt becomes active.

NOTE

Disabling zero-length packet termination allows transfers larger than the total bytes field spanning across two or more dTDs.

Upon successful completion of the packet(s) described by the dTD, the active bit in the dTD is cleared and the next pointer is followed when the terminate bit is clear. When the terminate bit is set, the USB OTG flushes the endpoint/direction and ceases operations for that endpoint/direction.

Upon unsuccessful completion of a packet (see long packet above), the dQH remains pointing to the dTD in error. To recover from this error condition, DCD must properly re-initialize the dQH by clearing the active bit and update the nextTD pointer before attempting to re-prime the endpoint.

NOTE

All packet level errors, such as a missing handshake or CRC error, are retried automatically by the device controller. There is no required interaction with the DCD for managing such errors.

Table 435. Interrupt/bulk endpoint bus response matrix

Token type	Stall	Not primed	Primed	Underflow	Overflow
Setup	Ignore	Ignore	Ignore	N/A	N/A
In	STALL	NAK	Transmit	BS Error[1]	N/A
Out	STALL	NAK	Receive + NYET/ACK[2]	N/A	NAK
Ping	STALL	NAK	ACK	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

1. Force bit stuff error
2. NYET/ACK - NYET; or, if the TD has packets remaining according to the USB variable length protocol, then ACK.

55.9.6 Control endpoint operation

55.9.6.1 Setup phase

All requests to a control endpoint begin with a setup phase followed by an optional data phase and a required status phase.

Setup packet managing:

- Disable setup lockout by writing 1 to USBMODE[SLOM] at initialization. Setup lockout is not necessary when using the tripwire as described below.

NOTE

Leaving the setup lockout mode cleared results in a potential compliance issue.

- After receiving an interrupt and inspecting Endpoint Setup Status (ENDPTSETUPSTAT) to determine a setup packet was received on a particular pipe:
  1. Write 1 to clear the corresponding field in Endpoint Setup Status (ENDPTSETUPSTAT).
  2. Set the setup tripwire field (USBCMD[SUTW]).
  3. Duplicate contents of dQH.SetupBuffer into a local software byte array.
  4. Read USBCMD[SUTW]. If set to 1, continue; if cleared to 0, go to step 2.
  5. Clear USBCMD[SUTW].
  6. Poll Endpoint Setup Status (ENDPTSETUPSTAT) until it clears to 0.
  7. Process the setup packet using the local software byte array copy and execute status/handshake phases.

NOTE

After receiving a new setup packet, status and/or handshake phases may remain pending from a previous control sequence. These must be flushed and de-allocated before linking a new status and/or handshake dTD for the most recent setup packet.

55.9.6.2 Data phase

Following the setup phase, the DCD must create a dTD for the data phase and prime the transfer.

After priming the packet, the DCD must verify a new setup packet is not received by reading Endpoint Setup Status (ENDPTSETUPSTAT) immediately, verifying that the prime had completed. A prime completes when the associated field in Endpoint Initialization (ENDPTPRIME) is cleared to 0 and the associated field in Endpoint Status (ENDPTSTAT) is set to 1. If Endpoint Initialization (ENDPTPRIME) goes to 0 and Endpoint Status (ENDPTSTAT) is not set to 1, the prime fails. This can only happen

because of improper setup of the dQH, dTD, or a setup arriving during the prime operation. If a new setup packet is indicated after Endpoint Initialization (ENDPTPRIME) is cleared to 0, then the TD can be freed and the DCD must reinterpret the setup packet.

Should a setup arrive after the data stage is primed, the device controller automatically clears the prime status (Endpoint Status (ENDPTSTAT)) to enforce data coherency with the setup packet.

NOTE

Error managing of data phase packets is the same as for bulk packets described previously.

55.9.6.3 Status phase

Similar to the data phase, the DCD must create a TD (with byte length equal to 0) and prime the endpoint for the status phase. The DCD must also perform the same checks of Endpoint Setup Status (ENDPTSETUPSTAT) as described above in the data phase.

NOTE

Error managing of status phase packets is the same as bulk packets described previously.

55.9.6.4 Control endpoint bus response matrix

The next table shows the device controller response to packets on a control endpoint according to the device controller state.

Table 436. Control endpoint bus response matrix

Token type	Endpoint state					Setup lockout
	Stall	Not primed	Primed	Underflow	Overflow	
Setup	ACK	ACK	ACK	N/A	SYSERR[1]	
In	STALL	NAK	Transmit	BS Error[2]	N/A	N/A
Out	STALL	NAK	Receive + NYET/ACK[3]	N/A	NAK	N/A
Ping	STALL	NAK	ACK	N/A	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore	Ignore

1. SYSERR - System error must never occur when the latency FIFOs are correctly sized and the DCD is responsive.
2. Force bit stuff error.
3. NYET/ACK - NYET; or, if the TD has packets remaining according to the USB variable length protocol, then ACK.

55.9.7 Isochronous endpoint operation

Isochronous endpoints are used for real-time scheduled delivery of data, and their operational model is significantly different than the host throttled bulk, interrupt, and control data pipes. Real time delivery by the USB OTG is accomplished by:

- Exactly MULT packets per (micro)frame are transmitted/received.

NOTE

MULT is a 2-bit field in the dQH. Isochronous endpoints do not use the variable length packet protocol.

- NAK responses are not used. Instead, zero-length packets are sent in response to an IN request to unprimed endpoints. For unprimed RX endpoints, the response to an OUT transaction is to ignore the packet within the device controller.
- Prime requests always schedule the transfer described in the dTD for the next (micro)frame. If ISO-dTD remains active after that frame, ISO-dTD holds ready until executed or canceled by the DCD.

The USB OTG in Host mode uses the periodic frame list to schedule data exchanges to isochronous endpoints. The operational model for Device mode does not use such a data structure. Instead, the same dTD used for control/bulk/interrupt endpoints is also used for isochronous endpoints. The difference is in the managing of the dTD.

The first difference between bulk and ISO-endpoints is that priming an ISO-endpoint is a delayed operation such that an endpoint becomes primed only after a SOF is received. After the DCD writes the prime bit, the prime bit clears as usual to indicate to you that the device controller completed priming the dTD for transfer. Internal to the design, the device controller hardware masks that prime start until the next frame boundary. This behavior is hidden from the DCD, but occurs so the device controller can match the dTD to a specific (micro)frame.

Another difference with isochronous endpoints is that the transaction must wholly complete in a (micro)frame. After an ISO transaction is started in a (micro)frame, it retires the corresponding dTD when MULT transactions occur or the device controller finds a fulfillment condition.

The transaction error bit set in the status field indicates a fulfillment error condition. When a fulfillment error occurs, the frame after the transfer failed to complete wholly, and the device controller retires the current ISO-dTD and moves to the next ISO-dTD.

Fulfillment errors are only caused due to partially completed packets. If no activity occurs to a primed ISO-dTD, the transaction stays primed indefinitely. This means you must discard transmit ISO-dTDs that pile up from a failure of the host to move the data.

Finally, the last difference with ISO packets is in the data level error managing. When a CRC error occurs on a received packet, the packet is not retried as is done for bulk and control endpoints. Instead, the CRC is noted by setting the transaction error bit and the data is stored as usual for you to sort out.

- TX packet retired:
  - MULT counter reaches zero.
  - Fulfillment error (Transaction Error bit is 1):
    - # packets occurred > 0 AND # packets occurred < MULT

    NOTE

    For TX-ISO, MULT counter can be loaded with a lesser value in the dTD multiplier override field. If the multiplier override field is 0, the MULT counter initializes to the multiplier in the QH.
- RX packet retired:
  - MULT counter reaches zero.
  - Non-MDATA data PID is received.
  - Overflow error (Buffer Error bit is 1):
    - Packet received is > maximum packet length.
    - Packet received exceeds total bytes allocated in dTD.
  - Fulfillment error (Transaction Error bit is 1):
    - # packets occurred > 0 AND # packets occurred < MULT
  - CRC error (Transaction Error bit is 1)

    NOTE

    For ISO, when a dTD is retired, the next dTD is primed for the next frame. For continuous (micro)frame to (micro)frame operation, DCD must ensure the dTD linked list is out ahead of the device controller by at least two (micro)frames.

55.9.7.1 Isochronous pipe synchronization

When it is necessary to synchronize an isochronous data pipe to the host, the (micro)frame number (USB Frame Index (FRINDEX)) can act as a marker. To cause a packet transfer to occur at a specific (micro)frame number (N), the DCD must interrupt on SOF during frame N-1. When USB Frame Index (FRINDEX) equals N-1, the DCD must write the prime bit. The USB OTG primes the isochronous endpoint in (micro)frame N-1 so the device controller executes delivery during (micro)frame N.

CAUTION

Priming an endpoint towards the end of (micro)frame N-1 does not guarantee delivery in (micro)frame N. The delivery may actually occur in (micro)frame N+1 if the device controller does not have enough time to complete the prime before the SOF for packet N is received.

55.9.7.2 Isochronous endpoint bus response matrix

Table 437. Isochronous endpoint bus response matrix

Token type	Stall	Not primed	Primed	Underflow	Overflow
Setup	STALL	STALL	STALL	N/A	N/A
In	NULL[1] packet	NULL packet	Transmit	BS error[2]	N/A
Out	Ignore	Ignore	Receive	N/A	Drop packet
Ping	Ignore	Ignore	Ignore	Ignore	Ignore
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

1. Zero length packet
2. Force bit stuff error

55.9.8 Managing QHs

The dQH points to the linked list of transfer tasks, each depicted by the dTD. An area of memory pointed to by ENDPOINTLISTADDR register contains a group of every dQH in a sequential list (see the next figure). The even elements in the dQH list receive endpoints (OUT/SETUP) and the odd elements transmit endpoints (IN/INTERRUPT). dTDs are linked head to tail starting at the QH and ending at a terminate bit. After the dTD retires, it is no longer part of the linked list from the QH. Therefore, you are required to track all transfer descriptors because pointers no longer exist within the QH after the dTD is retired (see Software link pointers).

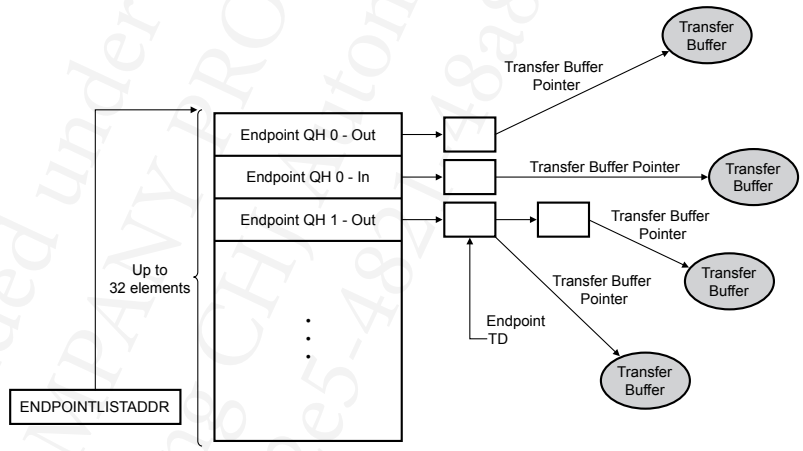


Figure 459. Endpoint QH diagram

In addition to current and next pointers and the dTD overlay examined in Packet transfers, the dQH also contains the following parameters for the associated endpoint:

- Multiplier
- Maximum packet length

- Interrupt on setup

The next section includes a demonstration of complete initialization of the dQH including these fields.

55.9.8.1 QH initialization

One dQH pair must be initialized for each active endpoint. To initialize a dQH:

1. Write to the wMaxPacketSize field as required by the USB specification chapter 9 or application-specific protocol.
2. Write the multiplier field to 0 for control, bulk, and interrupt endpoints. For ISO endpoints, set the multiplier to 1, 2, or 3 as required for bandwidth with the USB specification chapter 9 protocol. In FS mode, the multiplier field can only be 1 for ISO endpoints.
3. Set the next dTD terminate bit field.
4. Clear the active bit in the status field.
5. Clear the halt bit in the status field.

NOTE

The DCD must only modify dQH if the associated endpoint is not primed and there are no outstanding dTDs.

55.9.8.2 Setup transfers operation

As discussed in Control endpoint operation, setup transfers require special treatment by the DCD. A setup transfer does not use a dTD, but instead stores the incoming data from a setup packet in an 8-byte buffer within the dQH.

Upon receiving notification of the setup packet, the DCD must manage the setup transfer by:

1. Copying setup buffer contents from dQH-RX to the software buffer.
2. Acknowledging setup backup by writing a 1 to the corresponding field in Endpoint Setup Status (ENDPTSETUPSTAT).

NOTE

The acknowledge must occur before continuing to process the setup packet. After acknowledge occurs, DCD must not attempt to access the setup buffer in dQH-RX. Only the local software copy should be examined.

3. Checking for pending data or status dTDs from previous control transfers and flushing any that exist as discussed in Flushing/de-priming an endpoint.

NOTE

It is possible for the device controller to receive setup packets before previous control transfers complete. Existing control packets in progress must be flushed and the new control packet completed.

4. Decoding the setup packet and prepare data phase (optional) and status phase transfer as required by the USB specification chapter 9 or application-specific protocol.

55.9.9 Managing transfers with transfer descriptors

55.9.9.1 Software link pointers

It is necessary for the DCD software to maintain head and tail pointers for the linked list of dTDs for each respective QH. This is necessary because the dQH only maintains pointers to the current working dTD and the next dTD executed. The operations described in the next section for managing dTDs assumes the DCD can reference the head and tail of the dTD linked list.

NOTE

To conserve memory, the reserved fields at the end of the dQH can be used to store the head and tail pointers, but the DCD must continue maintaining the pointers.

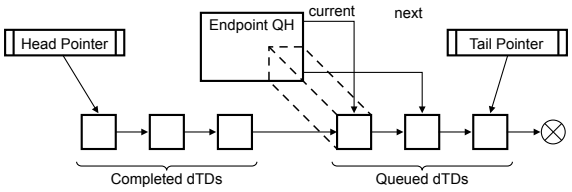


Figure 460. Software link pointers

NOTE

Check the status of each dTD to determine completed status.

55.9.9.2 Building a TD

Before a transfer can be executed from the linked list, a dTD must be built to describe the transfer. You use the following procedure for building dTDs.

First, you allocate a 32-byte dTD block of memory aligned to 32-byte boundaries. The last 5 bits of the address must equal 00000.

Then, you need to do the following:

1. Initialize the first 7 entries (28 bytes) to 0.
2. Set the terminate field.
3. Fill in total bytes with transfer size.
4. Set the IOC field if desired.
5. Initialize the Status field with the Active field set to 1, and all remaining status bits cleared to 0.
6. Fill in buffer pointer page 0 and the current offset to point to the start of the data buffer.
7. Initialize buffer pointer page 1 through page 4 to be one greater than each of the previous buffer pointers.

55.9.9.3 Executing a TD

To safely add a dTD, the DCD must follow this procedure that manages the event where the device controller reaches the end of the dTD list. At the same time, a new dTD is added to the end of the list.

The DCD needs to determine whether the linked list is empty by checking the DCD driver to see if the pipe is empty (the internal representation of the linked list must indicate if any packets are outstanding).

Case 1: Linked list is empty.

1. Write the dQH next pointer AND dQH terminate bit to 0 as a single 32-bit operation.
2. Clear the active and halt bits in the dQH (in case set from a previous error).
3. Prime the endpoint by writing 1 to the correct bit position in Endpoint Initialization (ENDPTPRIME).

Case 2: Linked list is not empty.

1. Add the dTD to the end of the linked list.
2. Read the correct prime bit in Endpoint Initialization (ENDPTPRIME); if 1, then we are done.
3. Write 1 to USBCMD[ATDTW].
4. Read the correct status bit in Endpoint Status (ENDPTSTAT), and store it in a temporary variable for later.
5. Read USBCMD[ATDTW]:

   If it is 0, go to step 3.

   If it is 1, continue to step 6.

6. Write 0 to USBCMD[ATDTW].
7. If the status bit read in step 4 is 1, then DONE.
8. If the status bit read in step 4 is 0, then go to step 1 in case 1.

55.9.9.4 Transfer completion

After a dTD is initialized and the associated endpoint is primed, the device controller executes the transfer upon the host-initiated request. The DCD is notified with a USB interrupt if the IOC field was set, or alternatively, the DCD can poll the endpoint complete register to determine when the dTD was executed. After a dTD is executed, the DCD can check the status bits to determine success or failure.

CAUTION

Multiple dTDs can be completed in a single endpoint complete notification. After clearing the notification, the DCD must search the dTD linked list and retire all finished (active bit cleared) dTDs.

By reading the status fields of the completed dTDs, the DCD can determine if the transfers completed successfully. Success is determined with the following combination of status bits:

- Active = 0
- Halted = 0
- Transaction error = 0
- Data buffer error = 0

Should any combination other than the one shown above exist, the DCD must take proper action. Transfer failure mechanisms are indicated in Device error matrix.

In addition to checking the status bit, the DCD must read the transfer bytes field to determine the actual bytes transferred. When a transfer is complete, the total bytes transferred decrements by the actual bytes transferred. For transmit packets, a packet is only complete after the actual bytes reaches zero. However, for receive packets, the host may send fewer bytes in the transfer according to the USB variable length packet protocol.

55.9.9.5 Flushing/de-priming an endpoint

It is necessary for the DCD to flush or de-prime endpoints during a USB device reset or during a broken control transfer. There may also be application-specific requirements to stop transfers in progress. The DCD can use this procedure to stop a transfer in progress:

1. Set the corresponding bit(s) in Endpoint De-Initialize (ENDPTFLUSH).
2. Wait until all bits in Endpoint De-Initialize (ENDPTFLUSH) are cleared to 0.

   NOTE

   This operation may take a large amount of time depending on the USB bus activity. It is not desirable to have this wait loop within an interrupt service routine.

3. Read Endpoint Status (ENDPTSTAT) to ensure that, for all endpoints commanded to be flushed, the corresponding bits are now cleared. If the corresponding bits are set after step #2 has finished, the flush failed as described below:

   In very rare cases, a packet is in progress to the particular endpoint when commanded to flush using Endpoint De-Initialize (ENDPTFLUSH). A safeguard is in place to refuse the flush to ensure that the packet in progress completes successfully. The DCD may need to repeatedly flush any endpoints that fail to flush by repeating steps 1-3 until each endpoint successfully flushes.

55.9.9.6 Device error matrix

The following table summarizes packet errors not automatically managed by the USB OTG module.

Table 438. Device error matrix

Error	Direction	Packet type	Data buffer error bit	Transaction error bit
Data buffer overflow	RX	Any	1	0
ISO packet error	RX	ISO	0	1
ISO fulfillment error	Both	ISO	0	1

The device controller manages all errors on bulk/control/interrupt endpoints except for a data buffer overflow. However, for ISO endpoints, errors packets are not retried and errors are tagged as indicated.

Table 439. Error descriptions

Overflow	Number of bytes received exceeded the maximum packet size or total buffer length. **NOTE** This error also sets the halt bit in the dQH, and if there are dTDs remaining in the linked list for the endpoint, those are not executed.
ISO packet error	CRC error on received ISO packet. Contents are not guaranteed to be correct.
ISO fulfillment error	Host failed to complete the number of packets defined in the dQH Mult field within the given (micro)frame. For scheduled data delivery, the DCD may need to readjust the data queue because a fulfillment error causes the device controller to cease data transfers on the pipe for one (micro)frame. During the dead (micro)frame, the device controller reports an error on the pipe and primes for the following frame.

55.9.10 Servicing interrupts

The interrupt service routine programmer must understand that there are high-frequency, low-frequency, and error operations to order accordingly.

55.9.10.1 High frequency interrupts

In particular, high frequency interrupts must be managed in the following order. The most important of these is listed first because the DCD must acknowledge a setup buffer in the timeliest manner possible.

Table 440. Interrupt managing order

Execution order	Interrupt	Action
1a	USB interrupt[1] ENDPTSETUPSTAT	Copy the contents of the setup buffer and acknowledge the setup packet (as indicated in Managing QHs). Process the setup packet according to USB specification chapter 9 or application-specific protocol.
1b	USB interrupt ENDPTCOMPLETE	Manage completion of the dTD as indicated in Managing QHs.
2	SOF interrupt	Action as deemed necessary by application. This interrupt may not have a use in all applications.

1. It is likely that multiple interrupts stack up on any call to the interrupt service routine and during the interrupt service routine.

55.9.10.2 Error interrupts

Error interrupts are the least frequent and should be placed last in the interrupt service routine.

Table 441. Error interrupt events

Interrupt	Action
USB error interrupt	This error is redundant because it combines USB interrupt and an error status in the dTD. The DCD more aptly manages packet-level errors by checking the dTD status field upon receipt of USB interrupt (with Endpoint Complete (ENDPTCOMPLETE)).
System error	Unrecoverable error. Immediate reset of module; free transfer buffers in progress and restart the DCD.

55.10 Host mode operation

The EHCI specification defines the general operational model for a USB module in Host mode. The EHCI specification describes the register-level interface for a host controller for USB Revision 2.0. It includes a description of the hardware/software interface between system software and host controller hardware. The next section has information about the initialization of the USB modules; however, full details of the EHCI specification are beyond the scope of this document.

55.10.1 Servicing interrupts

The interrupt service routine programmer must understand that there are high-frequency, low-frequency, and error operations to order accordingly.

55.10.1.1 High frequency interrupts

In particular, high frequency interrupts must be managed in the following order. The most important of these is listed first because the DCD must acknowledge a setup buffer in the timeliest manner possible.

Table 442. Interrupt managing order

Execution order	Interrupt	Action
1a	USB interrupt[1] ENDPTSETUPSTAT	Copy the contents of the setup buffer and acknowledge the setup packet (as indicated in Managing QHs). Process the setup packet according to USB specification chapter 9 or application-specific protocol.
1b	USB interrupt ENDPTCOMPLETE	Manage completion of the dTD as indicated in Managing QHs.
2	SOF interrupt	Action as deemed necessary by application. This interrupt may not have a use in all applications.

1. It is likely multiple interrupts stack up on any call to the interrupt service routine and during the interrupt service routine.

55.10.1.2 Low frequency interrupts

The low frequency events include the following interrupts. These interrupts can be managed in any order because they do not occur often in comparison to the high-frequency interrupts.

Table 443. Low frequency interrupt events

Interrupt	Action
Port change	Change software state information.
Sleep enable (Suspend)	Change software state information. Low-power managing as necessary.
Reset received	Change software state information. Abort pending transfers.

55.10.1.3 Error interrupts

Error interrupts are the least frequent and should be placed last in the interrupt service routine.

Table 444. Error interrupt events

Interrupt	Action
USB error interrupt	This error is redundant because it combines USB interrupt and an error status in the dTD. The DCD more aptly manages packet-level errors by checking the dTD status field upon receipt of USB interrupt (with Endpoint Complete (ENDPTCOMPLETE)).
System error	Unrecoverable error. Immediate reset of module; free transfer buffers in progress and restart the DCD.

55.10.2 Managing isochronous transfers using iTDs

The structure of an iTD is presented in iTD. The four distinct sections to an iTD are:

- Next link pointer - This first field is for schedule linkage purposes only.
- Transaction description array - This area is an eight-element array. Each element represents control and status information for one microframe's worth of transactions for a single HS isochronous endpoint.
- The buffer page pointer array - This is a 7-element array of physical memory pointers to data buffers. These are 4 KB aligned pointers to physical memory.
- Endpoint capabilities - This area utilizes the unused low-order 12 bits of the buffer page pointer array. The fields in this area are used across all transactions executed for this iTD, including endpoint addressing, transfer direction, maximum packet size, and high-bandwidth multiplier.

55.10.2.1 Host controller operational model for iTDs

The host controller uses USB Frame Index (FRINDEX) register bits [12:3] to index into the periodic frame list. This means that the host controller visits each frame list element eight consecutive times before incrementing to the next periodic frame list element. Each iTD contains eight transaction descriptions, which map directly to USB Frame Index (FRINDEX) register bits [2:0]. Therefore, each transaction descriptor corresponds to one microframe. Each iTD can span eight microframes worth of transactions.

When the host controller fetches an iTD, it uses USB Frame Index (FRINDEX) register bits [2:0] to index into the transaction description array.

If the value of the Active field in the Status field of the indexed transaction description is set to 0, the host controller ignores the iTD and follows the next pointer to the next schedule data structure.

When the value of the indexed Active field is 1, the host controller continues to parse the iTD. It stores the indexed transaction description and the general endpoint information (device address, endpoint number, maximum packet size, and so on). It also uses the Page Select (PG) field to index the buffer pointer array, storing the selected buffer pointer and the next sequential buffer pointer. For example, if the PG field is 0, then the host controller stores Page 0 and Page 1.

The host controller constructs a physical data buffer address by concatenating the current buffer pointer (as selected using the current transaction description's PG field) and the transaction description's Transaction Offset field. The host controller uses

the endpoint addressing information and I/O-bit to execute a transaction to the appropriate endpoint. When the transaction is complete, the host controller clears the Active field to 0 and writes back any additional status information to the Status field in the currently selected transaction description.

The data buffer associated with the iTD must be virtually contiguous memory. Seven page pointers are provided to support eight high-bandwidth transactions regardless of the starting packet's offset alignment into the first page. A starting buffer pointer (physical memory address) is constructed by concatenating the page pointer (for example, page 0 pointer) selected by the active transaction description's PG (for example, value: 00B) field with the transaction offset field. As the transaction moves data, the host controller must detect when an increment of the current buffer pointer crosses a page boundary. When this occurs, the host controller simply replaces the current buffer pointer's page portion with the next page pointer (for example, page 1 pointer) and continues to move data. The size of each bus transaction is determined by the value in the Maximum Packet Size field. An iTD supports high-bandwidth pipes through the Mult (multiplier) field. When the Mult field is 1, 2, or 3, the host controller executes the specified number of bus transactions of maximum packet size for the endpoint in the current microframe. In other words, the Mult field represents a T-Count for the endpoint in the current microframe. If the Mult field is 0, the operation of the host controller is undefined. The transfer description is used to service all transactions indicated by the Mult field.

For OUT transfers:

- The value of the Transaction X Length field represents the total bytes to be sent during the microframe.
- You must set the Mult field to be consistent with Transaction X Length and Maximum Packet Size.
- The host controller:
  - Sends the bytes in portions that are of Maximum Packet Size
  - Decrements its local copy of Transaction X Length by Maximum Packet Size after each transaction
  - Always sends a number of bytes that are of Maximum Packet Size or Transaction X Length, whichever is less
  - Advances the transfer state in the transfer description
  - Updates the appropriate record in the iTD
  - Moves to the next schedule data structure
- The maximum sized transaction supported is 3 x 1024 bytes.

For IN transfers:

- The host controller issues Mult transactions.
- It is assumed that you have properly initialized the iTD to accommodate all of the possible data.
- During each IN transaction, the host controller:
  - Must use the maximum packet size during each IN transaction to detect packet babble errors
  - Keeps the sum of bytes received in the Transaction X Length field
- After all transactions for the endpoint have completed for the microframe, Transaction X Length contains the total number of bytes received. If the final value of Transaction X Length is less than the value of Maximum Packet Size:
  - Less data than was allowed for was received from the associated endpoint.
  - This short packet condition does not write 1 to USBSTS[UI].
  - The host controller will not detect this condition.
- If the device sends more than Transaction X Length or Maximum Packet Size bytes (whichever is less):
  - The host controller sets the Babble Detected field to 1 and sets the Active field to 0.
  - Note that the host controller is not required to update the iTD Transaction X Length field in this error scenario.
- If the value of the Mult field is greater than 1, then the host controller automatically executes the value of Mult transactions. The host controller does not execute all Mult transactions if:
  - The endpoint is an OUT and Transaction X Length goes to zero before all the Mult transactions have executed (run out of data), or

— The endpoint is an IN and the endpoint delivers a short packet, or an error occurs on a transaction before Mult transactions have been executed. The end of microframe may occur before all of the transaction opportunities have been executed. When this happens, the transfer state of the transfer description is advanced to reflect the progress that was made, the result written back to the iTD and the host controller proceeds to processing the next microframe. See Appendix D for a table summary of the host controller required behavior for all the high-bandwidth transaction cases.

55.10.2.2 Software operational model for iTDs

A client buffer request to an isochronous endpoint may span 1 to N microframes. When N is larger than one, system software may have to use multiple iTDs to read or write data with the buffer (if N is larger than eight, it must use more than one iTD).

The following figure illustrates how a client buffer is mapped by system software to the periodic schedule (that is, the periodic frame list and a set of iTDs).

- On the right side of this figure is the client description of its request. The description includes a buffer base address plus additional annotations to identify which portions of the buffer must be used with each bus transaction.
- In the middle are the iTD data structures used by the system software to service the client request. Each iTD can be initialized to service up to 24 transactions, organized into eight groups of up to three transactions each. Each group maps to one microframe's worth of transactions. The EHCI controller does not provide per-transaction results within a microframe. It treats the per-microframe transactions as a single logical transfer.
- On the left is the host controller's Frame List. System software establishes references from the appropriate locations in the Frame List to each of the appropriate iTDs. If the buffer is large, then system software can use a small set of iTDs to service the entire buffer.

System software can activate the transaction description records (contained in each iTD) in any pattern required for the particular data stream.

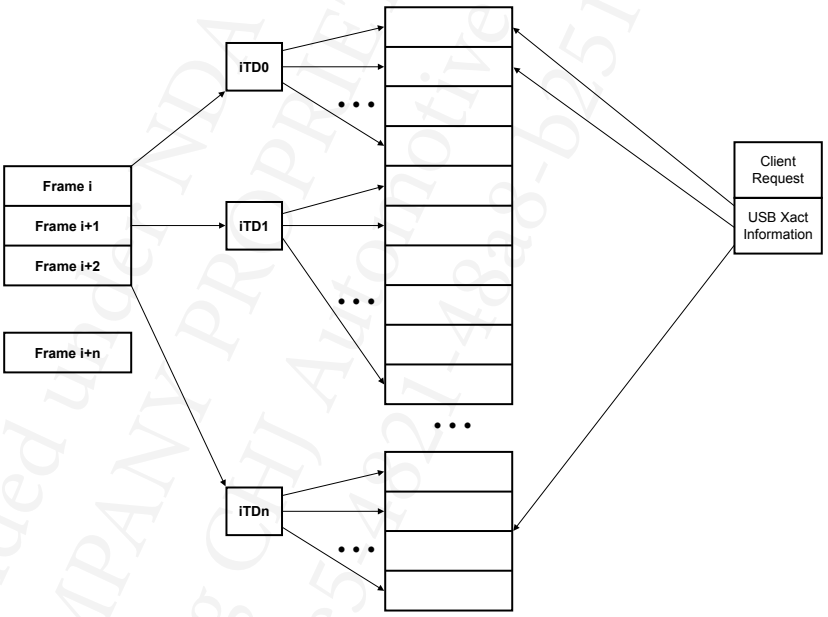


Figure 461. Example association of iTDs to client request buffer

As noted above, the client request includes a pointer to the base of the buffer and offsets into the buffer to annotate which buffer sections are to be used on each bus transaction that occurs on this endpoint. System software must initialize each transaction description in an iTD to ensure it uses the correct portion of the client buffer. For example, for each transaction description, the PG field is set to index the correct physical buffer page pointer and the Transaction Offset field is set relative to the correct buffer pointer page (for example, the same one referenced by the PG field). When the host controller executes a transaction it selects a transaction description record based on FRINDEX[2:0]. It then uses the current Page Buffer Pointer (as selected by the PG field) and concatenates to the Transaction Offset field. The result is a starting buffer address for the transaction. As the host controller

moves data for the transaction, it must watch for a page wrap condition and properly advance to the next available Page Buffer Pointer. System software must not use the Page 6 buffer pointer in a transaction description where the length of the transfer wraps a page boundary. Doing so yields undefined behavior. The host controller hardware is not required to 'alias' the page selector to Page 0. USB 2.0 isochronous endpoints can specify a period greater than one. You can achieve the appropriate scheduling by linking iTDs into the appropriate frames (relative to the Frame List) and by setting appropriate transaction description elements active bits to 1.

55.10.2.2.1 Periodic scheduling threshold

HCCPARAMS[IST] is an indicator to system software as to how the host controller pre-fetches and effectively caches schedule data structures.

It is used by system software when adding isochronous work items to the periodic schedule. The value of this field indicates to system software the minimum distance it can update isochronous data (relative to the current location of the host controller execution in the periodic list) and still have the host controller process them.

Each of the iTD and siTD data structures describe 8 microframes of transactions. The host controller is allowed to cache one (or more) of these data structures to reduce memory traffic. Three basic caching models account for the fact that the isochronous data structures span eight microframes. Those are:

- No caching
- Microframe caching
- Frame caching

When you add new isochronous transactions to the schedule, it always performs a read of USB Frame Index (FRINDEX) to determine the frame and microframe the host controller is currently executing. Of course, there is no information about where in the microframe the host controller is, so a constant uncertainty-factor of one microframe must be assumed. Combining the knowledge of where the host controller is executing with the knowledge of the caching model allows the definition of simple algorithms for how closely software can reliably work to the executing host controller.

No caching is indicated with a value of zero in the Isochronous Scheduling Threshold field. The host controller may pre-fetch data structures during a periodic schedule traversal (per microframe) but always dumps any accumulated schedule state at the end of the microframe. At the appropriate time relative to the beginning of every microframe, the host controller always begins schedule traversal from the frame list. You can use the value of USB Frame Index (FRINDEX) (plus the constant uncertainty-factor of one microframe) to determine the approximate position of the executing host controller. When no caching is selected, you can add an isochronous transaction as near as two microframes in front of the current executing position of the host controller.

Frame caching is indicated with a non-zero value in bit [7] of the Isochronous Scheduling Threshold field. In the frame-caching model, system software assumes that the host controller caches one (or more) isochronous data structures for an entire frame (eight microframes). You use the value of USB Frame Index (FRINDEX) (plus the constant uncertainty-factor of one microframe) to determine the current microframe/frame (assume modulo 8 arithmetic in adding the constant 1 to the microframe number). For any current frame N, if the current microframe is 0 to 6, then you can safely add isochronous transactions to frame N + 1. If the current microframe is 7, then you can add isochronous transactions to frame N + 2.

Microframe caching is indicated with a non-zero value in the least-significant 3 bits of the Isochronous Scheduling Threshold field. System software assumes the host controller caches one or more periodic data structures for the number of microframes indicated in the Isochronous Scheduling Threshold field. For example, if the count value were 2, then the host controller keeps a window of two microframes worth of state (the current microframe, plus the next frame) on-chip. On each microframe boundary, the host controller releases the current microframe state and begins accumulating the next microframe state.

55.10.3 Managing control/bulk/interrupt transfers through QHs

This section presents an overview of how the host controller interacts with queuing data structures.

QHs use the qTD structure. One QH is used to manage the data stream for one endpoint. The QH structure contains static endpoint characteristics and capabilities. It also contains a working area from where individual bus transactions for an endpoint are executed (see the Overlay area, defined in QH). Each qTD represents one or more bus transactions, which is defined in the context of this specification as a transfer.

The general processing model for the host controller's use of a QH is this:

1. Read a QH
2. Execute a transaction from the overlay area
3. Write back the results of the transaction to the overlay area
4. Move to the next QH

If the host controller encounters errors during a transaction, the host controller sets one (or more) of the error reporting bits in the QH's Status field. The Status field accumulates all errors encountered during the execution of a qTD (for example, the error bits in the QH Status field are "sticky" until the transfer (qTD) has completed). This state is always written back to the source qTD when the transfer is complete. On transfer (for example, buffer or halt conditions) boundaries, the host controller must auto-advance (without software intervention) to the next qTD. Additionally, the hardware must be able to halt the queue so that no additional bus transactions occur for the endpoint and the host controller does not advance the queue.

An example host controller operational state machine of a QH traversal is illustrated in the following figure. This state machine is a model for how a host controller should traverse a QH. The host controller must be able to advance the queue from the Fetch QH state in order to avoid all hardware/software race conditions. This simple mechanism allows you to simply link qTDs to the QH and activate them, then the host controller can always find them if/when they are reachable. The figure below illustrates the host controller QH traversal state machine.

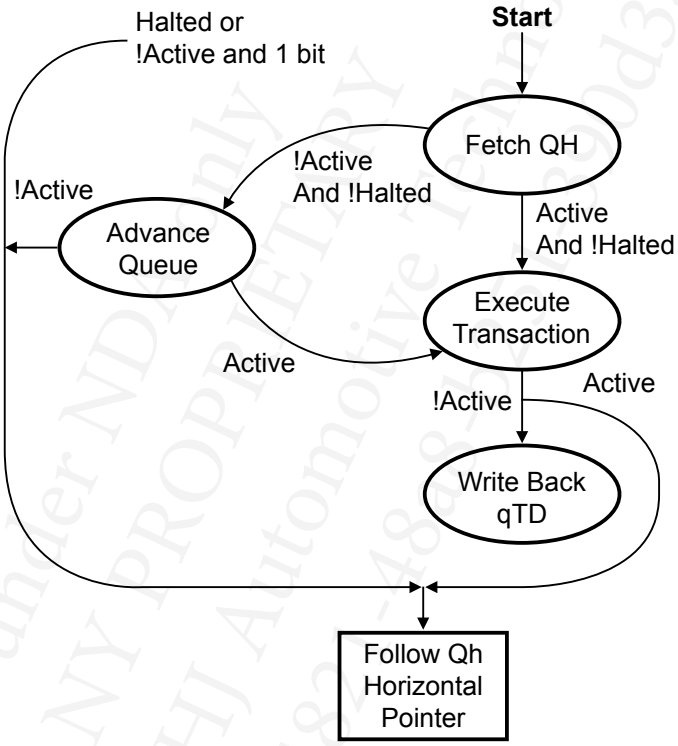


Figure 462. Host controller QH traversal state machine

This traversal state machine applies to all QHs, regardless of transfer type or whether split transactions are required. The following sections describe each state, and each state description describes the entry criteria. The Execute Transaction state (see Execute transaction) describes the basic requirements for all endpoints. Split transactions for asynchronous transfers and Split transaction interrupt describe details of the required extensions to the Execute Transaction state for endpoints requiring split transactions.

NOTE

Prior to placing a QH into either the periodic or asynchronous list, you must ensure the QH is properly initialized. At a minimum, the QH must be initialized to the following (see section QH for layout of a QH):

Valid static endpoint state:

- For the very first use of a QH, you may zero-out the QH transfer overlay, then set the next qTD pointer field value to reference a valid qTD.

55.10.3.1 Fetch QH

A QH can be referenced from the physical address stored in ASYNCLISTADDR. Additionally, it may be referenced from the Next LinkPointer field of an iTD, siTD, FSTN, or another QH. If the referencing link pointer has the Typ field set to indicate a QH, it is assumed to reference a QH structure as defined in Table 421.

When in this state, the host controller performs operations to implement empty schedule detection and Nak Counter reloads. After the QH has been fetched, the host controller checks the following conditions for empty schedule detection:

- The QH is not an interrupt QH (that is, S-mask is 0).
- The H-bit is 1.
- USBSTS[RCL] is 0.

When the above criteria are met, the host controller stops traversing the asynchronous list. When the criteria are not met, the host controller continues schedule traversal. If the QH is not an interrupt and the H-bit is one and USBSTS[RCL] is 1, then the host controller writes USBSTS[RCL] to 0 before completing this state. The operations for reloading of the Nak Counter are described in detail in the Operational Model for Nak Counter. For details, see the standard documentation of *Enhanced Host Controller Interface Specification for Universal Serial Bus (EHCI)* available at intel.in.

This state is complete when the QH has been read on-chip.

55.10.3.2 Advance queue

To advance the queue, the host controller must find the next qTD, adjust the pointers, perform the overlay, and write the results back to the QH.

This state is entered from the FetchQHD state if the overlay Active and Halt bits are set to 0. Upon entry into this state, the host controller determines which next pointer to use to fetch a qTD, fetches a qTD, and then determines whether or not to perform an overlay.

NOTE

If the I-bit is 1 and the Active field is 1, the host controller immediately skips processing of this QH, exits this state, and uses the horizontal pointer to the next schedule data structure. If the Bytes to Transfer field is not 0 and the T-bit in the Alternate Next qTD Pointer is set to 0, then the host controller uses the Alternate Next qTD Pointer. Otherwise, the host controller uses the NextqTD Pointer. If NextqTD Pointer's T-bit is set to 1, then the host controller exits this state and uses the horizontal pointer to the next schedule data structure.

Using the selected pointer, the host controller fetches the referenced qTD. If the fetched qTD has its Active field set to 1, the host controller moves the pointer value used to reach the qTD (next or alternate next) to the Current qTD Pointer field, then performs the overlay. If the fetched qTD has its Active field set to 0, the host controller aborts the queue advance and follows the QH's horizontal pointer to the next schedule data structure.

The host controller performs the overlay based on the following rules:

- The value of the Data Toggle (DT) field in the overlay area depends on the value of the Data Toggle Control (DTC) field (see Table 423).
- If the EPS field indicates the endpoint is a HS endpoint, the Ping state field is preserved by the host controller. The value of this field is not changed as a result of the overlay.
- C-prog-mask field is set to 0 (field from incoming qTD is ignored, as is the current contents of the overlay area).
- Frame Tag field is set to 0 (field from incoming qTD is ignored, as is the current contents of the overlay area).
- NakCnt field in the overlay area is loaded from the RL field in the QH's Static Endpoint State.
- All other areas of the overlay are set by the incoming qTD.

The host controller exits this state when it has committed the write to the QH.

55.10.3.3 Execute transaction

The host controller enters this state from the Fetch QH Head state only if the Active field in Status field of the QH is set to 1.

Upon entry to this state, the host controller executes a few pre-operations, then checks pre-condition criteria before committing to executing a transaction for the QH.

The pre-operations performed and the pre-condition criteria depend on whether the QH is an interrupt endpoint. The host controller can determine that a QH is an interrupt QH when the QH's S-mask field contains a non-zero value. It is your responsibility to ensure the S-mask field is appropriately initialized based on the transfer type. There are other criteria that must be met if the EPS field indicates that the endpoint is a LS or FS endpoint; see Split transactions for asynchronous transfers and Split transaction interrupt.

55.10.3.3.1 Interrupt transfer pre-condition criteria

If the QH is for an interrupt endpoint (for example, non-zero S-mask field), then the FRINDEX[2:0] field must identify a bit in the S-mask field that has 1 in it.

For example, an S-mask value of 00100000b would evaluate to true only when FRINDEX[2:0] is equal to 101b. If this condition is met, then the host controller considers this QH for a transaction.

55.10.3.3.2 Asynchronous transfer pre-operations and pre-condition criteria

If the QH is not for an interrupt endpoint (for example, 0 in the S-mask field), then the host controller performs one pre-operation and then evaluates one pre-condition criteria.

The pre-operation is:

- Check the Nak Counter reload state. It may be necessary for the host controller to reload the Nak Counter field. The reload is performed at this time.

The pre-condition evaluated is:

- Regardless of whether or not the NakCnt field has been reloaded, the host controller checks the value of the NakCnt field in the QH. If NakCnt is non-zero, or if the Reload Nak Counter field is 0, then the host controller considers this QH for a transaction.

55.10.3.3.3 Transfer type independent pre-operations

Regardless of the transfer type, the host controller always performs at least one pre-operation and evaluates one pre-condition. The pre-operation is:

- A host controller internal transaction (down) counter qHTransactionCounter is loaded from the QH's Mult field. A host controller implementation is allowed to ignore this for QHs on the asynchronous list. It is mandatory for interrupt QHs. You must ensure that the Mult field is set appropriately for the transfer type.

The pre-conditions evaluated are:

- The host controller determines whether there is enough time in the microframe to complete this transaction. If there is not enough time to complete the transaction, the host controller exits this state.
- If the value of qHTransactionCounter for an interrupt endpoint is 0, then the host controller exits this state.

When the pre-operations are complete and pre-conditions are met, the host controller sets USBSTS[RCL] to 1 and then begins executing one or more transactions using the endpoint information in the QH. The host controller iterates qHTransactionCounter times in this state executing transactions. After each transaction is executed, qHTransactionCounter is decremented by 1. The host controller exits this state when one of the following events occurs:

- The qHTransactionCounter decrements to 0.
- The endpoint responds to the transaction with any handshake other than an ACK,[4].
- The transaction experiences a transaction error.
- The Active field in the QH goes to 0.

- There is not enough time remaining in the microframe to execute the next transaction.

NOTE

For a high-bandwidth interrupt OUT endpoint, the host controller may optionally retry the transaction immediately if it fails.

The results of each transaction are recorded in the on-chip overlay area. If data was successfully moved during the transaction, the transfer state in the overlay area is advanced. To advance the QH's transfer state, the Total Bytes to Transfer field is decremented by the number of bytes moved in the transaction, the Data Toggle field (DT) is toggled, the current page offset is advanced to the next appropriate value (for example, advanced by the number of bytes successfully moved), and the C_Page field is updated to the appropriate value (if necessary). See Buffer pointer list use for data streaming with qTDs.

NOTE

The Total Bytes to Transfer field may be 0 when all the other criteria for executing a transaction are met. When this occurs, the host controller executes a zero-length transaction to the endpoint. If the PID_Code field indicates an IN transaction and the device delivers data, the host controller detects a packet babble condition, sets the babble and halted fields in the Status field, sets the Active field to 0, writes the results back to the source qTD, and then exits this state.

In the event an IN token receives a data PID mismatch response, the host controller must ignore the received data (for example, not advance the transfer state for the bytes received). Additionally, if the endpoint is an interrupt IN, then the host controller must record that the transaction occurred (for example, decrement qHTransactionCounter). It is recommended (but not required) that the host controller continue executing transactions for this endpoint if the resultant value of qHTransactionCounter is greater than 1.

If the response to the IN bus transaction is a Nak (or Nyet) and RL is non-zero, NakCnt is decremented by 1. If RL is 0, then no write-back by the host controller is required (for a transaction receiving a Nak or Nyet response and the value of CErr did not change). You must set the RL field to 0 if the QH is an interrupt endpoint. Host controller hardware is not required to enforce this rule or operation.

After the transaction has finished and the host controller has completed the post processing of the results (advancing the transfer state and possibly NakCnt), the host controller writes back the results of the transaction to the QH's overlay area in main memory.

The number of bytes moved during an IN transaction depends on how much data the device endpoint delivers. The maximum number of bytes a device can send is MaximumPacket Size. The number of bytes moved during an OUT transaction is either Maximum Packet Length bytes or Total Bytes to Transfer, whichever is less.

If there was a transaction error during the transaction, the transfer state (as defined above) is not advanced by the host controller. The CErr field is decremented by 1 and the status field is updated to reflect the type of error observed.

The following events will cause the host controller to clear the Active field to 0 in the QH's overlay status field. When the Active field transitions from 1 to 0, the transfer in the overlay is considered complete. The reason for the transfer completion (clearing the Active field to 0) determines the next state.

- The CErr field decrements to 0. When this occurs, the Halted field is set to 1 and Active is set to 0. This results in the hardware not advancing the queue and the pipe halts. You must intercede to recover.
- The device responds to the transaction with a STALL PID. When this occurs, the Halted field is set to 1 and the Active field is set to 0. This results in the hardware not advancing the queue and the pipe halts. You must intercede to recover.
- The Total Bytes to Transfer field is 0 after the transaction completes.
  - For a zero length transaction, it was 0 before the transaction was started. When this condition occurs, the Active field is set to 0.
- The PID code is an IN, and the number of bytes moved during the transaction is less than the Maximum Packet Length. When this occurs, the Active field is set to 0 and a short-packet condition exists. The short-packet condition is detected during the Advance Queue state. See Split transactions for additional rules for managing LS and FS transactions.

With the exception of a NAK response (when the value of the RL field is 0), the host controller always writes the results of the transaction back to the overlay area in main memory. This includes when the transfer completes. For a HS endpoint, the QH

information written back includes the following fields at a minimum: The PID Code field indicates an IN and the device sends more than the expected number of bytes (for example Maximum Packet Length or Total Bytes to Transfer bytes, whichever is less) (for example, a packet babble). This results in the host controller setting the Halted field to 1.

- NakCnt, DT, Total Bytes to Transfer, C_Page, Status, CERR, and Current Offset

For a LS or FS device, the QH information written back also includes these fields:

- C-prog-mask, FrameTag and S-bytes

The duration of this state depends on the time it takes to complete the transaction(s) and the status write to the overlay is committed.

55.10.3.3.4 Halting a QH

A halted endpoint is defined only for the transfer types that are managed through QHs (control, bulk and interrupt). The following events indicate that the endpoint has reached a condition where no more activity can occur without intervention from the driver:

- An endpoint may return a STALL handshake during a transaction,
- A transaction had three consecutive error conditions, or
- A Packet Babble error occurs on the endpoint.

When any of these events occur (for a QH), the host controller halts the QH and sets the USBSTS[UEI] status field to 1. To halt the QH, the Active field is set to 0 and the Halted field is set to 1. There may be other error status fields that are set when a queue is halted. The host controller always writes back the overlay area to the source qTD when the transfer is complete, regardless of the reason (normal completion, short packet or halt). The host controller does not advance the transfer state on a transaction that results in a Halt condition (for example, no updates necessary for Total Bytes to Transfer, C_Page, Current Offset, and DT). The host controller must update CErr as appropriate. When a QH is halted, USBSTS[UEI] is set to 1. If the USBINTR[UEE] is set to 1, a hardware interrupt is generated at the next interrupt threshold.

55.10.3.3.5 Asynchronous Schedule Park mode

Asynchronous Schedule Park mode is a special execution mode that can be enabled by system software, where the host controller is permitted to execute more than one bus transaction from a HS QH in the Asynchronous schedule before continuing horizontal traversal of the Asynchronous schedule.

This feature has no effect on QHs or other data structures in the Periodic schedule. This feature is similar in intent to the Mult feature that is used in the Periodic schedule. Where the Mult feature is a characteristic that is tunable for each endpoint, park-mode is a policy that is applied to all HS QHs in the Asynchronous schedule. It is essentially the specification of an iterator for consecutive bus transactions to the same endpoint. All of the rules for managing bus transactions and the results of those as defined in Execute transaction apply. This feature merely specifies how many consecutive times the host controller is permitted to execute from the same QH before moving to the next QH in the Asynchronous List. This feature allows the host controller to attain better bus usage for those devices that are capable of moving data at maximum rate, and at the same time provide a fair service to all endpoints.

A host controller exports its capability to support this feature to system software by setting the Asynchronous Schedule Park Capability field in the USB_n_HCCPARAMs register to 1. This information keys system software that the Asynchronous Schedule Park Mode Enable and Asynchronous Schedule Park Mode Count fields in the USB_n_USBCMD register are modifiable. System software enables the feature by writing a 1 to the Asynchronous Schedule Park Mode Enable field.

When park-mode is not enabled (for example, when the Asynchronous Schedule Park Mode Enable field in the USB_n_USBCMD register is 0), the host controller must not execute more than one bus transaction per HS QH, per traversal of the asynchronous schedule. When park-mode is enabled, the host controller must not apply the feature to a QH whose EPS field indicates an LS or FS device (for example, only one bus transaction is allowed from each LS or FS QH per traversal of the asynchronous schedule). Park-mode may only be applied to QHs in the asynchronous schedule whose EPS field indicates that it is an HS device.

The host controller must apply park mode to QHs whose EPS field indicates a HS endpoint. The maximum number of consecutive bus transactions a host controller may execute on a HS QH is determined by the value in the Asynchronous Schedule Park Mode Count field in the USB_n_USBCMD register. You must not set the Asynchronous Schedule Park Mode Enable field to 1 and also set the Asynchronous Schedule Park Mode Count field to 0; the resulting behavior is not defined. A behavioral example describes

the operational requirements for the host controller implementing park-mode. How the host controller manages the bus transaction as defined in Execute transaction is not affected by this feature. It only affects how many consecutive bus transactions for the current QH can be executed. All boundary conditions, error detection, and reporting apply as usual. This feature is similar in concept to the use of the Mult field for high-bandwidth interrupts for QHs in the periodic schedule.

The host controller effectively loads an internal down-counter PM-Count from Asynchronous Schedule Park Mode Count when the Asynchronous Schedule Park Mode Enable field is 1, and a HS QH is first fetched and meets all the criteria for executing a bus transaction. After the bus transaction, PM-Count is decremented. The host controller may continue to execute bus transactions from the current QH until PM-Count goes to 0, an error is detected, the buffer for the current transfer is exhausted, or the endpoint responds with a flow-control or STALL handshake.

The following table summarizes the responses that affect whether the host controller continues with another bus transaction for the current QH.

Table 445. Actions for park mode, based on endpoint response and residual transfer state

PID	Endpoint response	Transfer state after transaction		Action
		PM-Count	Bytes to transfer	
IN	DATA[0,1] with maximum packet-sized data	Not zero	Not zero	Allowed to perform another bus transaction[1,2]
		Not zero	Zero	Retire qTD and move to next QH
		Zero	Don't care	Move to next QH
	DATA[0,1] with short packet	Don't care	Don't care	Retire qTD and move to next QH
	NAK	Don't care	Don't care	Move to next QH
	STALL, XactErr	Don't care	Don't care	Move to next QH
OUT	ACK	Not zero	Not zero	Allowed to perform another bus transaction[2]
		Not zero	Zero	Retire qTD and move to next QH
		Zero	Don't care	Move to next QH
	NYET, NAK	Don't care	Don't care	Move to next QH
	STALL, XactErr	Don't care	Don't care	Move to next QH
PING	ACK	Not zero	Not zero	Allowed to perform another bus transaction [2]
	NAK	Don't care	Don't care	Move to next QH
	STALL, XactErr	Don't care	Don't care	Move to next QH

1. The host controller may continue to execute bus transactions from the current HS QH (if PM-Count is not 0), if a PID mismatch is detected (for example, expected DATA1 and received DATA0, or vice-versa).
2. This specification does not require that the host controller execute another bus transaction when PM-Count is non-zero. Implementations are encouraged to make appropriate complexity and performance trade-offs.

55.10.3.4 Write back qTD

This state is entered from the Execute Transaction state when the Active bit is set to 0.

The source data for the write-back is the transfer results area of the QH overlay area (see Table 445).

The host controller uses the Current qTD Pointer field as the target address for the qTD.

The QH transfer result area is written back to the transfer result area of the target qTD. This state is also referred to as "qTD Retirement". The fields that must be written back to the source qTD include Total Bytes to Transfer, Cerr, and Status.

The duration of this state depends on when the qTD write-back is committed.

55.10.3.5 Follow QH horizontal pointer

The host controller must use the horizontal pointer in the QH to the next schedule data structure when any of the following conditions exist:

- If the Active bit is 1 on exit from the Execute Transaction state
- When the host controller exits the Write Back qTD state
- If the Advance Queue state fails to advance the queue because the target qTD is not active
- If the Halted bit is 1 on exit from the fetch QH state

There is no functional requirement for the host controller to wait until the current transaction is complete before using the horizontal pointer to read the next linked data structure. However, it must wait until the current transaction is complete before executing the next data structure.

55.10.3.6 Buffer pointer list use for data streaming with qTDs

A qTD has an array of buffer pointers, which is used to reference the data buffer for a transfer. This specification requires that the buffer associated with the transfer be virtually contiguous. This means that, if it spans more than one physical page, the buffer must obey the following rules (the figure below illustrates an example):

- The first portion of the buffer must begin at some offset in a page and extend through the end of the page.
- The remaining buffer cannot be allocated in small chunks scattered around memory. For each 4 KB chunk beyond the first page, each buffer portion matches to a full 4 KB page. The final portion, which may only be large enough to occupy a portion of a page, must start at the top of the page and be contiguous within that page.

The buffer pointer list in the qTD is long enough to support a maximum transfer size of 20 KB. This case occurs when all five buffer pointers are used and the first offset is zero. A qTD manages a 16 KB buffer with any starting buffer alignment.

The host controller uses the C_Page field as an index value to determine which buffer pointer in the list must be used to start the current transaction. The host controller uses a different buffer pointer for each physical page of the buffer. This is always true, even if the buffer is physically contiguous.

The host controller must detect when the current transaction spans a page boundary and automatically move to the next available buffer pointer in the page pointer list. The next available pointer is reached by incrementing C_Page and pulling the next page pointer from the list. You must ensure there are sufficient buffer pointers to move the amount of data specified in the Bytes to Transfer field.

The following figure illustrates a nominal example of how system software would initialize the buffer pointers list and the C_Page field for a transfer size of 16383 bytes. C_Page is set to 0. The upper 20 bits of page 0 references the start of the physical page. Current Offset (the lower 12 bits of QH Dword 7) holds the offset in the page; for example, 2049 (that is, 4096-2047). The remaining page pointers are set to reference the beginning of each subsequent 4 KB page.

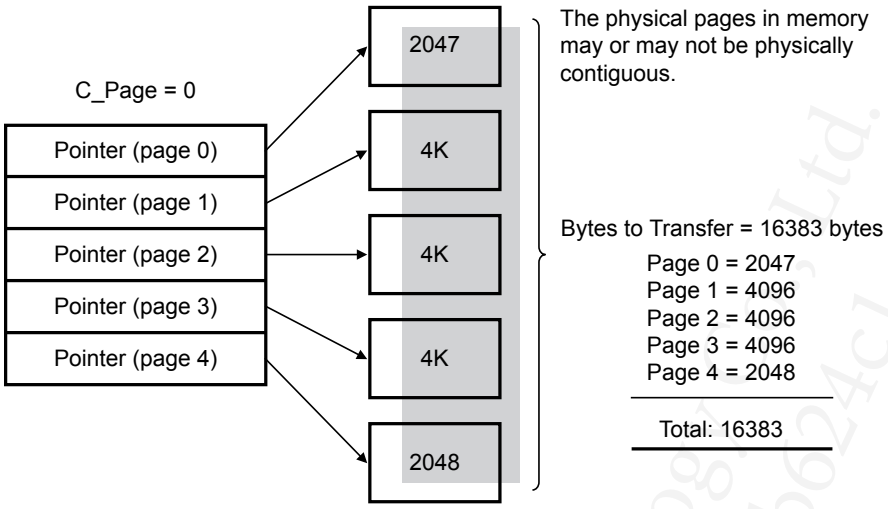


Figure 463. Example mapping of qTD buffer pointers to buffer pages

For the first transaction on the qTD (assuming a 512-byte transaction), the host controller uses the first buffer pointer (page 0 because C_Page is set to 0) and concatenates the Current Offset field. The 512 bytes are moved during the transaction, and current offset and total bytes to transfer are adjusted by 512 and written back to the QH working area.

During the 4th transaction, the host controller needs 511 bytes in page 0 and one byte in page 1. The host controller increments C_Page (to 1) and uses the page 1 pointer to move the final byte of the transaction. After the 4th transaction, the active page pointer is the page 1 pointer and Current Offset has rolled to 1, and both are written back to the overlay area. The transactions continue for the rest of the buffer, with the host controller automatically moving to the next page pointer (that is, C_Page) when necessary. The three conditions for how the host controller manages C_Page are:

- The current transaction does not span a page boundary. The value of C_Page is not adjusted by the host controller.
- The current transaction spans a page boundary. The host controller must detect the page cross condition and advance to the next buffer when streaming data to and from the USB.
- The current transaction completes on a page boundary (that is, the last byte moved for the current transaction is the last byte in the page for the current page pointer). The host controller must increment C_Page before writing back status for the transaction.

NOTE

The only valid adjustment the host controller may make to C_Page is to increment by 1.

55.10.3.7 Adding interrupt QHs to the periodic schedule

The link path(s) from the periodic frame list to a QH establishes in which frames a transaction can be executed for the QH. QHs are linked into the periodic schedule so they are polled at the appropriate rate.

System software sets a bit in a QH's S-Mask to indicate which microframe within a 1 ms period a transaction must be executed for the QH. You must ensure that all QHs in the periodic schedule have S-Mask set to a non-zero value. An S-mask with zero value in the context of the periodic schedule yields undefined results.

If the desired poll rate is greater than one frame, system software can use a combination of QH linking and S-Mask values to spread interrupts of equal poll rates through the schedule so that the periodic bandwidth is allocated and managed in the most efficient manner possible. Some examples are illustrated in the following table.

Table 446. Example periodic reference patterns for interrupt transfers with 2 ms poll rate

Frame # reference sequence	Description
0, 2, 4, 6, 8, and so on S-Mask = 01h	A QH for the bInterval of 2 ms (16 microframes) is linked into the periodic schedule so that it is reachable from the periodic frame list locations indicated in the previous column. In addition, the S-Mask field in the QH is set to 01h, indicating that the transaction for the endpoint must be executed on the bus during microframe 0 of the frame.
0, 2, 4, 6, 8, and so on S-Mask = 02h	Another example of a QH with a bInterval of 2 ms is linked into the periodic frame list at exactly the same interval as the previous example. However, the S-Mask is set to 02h indicating that the transaction for the endpoint must be executed on the bus during microframe 1 of the frame.

55.10.3.8 Managing transfer complete interrupts from queue heads

The host controller sets an interrupt to be signaled at the next interrupt threshold when the completed transfer (qTD) has an IOC field that has a value of 1, or whenever a transfer (qTD) completes with a short packet.

If system software needs multiple qTDs to complete a client request (such as a control transfer), the intermediate qTDs do not require interrupts. System software may only need a single interrupt to notify it that the complete buffer has been transferred. System software may set IOCs to occur more frequently. A motivation for this may be that it wants early notification so that interface data structures can be reused in a timely manner.

55.10.4 Split transactions

USB 2.0 defines extensions to the bus protocol for managing USB 1.x data streams through USB 2.0 Hubs.

This section describes how the host controller uses the interface data structures to manage data streams with FS and LS devices, connected below USB 2.0 hub, using the split transaction protocol.

See the USB 2.0 Specification for the complete definition of the split transaction protocol. FS and LS devices are enumerated identically as HS devices, but the transactions to the FS and LS endpoints use the split-transaction protocol on the HS bus. The split transaction protocol is an encapsulation of (or wrapper around) the FS or LS transaction. The HS wrapper portion of the protocol is addressed to the USB 2.0 Hub and Transaction Translator, below which the FS or LS device is attached.

The EHCI interface uses dedicated data structures for managing FS isochronous data streams (see siTD). Control, bulk, and interrupt are managed using the queuing data structures (see QH). The interface data structures need to be programmed with the device address and the Transaction Translator number of the USB 2.0 Hub operating as the LS or FS host controller for this link. The following sections describe the details of how the host controller must process and manage the split transaction protocol.

55.10.4.1 Split transactions for asynchronous transfers

A QH in the asynchronous schedule with an EPS field indicating a FS or LS device indicates to the host controller that it must use split transactions to stream data for this QH.

All FS bulk, full- and LS control are managed through QHs in the asynchronous schedule.

You must initialize the QH with the appropriate device address and port number for the transaction translator that is serving as the FS/LS host controller for the links connecting the endpoint. You must also initialize the split transaction state field (SplitXState) to Do-Start-Split. Finally, if the endpoint is a control endpoint, then system software must set the Control Transfer Type (C) field in the QH to 1. If this is not a control transfer type endpoint, you must initialize the C field to be 0. This information is used by the host controller to properly set the Endpoint Type (ET) field in the split transaction bus token. When the C field is 0, the split transaction token's ET field is set to indicate a bulk endpoint. When the C field is 1, the split transaction token's ET field is set to indicate a control endpoint. See the USB Specification Revision 2.0 chapter for details.

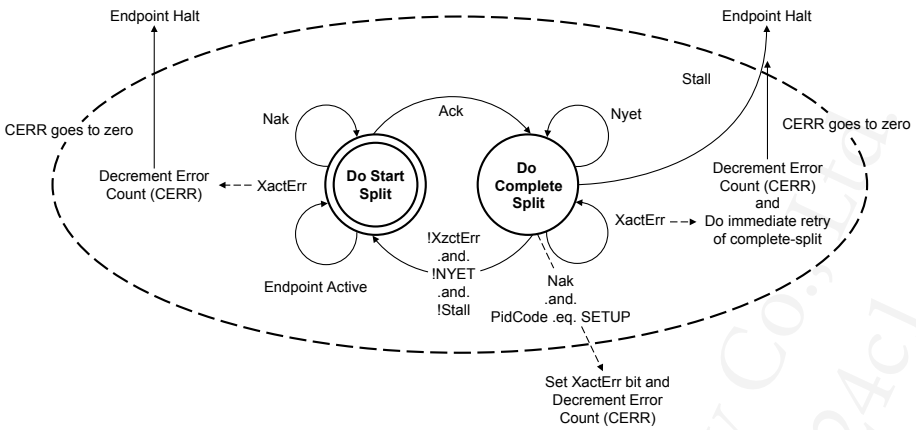


Figure 464. Host controller asynchronous schedule split-transaction state machine

55.10.4.1.1 Asynchronous—do start split

This is the state in which you must initialize a FS or LS asynchronous QH. This state is entered from the Do Complete Split state only after a complete-split transaction receives a valid response from the transaction translator that is not a Nyet handshake.

For QHs in this state, the host controller executes a start-split transaction to the appropriate transaction translator. If the bus transaction completes without an error and PidCode indicates an IN or OUT transaction, then the host controller reloads the error counter (CErr). If it is a successful bus transaction and the PidCode indicates a SETUP, the host controller does not reload the error counter. If the transaction translator responds with a Nak, the QH remains in this state, and the host controller proceeds to the next QH in the asynchronous schedule.

If the host controller times out the transaction (no response, or bad response) the host controller decrements Cerr and proceeds to the next QH in the asynchronous schedule.

55.10.4.1.2 Asynchronous—do complete split

This state is entered from the Do Start Split state only after a start-split transaction receives an Ack handshake from the transaction translator.

For QHs in this state, the host controller executes a complete-split transaction to the appropriate transaction translator. If the transaction translator responds with a Nyet handshake, the QH remains in this state, the error counter is reset and the host controller proceeds to the next QH in the asynchronous schedule. When a Nyet handshake is received for a bus transaction where the QH's PidCode indicates an IN or OUT, the host controller reloads the error counter (CErr). When a Nyet handshake is received for a complete-split bus transaction where the QH's PidCode indicates a SETUP, the host controller must not adjust the value of CErr.

Independent of PIDCode, the following responses have these effects:

- Transaction Error (XactErr). Timeout or data CRC failure, and so on. The error counter (Cerr) is decremented by 1 and the complete split transaction is immediately retried (if possible). If there is not enough time in the microframe to execute the retry, the host controller MUST ensure that the next time the host controller begins executing from the asynchronous schedule, it must begin executing from this QH. If another start-split (for some other endpoint) is sent to the transaction translator before the complete-split is really completed, the transaction translator could dump the results (which were never delivered to the host). This is why the core specification states the retries must be immediate. A method to accomplish this behavior is to not advance the asynchronous schedule. When the host controller returns to the asynchronous schedule in the next microframe, the first transaction from the schedule is the retry for this endpoint.

If Cerr went to 0, the host controller must halt the queue.

- NAK. The target endpoint generates a NAK response for the FS or LS transaction. The state of the transfer is not advanced and the state is exited. If the PidCode is a SETUP, then the Nak response is a protocol error. The XactErr status field is set to 1 and the CErr field is decremented.

- STALL. The target endpoint responded with a STALL handshake. The host controller sets the Halt field in the status byte and retires the qTD, but does not attempt to advance the queue.

If the PidCode indicates an IN, then any of following responses are expected:

- DATA0/1. On reception of data, the host controller ensures the PID matches the expected Data Toggle and checks CRC. If the packet is good, the host controller advances the state of the transfer; for example, it moves the data pointer by the number of bytes received, decrements the BytesToTransfer field by the number of bytes received, and toggles the DT field. The host controller then exits this state. The response and advancement of transfer may trigger other processing events, such as retirement of the qTD and advancement of the queue.

If the data sequence PID does not match the expected result, the data is ignored, the transfer state is not advanced, and this state is exited. If the PidCode indicates an OUT/SETUP, then any of following responses are expected:

- ACK. The target endpoint accepted the data, so the host controller must advance the state of the transfer. The Current Offset field is incremented by Maximum Packet Length or Bytes to Transfer, whichever is less. The field Bytes To Transfer is decremented by the same amount and the Data Toggle field (DT) is toggled. The host controller then exits this state.
- Advancing the transfer state may cause other processing events, such as retirement of the qTD and advancement of the queue (see Managing control/bulk/interrupt transfers through QHs).

55.10.4.2 Split transaction interrupt

Split-transaction interrupt-IN/OUT endpoints are managed through the same data structures used for HS interrupt endpoints; they coexist in the periodic schedule.

QHs/qTDs offer the set of features required for reliable data delivery, which is characteristic to interrupt transfer types. The split-transaction protocol is managed completely within this defined functional transfer framework. For example, for a HS endpoint, the host controller visits a QH, executes a HS transaction (if the criteria are met) and advances the transfer state (or not) depending on the results of the entire transaction. For LS and FS endpoints, the details of the execution phase are different (it takes more than one bus transaction to complete), but the remainder of the operational framework is intact. This means that the transfer advancement, and so on, occurs as defined in Managing control/bulk/interrupt transfers through QHs, but only occurs upon the completion of a split transaction.

55.10.4.2.1 Split transaction scheduling mechanisms for interrupt

FS and LS interrupt QHs have an EPS field indicating FS or LS and have a non-zero S-mask field.

The host controller can detect this combination of parameters and assume the endpoint is a periodic endpoint. LS and FS interrupt QHs require the use of the split transaction protocol. The host controller sets the Endpoint Type (ET) field in the split token to indicate the transaction is an interrupt. These transactions are managed through a transaction translator's periodic pipeline. You must not set these fields to indicate the QH is an interrupt unless the QH is used in the periodic schedule.

System software manages the per-transaction translator periodic pipeline by budgeting and scheduling exactly during which microframes the start-splits and complete-splits for each endpoint occur. The characteristics of the transaction translator are such that the HS transaction protocol must execute during explicit microframes, or else the data or response information in the pipeline is lost.

The following figure illustrates the general scheduling boundary conditions that are supported by the EHCI periodic schedule and QH data structure. The S and C_X labels indicate microframes in which you can schedule start-splits and complete splits, respectively.

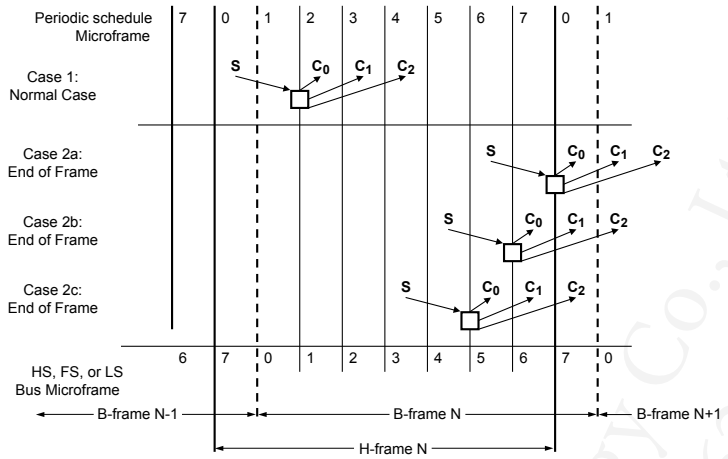


Figure 465. Split transaction, interrupt scheduling boundary conditions

The scheduling cases are:

- Case 1: The normal scheduling case is where the entire split transaction is completely bounded by a frame (H-Frame, in this case).
- Case 2a through Case 2c: The USB 2.0 Hub pipeline rules states clearly, determining when and how many complete-splits must be scheduled to account for earliest to latest execution on the FS/LS link. The complete-splits may span the H-Frame boundary when the start-split is in microframe four or later. When this occurs, the H-Frame to B-Frame alignment requires that the QH be reachable from consecutive periodic frame list locations. System software cannot build an efficient schedule that satisfies this requirement unless it uses FSTNs.

The figure below illustrates the general layout of the periodic schedule.

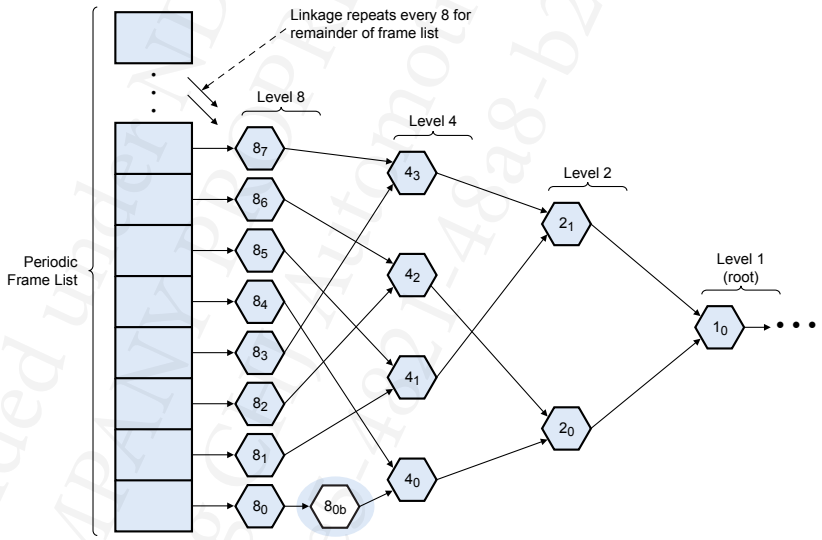


Figure 466. General structure of EHCI periodic schedule using interrupt spreading

The periodic frame list is effectively the leaf level of a binary tree, which is always traversed leaf to root. Each level in the tree corresponds to a 2^N poll rate. You can efficiently manage periodic bandwidth on the USB by spreading interrupt QHs that have the same poll rate requirement across all the available paths from the frame list. For example, system software can schedule eight poll rate and eight QHs and account for them once in the HS bus bandwidth allocation.

When an endpoint is allocated an execution footprint that spans a frame boundary, the QH for the endpoint must be reachable from consecutive locations in the frame list. An example would be if 8_{0b} were such an endpoint. Without additional support on the

interface, for 8_{0b} to be reachable at the correct time, you would have to link 8_1 to 8_{0b}. It would then have to move 4_1 and everything linked after into the same path as 4_0. This upsets the integrity of the binary tree and disallows the use of the spreading technique.

FSTN data structures are used to preserve the integrity of the binary-tree structure and enable the use of the spreading technique. Host controller operational model for FSTNs defines the hardware and software operational model requirements for using FSTNs.

The following QH fields are initialized by system software to instruct the host controller when to execute portions of the split-transaction protocol:

- SplitXState. This is a single bit that resides in the Status field of a QH (see Table 419). This bit is used to track the current state of the split transaction.
- Frame S-mask. The system software sets a bit within this field corresponding to the microframe (within an H-Frame) for which the host controller should execute a start-split transaction. This is always qualified by the value of the SplitXState bit in the Status field of the QH. For example, referring to case one in Figure 465, the S-mask would have a value of 0000_0001b; this indicates that if the QH is traversed by the host controller, and the SplitXState indicates Do_Start, and the current microframe indicated by FRINDEX[2:0] is 0, then execute a start-split transaction.
- Frame C-mask. The system software sets one or more bits within this field corresponding to the microframes (within an H-Frame) for which the host controller should execute complete-split transactions. The interpretation of this field is always qualified by the value of the SplitXState bit in the Status field of the QH. For example, referring to case one in Figure 465, the C-mask would have a value of 0001_1100b; this indicates that if the QH is traversed by the host controller, and the SplitXState indicates Do_Complete, and the current microframe indicated by FRINDEX[2:0] is 2, 3, or 4, then execute a complete-split transaction. You are responsible for ensuring that the translation between H-Frames and B-Frames is correctly performed when setting bits in S-mask and C-mask.

55.10.4.2.2 Host controller operational model for FSTNs

The FSTN data structure is used to manage LS/FS interrupt QHs that need to be reached from consecutive frame list locations (that is boundary cases 2a through 2c).

An FSTN is essentially a back pointer, similar in intent to the back pointer field in the siTD data structure (see siTD back link pointer).

This feature provides you with a simple primitive to save a schedule position, redirect the host controller to traverse the necessary QHs in the previous frame, then restore the original schedule position and complete normal traversal.

The four components to the use of FSTNs are:

- FSTN data structure.
- A Save Place indicator. This is always an FSTN with its Back Path Link Pointer.T-bit set to 0.
- A Restore indicator. This is always an FSTN with its Back Path Link Pointer.T-bit set to 1.
- Host controller FSTN traversal rules.

When the host controller encounters an FSTN during microframes 2 through 7 it simply follows the node's Normal Path Link Pointer to access the next schedule data structure.

NOTE

The FSTN's Normal Path Link Pointer.T-bit may set to 1, which the host controller must interpret as the end of periodic list mark.

When the host controller encounters a Save-Place FSTN in microframes 0 or 1, it saves the value of the Normal Path Link Pointer and sets an internal flag indicating that it is executing in Recovery Path mode. Recovery Path mode modifies the host controller's rules for how it traverses the schedule and limits which data structures are considered for execution of bus transactions. The host controller continues executing in Recovery Path mode until it encounters a Restore FSTN or it determines that it has reached the end of the microframe (see details in the list below).

The rules for schedule traversal and limited execution when in Recovery Path mode are:

- Always follow the Normal Path Link Pointer when it encounters an FSTN that is a Save-Place indicator. The host controller must not recursively follow Save-Place FSTNs. Therefore, when executing in Recovery Path mode, it must never follow an FSTN's Back Path Link Pointer.
- Do not process an siTD or iTD data structure. Simply follow its Next Link Pointer.
- Do not process a QH whose EPS field indicates a HS device. Simply follow its Horizontal Link Pointer.
- When a QH's EPS field indicates a FS/LS device, the host controller only considers it for execution if its SplitXState is DoComplete (note: this applies whether the PID Code indicates an IN or an OUT). See Execute transaction and Tracking split transaction progress for interrupt transfers for a complete list of additional conditions that must be met in general for the host controller to issue a bus transaction.
  - The host controller must not execute a Start-split transaction when executing in Recovery Path mode. See Periodic isochronous - do complete split for special handling when in Recovery Path mode.
- Stop traversing the recovery path when it encounters an FSTN that is a Restore indicator. The host controller unconditionally uses the saved value of the Save-Place FSTN's Normal Path Link Pointer when returning to the normal path traversal. The host controller must clear the context of executing a Recovery Path when it restores schedule traversal to the Save-Place FSTN's Normal Path Link Pointer.
- If the host controller determines that there is not enough time remaining in the microframe to complete processing of the periodic schedule, it abandons traversal of the recovery path and clears the context of executing a recovery path. The result is that at the start of the next consecutive microframe, the host controller starts traversal at the frame list.

An example traversal of a periodic schedule that includes FSTNs is illustrated in the following figure.

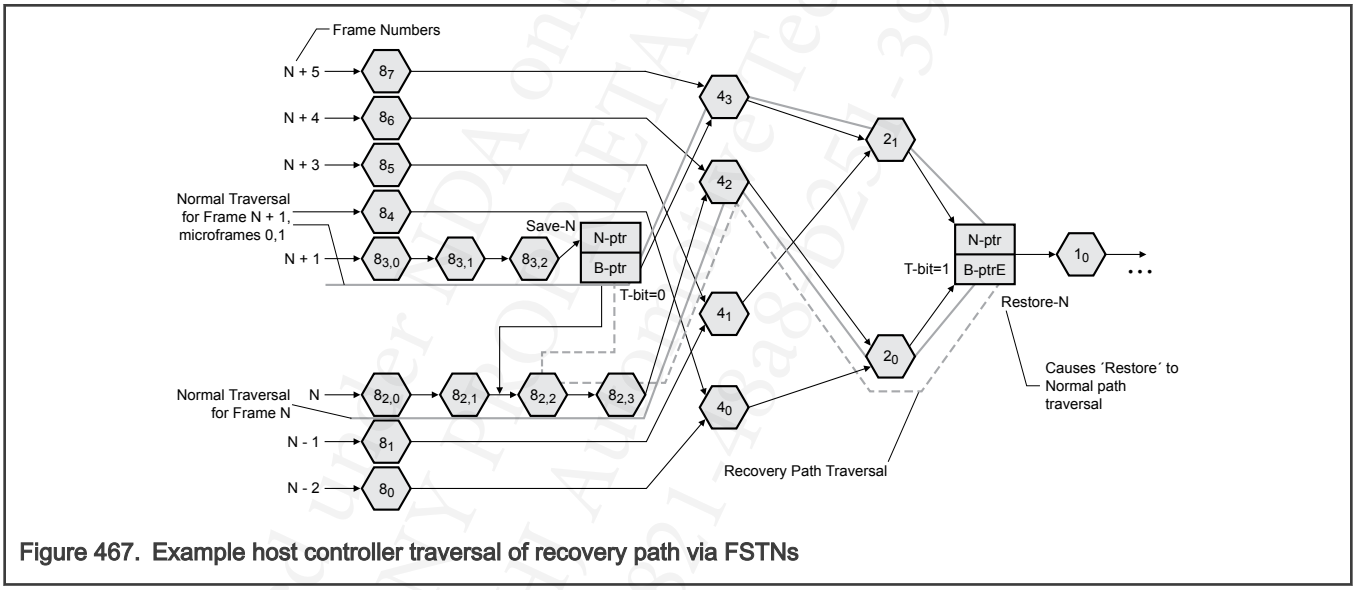


Figure 467. Example host controller traversal of recovery path via FSTNs

In frame N+1 (microframes 0 and 1), when the host controller encounters Save-Path FSTN (Save-N), it observes that the Save-N.Back Path Link Pointer.T-bit is 0 (definition of a Save-Path indicator). The host controller saves the value of Save-N.Normal Path Link Pointer and follows Save-N.Back Path Link Pointer. At the same time, it sets an internal flag indicating that it is now in Recovery Path mode (the recovery path is annotated in the figure above with a large dashed line). The host controller continues traversing data structures on the recovery path and executing only those bus transactions as noted above, on the recovery path until it reaches Restore FSTN (Restore-N). Restore-N.Back Path Link Pointer.T-bit is set to 1 (definition of a Restore indicator), so the host controller exits Recovery Path mode by clearing the internal Recovery Path mode flag and commences (restores) schedule traversal using the saved value of the Save-Place FSTN's Normal Path Link Pointer (for example Save-N.Normal Path Link Pointer). The nodes traversed during these microframes include: {$8_{3.0}$, $8_{3.1}$, $8_{3.2}$, Save-A, $8_{2.2}$, $8_{2.3}$, 4_2, 2_0, Restore-N, 4_3, 2_1, Restore-N, 1_0 ...}. The nodes on the recovery-path are in bold. In frame N (microframes 0-7), for this example, the host controller traverses all of the schedule data structures using the Normal Path Link Pointers in any FSTNs it encounters. This is because the host controller has not yet encountered a Save-Place FSTN so it is not executing in Recovery Path mode. When it encounters the Restore FSTN, (Restore-N), during microframes 0 and 1, it uses Restore-N.Normal Path Link Pointer

to traverse to the next data structure (that is normal schedule traversal). This is because the host controller must use a Restore FSTN's Normal Path Link Pointer when not executing in a Recovery Path mode. The nodes traversed during frame N include: {$8_{2.0}$, $8_{2.1}$, $8_{2.2}$, $8_{2.3}$, 4_2, 2_0, Restore-N, 1_0 ...}.

In frame N+1 (microframes 2-7), when the host controller encounters Save-Path FSTN Save-N, it unconditionally follows Save-N.Normal Path Link Pointer. The nodes traversed during these microframes include: {$8_{3.0}$, $8_{3.1}$, $8_{3.2}$, Save-A, 4_3, 2_1, Restore-N, 1_0 ...}.

55.10.4.2.3 Software operational model for FSTNs

You must create a consistent, coherent schedule for the host controller to traverse.

When using FSTNs, the system software must adhere to the following rules:

- Each Save-Place indicator requires a matching Restore indicator.
  - The Save-Place indicator is an FSTN with a valid Back Path Link Pointer and T-bit equal to 0.
    - Back Path Link Pointer.Typ field must be set to indicate the referenced data structure is a QH. The Restore indicator is an FSTN with its Back Path Link Pointer.T-bit set to 1.
  - A Restore FSTN may be matched to one or more Save-Place FSTNs. For example, if the schedule includes a poll-rate one level, then system software only needs to place a Restore FSTN at the beginning of this list in order to match all possible Save-Place FSTNs.
- If the schedule does not have elements linked at a poll-rate level of one, and one or more Save-Place FSTNs are used, then System software must ensure the Restore FSTN's Normal Path Link Pointer's T-bit is set to 1, as this is used to mark the end of the periodic list.
- When the schedule does have elements linked at a poll rate level of one, a Restore FSTN must be the first data structure on the poll rate one list. All traversal paths from the frame list converge on the poll-rate one list. System software must ensure that Recovery Path mode is exited before the host controller is allowed to traverse the poll rate level one list.
- A Save-Place FSTN's Back Path Link Pointer must reference a QH data structure. The referenced QH must be reachable from the previous frame list location. In other words, if the Save-Place FSTN is reachable from frame list offset N, then the FSTN's Back Path Link Pointer must reference a QH that is reachable from frame list offset N-1.

You should make the schedule as efficient as possible. What this means in this context is that you must have no more than one Save-Place FSTN reachable in any single frame. Note there are times when two (or more, depending on the implementation) could exist as FS/LS footprints change with bandwidth adjustments. This could occur, for example when a bandwidth re-balance causes system software to move the Save-Place FSTN from one poll rate level to another. During the transition, you must preserve the integrity of the previous schedule until the new schedule is in place.

55.10.4.2.4 Tracking split transaction progress for interrupt transfers

To correctly maintain the data stream, the host controller must be able to detect and report errors where data is lost.

For interrupt-IN transfers, data is lost when it makes it into the USB 2.0 hub, but the USB 2.0 host system is unable to obtain it from the USB 2.0 Hub and into the system before it expires from the transaction translator pipeline.

When a lost data condition is detected, the queue must be halted, thus signaling system software to recover from the error. A data-loss condition exists whenever a start-split is issued, accepted, and successfully executed by the USB 2.0 Hub, but the complete-splits generate unrecoverable errors on the HS link, or the complete-splits do not occur at the correct times. One reason complete-splits might not occur at the correct time would be due to host-induced system hold-offs that cause the host controller to miss bus transactions because it cannot obtain timely access to the schedule in system memory.

The same condition can occur for an interrupt-OUT, but the result is not an endpoint halt condition and instead affects only the progress of the transfer. The QH has the following fields to track the progress of each split transaction. These fields are used to record an incremental state about which (and when) portions have been executed.

- C-prog-mask. This is an eight-bit bit-vector where the host controller keeps track of which complete-splits have been executed. Due to the nature of the Transaction Translator periodic pipeline, the complete-splits need to be executed in order. The host controller needs to detect when the complete-splits have not been executed in order. This can only occur

due to system hold-offs where the host controller cannot access the memory-based schedule. C-prog-mask is a simple bit-vector in which the host controller sets one of the C-prog-mask bits for each complete-split executed. The bit position is determined by the microframe number in which the complete-split was executed. The host controller always checks C-prog-mask before executing a complete-split transaction. If the previous complete-splits have not been executed, then one (or more) have been skipped and data has potentially been lost.

- FrameTag. This field is used by the host controller during the complete-split portion of the split transaction to tag the QH with the frame number (H-Frame number) when the next complete split must be executed.
- S-bytes. This field can be used to store the number of data payload bytes sent during the start-split (if the transaction was an OUT). The S-bytes field must be used to accumulate the data payload bytes received during the complete-splits (for an IN).

55.10.4.2.5 Split transaction execution state machine for interrupt

In this section, all references to microframe are in the context of a microframe within an H-Frame.

As with asynchronous FS and LS endpoints, a split-transaction state machine is used to manage the split transaction sequence.

In addition to the fields defined in the QH for scheduling and tracking the split transaction, the host controller calculates one internal mechanism that is also used to manage the split transaction. The internal calculated mechanism is:

- cMicroFrameBit. This is a single-bit encoding of the current microframe number. It is an eight-bit value calculated by the host controller at the beginning of every microframe. It is calculated from the three least significant bits of USB Frame Index (FRINDEX) (that is, cMicroFrameBit = (1 shifted-left(FRINDEX[2:0])). The cMicroFrameBit has at most one bit asserted, which always corresponds to the current microframe number. For example, if the current microframe is 0, then cMicroFrameBit will equal 00000001b. The variable cMicroFrameBit is used to compare against the S-mask and C-mask fields to determine whether the QH is marked for a start- or complete-split transaction for the current microframe.

The following figure illustrates the state machine for managing a complete interrupt split transaction. There are two phases to each split transaction. The first is a single start-split transaction, which occurs when the SplitXState is at Do_Start and the single bit in cMicroFrameBit has a corresponding bit active in QH.S-mask. The transaction translator does not acknowledge the receipt of the periodic start-split, so the host controller unconditionally transitions the state to Do_Complete. Due to the available jitter in the transaction translator pipeline, there will be more than one complete-split transaction scheduled by software for the Do_Complete state. This translates simply to the fact that there are multiple bits set to 1 in the QH.C-mask field.

The host controller keeps the QH in the Do_Complete state until the split transaction is complete (see definition below), or an error condition triggers the three-strikes-rule (for example, after the host tries the same transaction three times, and each encounters an error, the host controller will stop retrying the bus transaction and halt the endpoint, thus requiring system software to detect the condition and perform system-dependent recovery).

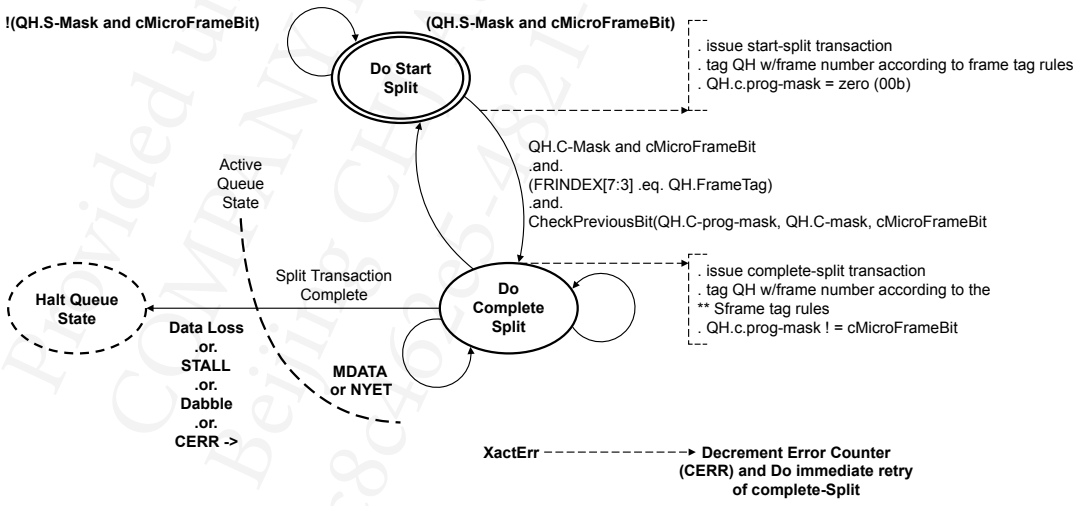


Figure 468. Split transaction state machine for interrupt

See the previous sections for the frame tag management rules.

55.10.4.2.5.1 Periodic interrupt—do start split

This is the state in which you must initialize a FS or LS interrupt QH StartXState bit. The state is entered from the Do_Complete Split state only after the split transaction is complete. This occurs when one of the following events occur. The transaction translator responds to a complete-split transaction with one of the following:

- NAK. A NAK response is a propagation of the FS or LS endpoint's NAK response.
- ACK. An ACK response is a propagation of the FS or LS endpoint's ACK response. It only occurs on an OUT endpoint.
- DATA 0/1. Only occurs for INs. Indicates that this is the last of the data from the endpoint for this split transaction.
- ERR. The transaction on the LS/FS link below the transaction translator had a failure (for example, timeout, bad CRC, and so on.).
- NYET (and Last). The host controller issued the last complete-split and the transaction translator responded with a NYET handshake. This means that the start-split was not correctly received by the transaction translator, so it never executed a transaction to the FS or LS endpoint. See Section Periodic isochronous - do complete split for the definition of "Last".

Each time the host controller visits a QH in this state (when within the Execute Transaction state), it performs the following test to determine whether to execute a start-split:

- QH.S-mask is bit-wise anded with cMicroFrameBit.

If the result is non-zero, then the host controller will issue a start-split transaction. If the PIDCode field indicates an IN transaction, the host controller must zero-out the QH.S-bytes field. After the split-transaction has been executed, the host controller sets up state in the QH to track the progress of the complete-split phase of the split transaction. Specifically, it records the expected frame number into the QH.FrameTag field, sets the C-prog-mask to 0 (00h), and exits this state. Note that the host controller must not adjust the value of CErr as a result of completion of a start-split transaction.

55.10.4.2.5.2 Periodic interrupt—do complete split

This state is entered unconditionally from the Do Start Split state after a start-split transaction is executed on the bus. Each time the host controller visits a QH in this state (when within the Execute Transaction state), it checks to determine whether a complete-split transaction must be executed at that time.

There are four tests to determine whether a complete-split transaction must be executed.

- Test A. cMicroFrameBit is bit-wise anded with QH.C-mask field. A non-zero result indicates that you scheduled a complete-split for this endpoint, during this microframe.
- Test B. QH.FrameTag is compared with the current contents of FRINDEX[7:3]. An equal indicates a match.
- Test C. The complete-split progress bit vector is checked to determine whether the previous bit is set, indicating that the previous complete-split was appropriately executed. An example algorithm for this test is provided below:

```
Algorithm Boolean CheckPreviousBit(QH.C-prog-mask, QH.C-mask, cMicroFrameBit)
Begin
-- Return values:
-- TRUE - no error
-- FALSE - error
--
Boolean rvalue = TRUE;
previousBit = cMicroframeBit logical-rotate-right(1)
-- Bit-wise anding previousBit with C-mask indicates
-- whether there was an intent
-- to send a complete split in the previous microframe. So,
-- if the
-- 'previous bit' is set in C-mask, check C-prog-mask to
-- make sure it
-- happened.
If (previousBit bitAND QH.C-mask)then
```

```
                                If not(previousBit bitAND QH.C-prog-mask) then
                        rvalue = FALSE;
              End if
End If
-- If the C-prog-mask already has a 1 in this bit position,
-- then an aliasing
-- error has occurred. It will probably be caught by the
-- FrameTag Test, but
-- at any rate it is an error condition that as detectable here
-- must not allow
-- a transaction to be executed.
If (cMicroFrameBit bitAND QH.C-prog-mask) then
 rvalue = FALSE;
End if
return (rvalue)
End Algorithm
```

- Test D. Check to see if a start-split should be executed in this microframe. Note this is the same test performed in the Do Start Split state (see Section Periodic isochronous - do start split). Whenever it evaluates to TRUE and the controller is NOT processing in the context of a Recovery Path mode, it means a start-split should occur in this microframe. Test D and Test A evaluating to TRUE at the same time is a system software error. Behavior is undefined.

If logically (A .and. B .and. C .and. not(D)) is true, then the host controller will execute a complete-split transaction. When the host controller commits to executing the complete-split transaction, it updates QH.C-prog-mask by performing a bit-OR with cMicroFrameBit. On completion of the complete-split transaction, the host controller records the result of the transaction in the QH and sets QH.FrameTag to the expected H-Frame number. The effect to the state of the QH and thus the state of the transfer depends on the response by the transaction translator to the complete-split transaction. The following responses have the given effects (note that any responses that result in decrementing of CErr will result in the QH being halted by the host controller if the result of the decrement is 0):

- NYET (and Last). On each NYET response, the host controller checks to determine whether this is the last complete-split for this split transaction. Last is defined in this context as the condition in which all of the scheduled complete-splits have been executed. If it is the last complete-split (with a NYET response), then the transfer state of the QH is not advanced (never received any data) and this state exited. The transaction translator must have responded to all the complete-splits with NYETs, meaning that the start-split issued by the host controller was not received. The start-split should be retried at the next poll period.
- The test for whether this is the Last complete split can be performed by XOR QH.C-mask with QH.C-prog-mask. If the result is all zeros then all complete-splits have been executed. When this condition occurs, the XactErr status bit is set to a 1 and the CErr field is decremented.
- NYET (and not Last). See the above description for testing for Last. The complete-split transaction received a NYET response from the transaction translator. Do not update any transfer state (except for C-prog-mask and FrameTag) and stay in this state. The host controller must not adjust CErr on this response.
- Transaction Error (XactErr). Timeout, data CRC failure, and so on. The CErr field is decremented and the XactErr bit in the Status field is set to a 1. The complete split transaction is immediately retried (if Cerr is non-zero). If there is not enough time in the microframe to complete the retry and the endpoint is an IN, or CErr is decremented to a 0 from a 1, the queue is halted. If there is not enough time in the microframe to complete the retry and the endpoint is an OUT and CErr is not 0, then this state is exited (that is, returned to Do Start Split). This results in a retry of the entire OUT split transaction, at the next poll period. See the Hubs (specifically, the "FS and LS interrupts") section in the USB Specification Revision 2.0 for detailed requirements on why these errors must be immediately retried.
- ACK. This can only occur if the target endpoint is an OUT. The target endpoint responded with ACK for the data, and this response is a propagation of the endpoint ACK up to the host controller. The host controller must advance the state of the transfer. The Current Offset field is incremented by Maximum Packet Length or Bytes to Transfer, whichever is less. The field Bytes To Transfer is decremented by the same amount. And the data toggle bit (DT) is toggled. The host controller will then exit this state for this QH. The host controller must reload CErr with the maximum value on this

response. Advancing the transfer state may cause other process events such as retirement of the qTD and advancement of the queue (see Managing control/bulk/interrupt transfers through QHs).

- MDATA. This response will only occur for an IN endpoint. The transaction translator responded with zero or more bytes of data and an MDATA PID. The incremental number of bytes received is accumulated in QH.S-bytes. The host controller must not adjust CErr on this response.
- DATA0/1. This response may only occur for an IN endpoint. The number of bytes received is added to the accumulated byte count in QH.S-bytes. The state of the transfer is advanced by the result and the host controller will exit this state for this QH.
- Advancing the transfer state may cause other processing events such as retirement of the qTD and advancement of the queue (see Managing control/bulk/interrupt transfers through QHs).
- If the data sequence PID does not match the expected, the entirety of the data received in this split transaction is ignored, the transfer state is not advanced, and this state is exited.
- NAK. The target endpoint generated a NAK response for the FS or LS transaction. The state of the transfer is not advanced, and this state is exited. The host controller must reload CErr with the maximum value on this response.
- ERR. There was an error during the FS or LS transaction. The ERR status bit is set to a 1, Cerr is decremented, the state of the transfer is not advanced, and this state is exited.
- STALL. The queue is halted (an exit condition of the Execute Transaction state). The Active status field is set to 0, the Halted status field is set to a 1, and the qTD is retired. Responses which are not enumerated in the list or which are received out of sequence are illegal and may result in undefined host controller behavior. The other possible combinations of tests A, B, C, and D may indicate that the data or the response was lost. The table below lists the possible combinations and the appropriate actions.

Table 447. Interrupt IN/OUT do complete split state execution criteria

Condition	Action	Description
not(A) not(D)	Ignore QHD	Neither a start nor complete-split is scheduled for the current microframe. The host controller should continue walking the schedule.
A not(C)	If PIDCode = IN then halt QHD If PIDCode = OUT then retry start-split	Progress bit check failed. This means a complete-split has been missed. There is the possibility of lost data. If PIDCode is an IN, then the QH must be halted. If PIDCode is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not effect CERR. In either case, set the Missed Microframe bit in the status field to a 1.
A not(B) C	If PIDCode = IN then halt QHD If PIDCode = OUT then retry start-split	QH.FrameTag test failed. This means that exactly one or more H-Frames have been skipped. This means complete-splits and have missed. There is the possibility of lost data. If PIDCode is an IN, then the QH must be halted. If PIDCode is an OUT, then the transfer state is not advanced and the state is exited (for example, start-split is retried). This is a host-induced error and does not effect CERR. In either case, set the Missed Microframe bit in the status field to a 1.
A B	Execute complete-split	This is the non-error case where the host controller executes a complete-split transaction.

Table continues on the next page...

Table 447. Interrupt IN/OUT do complete split state execution criteria (continued)

Condition	Action	Description
C not(D)		
D	If PIDCode = IN then halt QHD If PIDCode = OUT then retry start-split	This is a degenerate case where the start-split was issued, but all of the complete-splits were skipped and all possible intervening opportunities to detect the missed data failed to result in detection. If PIDCode is an IN, then the QH must be halted. If PIDCode is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not affect CERR. In either case, you must set the Missed Microframe bit in the status field to a 1. Note: When executing in the context of a Recovery Path mode, the host controller is allowed to process the QH and take the actions indicated above, or it may wait until the QH is visited in the normal processing mode. Regardless, the host controller must not execute a start-split in the context of a executing in a Recovery Path mode.

55.10.4.2.5.3 Managing the QH.FrameTag field

The QH.FrameTag field in a QH is completely managed by the host controller. The rules for setting QH.FrameTag are these (see USB Frame Index (FRINDEX)):

- Rule 1: If transitioning from Do Start Split to Do Complete Split and the current value of FRINDEX[2:0] is 6, then QH.FrameTag is set to FRINDEX[7:3] + 1. This accommodates split transactions whose start-split and complete-splits are in different H-Frames (case 2a, see Figure 465).
- Rule 2: If the current value of FRINDEX[2:0] is 7, then QH.FrameTag is set to FRINDEX[7:3] + 1. This accommodates staying in Do Complete Split for cases 2a, 2b, and 2c (see Figure 465).
- Rule 3: If transitioning from Do_Start Split to Do Complete Split and the current value of FRINDEX[2:0] is not 6, or currently in Do Complete Split and the current value of (FRINDEX[2:0]) is not 7, FrameTag is set to FRINDEX[7:3]. This accommodates all other cases (see Figure 465).

55.10.4.2.6 Rebalancing the periodic schedule

System software must occasionally adjust a periodic QH's S-mask and C-mask fields during operation.

This need occurs when adjustments to the periodic schedule create a new bandwidth budget and one or more QHs are assigned new execution footprints (that is, new S-mask and C-mask values).

It is imperative that the system software must not update these masks to new values in the midst of a split transaction. In order to avoid any race conditions with the update, the EHCI host controller provides a simple assist to system software. System software sets the Inactivate-on-next-Transaction (I) field to a 1 to signal the host controller that it intends to update the S-mask and C-mask on this QH. System software will then wait for the host controller to observe the I-bit is a 1 and transition the Active field to a 0. Following are the rules for how and when the host controller sets the Active field to 0:

- If the Active field is a 0, no action is taken. The host controller does not attempt to advance the queue when the I-bit is a 1.
- If the Active field is a 1 and the SplitXState is DoStart (regardless of the value of S-mask), the host controller will simply set the Active field to a 0. The host controller is not required to write the transfer state back to the current qTD. Note that if the S-mask indicates that a start-split is scheduled for the current microframe, the host controller must not issue the start-split bus transaction. It must set the Active field to 0.

The system software must save the transfer state before setting the I-bit to a 1. This is required so that it can correctly determine what transfer progress (if any) occurred after the I-bit was set to a 1 and the host controller executed its final bus transaction and set Active to 0.

After system software has updated the S-mask and C-mask, it must then reactivate the QH. Because the Active field and the I-bit cannot be updated with the same write, the system software needs to use the following algorithm to coherently reactivate a QH that has been stopped via the I-bit.

1. Set the Halted field to 1.
2. Set the I-bit to 0.
3. Set the Active field to 1 and the Halted field to 0 in the same write.

Setting the Halted field to 1 inhibits the host controller from attempting to advance the queue between the time the I-bit goes to 0 and the Active field goes to 1.

55.10.4.3 Split transaction isochronous

FS isochronous transfers are managed using the split-transaction protocol through a USB 2.0 transaction translator in a USB2.0 Hub. The EHCI controller uses the siTD data structure to support the special requirements of isochronous split-transactions.

This data structure uses the scheduling model of isochronous TDs (see iTD; also see Managing isochronous transfers using iTDs for the operational model of iTDs) with the contiguous data feature provided by queue heads. This arrangement allows a single isochronous scheduling model with the additional feature that all data received from the endpoint (per split transaction) must land into a contiguous buffer.

55.10.4.3.1 Split transaction scheduling mechanisms for isochronous

FS isochronous transactions are managed through a transaction translator's periodic pipeline. As with FS and LS interrupts, system software manages each transaction translator's periodic pipeline by budgeting and scheduling exactly during which microframes the start-splits and complete-splits for each FS isochronous endpoint occur.

The requirements described in Split transaction scheduling mechanisms for interrupt apply. The following figure illustrates the general scheduling boundary conditions that are supported by the EHCI periodic schedule. The S_X and C_X labels indicate microframes where you can schedule start- and complete-splits (respectively). The H-Frame boundaries are marked with a large, solid, bold vertical line. The B-Frame boundaries are marked with a large, bold, dashed line. The bottom of the figure illustrates the relationship of an siTD to the H-Frame.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a2-8251-390ca3062461

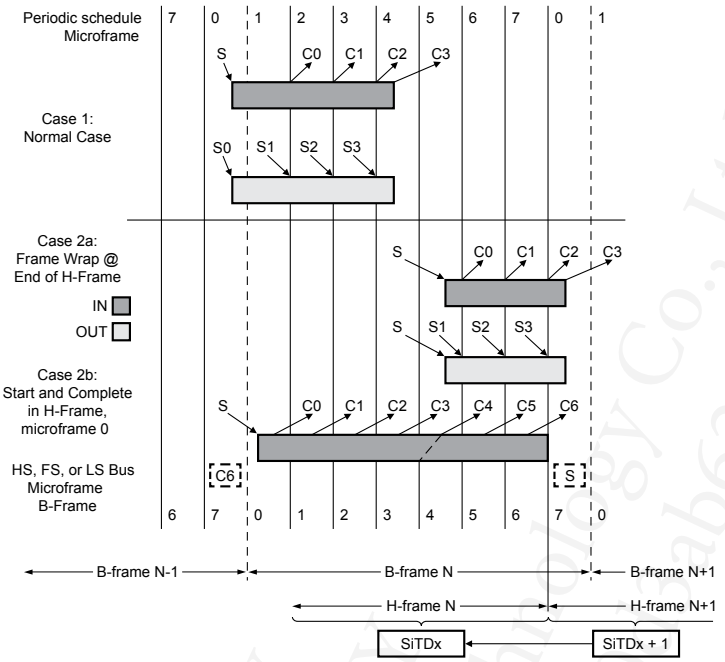


Figure 469. Split transaction, isochronous scheduling boundary conditions

When the endpoint is an isochronous OUT, there are only start-splits and no complete-splits. When the endpoint is an isochronous IN, there is at most one start-split and one to N complete-splits. The scheduling boundary cases are:

- Case 1: The entire split transaction is completely bounded by an H-Frame. For example, the start-splits and complete-splits are all scheduled to occur in the same H-Frame.
- Case 2a: In this boundary case, one or possibly two complete-splits of a split transaction IN are scheduled across an H-Frame boundary. This can only occur when the split transaction has the possibility of moving data in B-Frame, microframes 6 or 7 (H-Frame microframe 7 or 0). When an H-Frame boundary wrap condition occurs, the scheduling of the split transaction spans more than one location in the periodic list. (for example, it takes two siTDs in adjacent periodic frame list locations to fully describe the scheduling for the split transaction.)
- Although the scheduling of the split transaction may take two data structures, all of the complete-splits for each FS IN isochronous transaction must use only one data pointer. For this reason, siTDs contain a back pointer, the use of which is described below.
- You must never schedule FS isochronous OUTs across an H-Frame boundary.
- Case 2b: This case can only occur for a very large isochronous IN, and is the only allowed scenario in which a start-split and complete-split for the same endpoint can occur in the same microframe. You must enforce this rule by scheduling the large transaction first. Large is defined to be anything larger than 579-byte maximum packet size.

A subset of the same mechanisms employed by full- and LS interrupt queue heads are employed in siTDs to schedule and track the portions of isochronous split transactions. The following fields are initialized by the system software to instruct the host controller when to execute portions of the split transaction protocol.

- SplitXState. This is a single bit residing in the Status field of an siTD (see Figure 470). This bit is used to track the current state of the split transaction. The rules for managing this bit are described in Split transaction execution state machine for interrupt.
- Frame S-mask. This is a field in which the system software sets a bit corresponding to the microframe (within an H-Frame) for which the host controller should execute a start-split transaction. This is always qualified by the value of the SplitXState bit. For example, referring to the IN example in Figure 469, Case one, the S-mask would have a value of 00000001b, indicating that if the siTD is traversed by the host controller, and the SplitXState indicates Do Start Split, and the current microframe as indicated by USB_*n*_FRINDEX[2:0] is 0, then execute a start-split transaction.

- Frame C-mask. This is a field in which system software sets one or more bits corresponding to the microframes (within an H-Frame) for which the host controller should execute complete-split transactions. The interpretation of this field is always qualified by the value of the SplitXState bit. For example, referring to the IN example in Figure 469, Case one, the C-mask would have a value of 00111100b, indicating that if the siTD is traversed by the host controller, and the SplitXState indicates Do Complete Split, and the current microframe as indicated by USB_*n*_FRINDEX[2:0] is 2, 3, 4, or 5, then execute a complete-split transaction.
- Back Pointer. This field in an siTD is used to complete an IN split-transaction using the previous H-Frame's siTD. This is only used when the scheduling of the complete-splits span an H-Frame boundary.

There exists a one-to-one relationship between a HS isochronous split transaction (including all start- and complete-splits) and one FS isochronous transaction. An siTD contains (among other things) buffer state and split transaction scheduling information. An siTD's buffer state always maps to one FS isochronous data payload. This means that for any FS transaction payload, a single siTD's data buffer must be used. This rule applies to both IN and OUT transactions. An siTD's scheduling information usually also maps to one HS isochronous split transaction. The exception to this rule is the H-Frame boundary wrap cases mentioned above.

The siTD data structure describes, at most, one frame's worth of HS transactions, and that description is strictly bounded within a frame boundary. The figure below illustrates some examples. On the top are examples of the FS transaction footprints for the boundary scheduling cases described above. In the middle are time-frame references for both the B-Frames (HS, FS, or LS Bus) and the H-Frames. On the bottom is illustrated the relationship between the scope of an siTD description and the time references. Each H-Frame corresponds to a single location in the periodic frame list. The implication is that each siTD is reachable from a single periodic frame list location at a time.

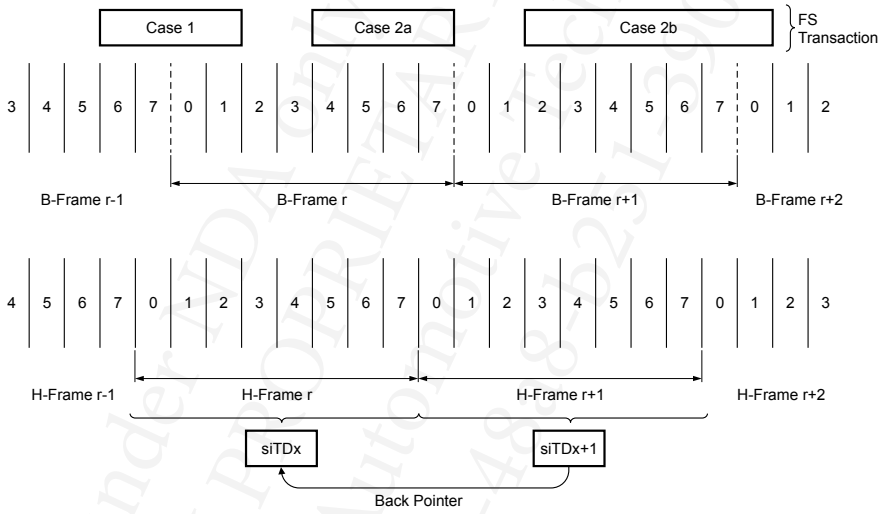


Figure 470. siTD scheduling boundary examples

Each case is described below:

- Case 1: One siTD is sufficient to describe and complete the isochronous split transaction because the whole isochronous split transaction is tightly contained within a single H-Frame.
- Case 2a, 2b: Although both INs and OUTs can have these footprints, OUTs always take only one siTD to schedule. However, INs (for these boundary cases) require two siTDs to complete. The scheduling of the isochronous split transaction $siTD_X$ is used to always issue the start-split and the first *N* complete-splits. The FS transaction (for these cases) can deliver data on the FS bus segment during microframe 7 of $H\text{-}Frame_{Y+1}$, or microframe 0 of $H\text{-}Frame_{Y+2}$. The complete splits are scheduled using $siTD_{X+2}$ (not shown). The complete-splits used to extract this data must use the buffer pointer from $siTD_{X+1}$. The only way for the host controller to reach $siTD_{X+1}$ from $H\text{-}Frame_{Y+2}$ is to use $siTD_{X+2}$'s back pointer. The host controller rules for when to use the back pointer are described in Periodic isochronous - do complete split .

You must apply the following rules when calculating the schedule and linking the schedule data structures into the periodic schedule:

- You must ensure that an isochronous split-transaction is started so that it will complete before the end of the B-Frame.
- You must ensure that, for a single FS isochronous endpoint, there is never a start-split and complete-split in H-Frame, microframe 1. This is mandated as a rule so that Case 2a and Case 2b can be discriminated. According to the core USB specification, the long isochronous transaction illustrated in Case 2b, could be scheduled so that the start-split is in microframe 1 of H-Frame N and the last complete-split would need to occur in microframe 1 of H-Frame N+1. However, it is impossible to discriminate between Case 2a and Case 2b, which has a significant impact on the complexity of the host controller.

55.10.4.3.2 Tracking split transaction progress for isochronous transfers

To correctly maintain the data stream, the host controller must be able to detect and report errors where device-to-host data is lost. Isochronous endpoints do not employ the concept of "halt on error"; however, the host is required to identify and report per-packet errors observed in the data stream. This includes schedule traversal problems (skipped microframes), timeouts, and corrupted data received.

As is the case for interrupt split-transactions, the portions of the split transaction protocol must execute in the microframes they are scheduled. The QH data structure used to manage FS and LS interrupts has several mechanisms for tracking when portions of a transaction have occurred. Isochronous transfers use siTDs for their transfers, and the data structures are only reachable via the schedule in the exact microframe in which they are required (therefore, all of the mechanisms employed for tracking in QHs are not required for siTDs). You have the option of reusing an siTD several times in the complete periodic schedule. However, it must ensure that the results of split transaction N are consumed and the siTD reinitialized (activated) before the host controller returns to the siTD (in a future microframe).

Split-transaction isochronous OUTs use a low-level protocol to indicate which portions of the split transaction data have arrived. Control over the low-level protocol is exposed in an siTD via the fields TP and T-Count. If the entire data payload for the OUT split transaction is larger than 188 bytes, there will be more than one start-split transaction, each of which require proper annotation. If host hold-offs occur, then the sequence of annotations received from the host will not be complete, which is detected and managed by the transaction translator. See Periodic isochronous - do start split for a description of how these fields are used during a sequence of start-split transactions.

The fields siTD.T-Count and siTD.TP are used by the host controller to drive and sequence the TP annotations. It is the responsibility of the system software to properly initialize these fields in each siTD. After the budget for a split-transaction isochronous endpoint is established, S-mask, T-Count, and TP initialization values for all the siTD associated with the endpoint are constant, and they remain constant until you recalculate the budget for the endpoint and the periodic schedule is adjusted.

For IN-endpoints, the transaction translator simply annotates the response data packets with enough information to allow the host controller to identify the last data. As with a split transaction interrupt, it is the host controller's responsibility to detect when it has missed an opportunity to execute a complete-split. The following field in the siTD is used to track and detect errors in the execution of a split transaction for an IN isochronous endpoint.

- C-prog-mask. This is an eight-bit bit-vector in which the host controller keeps track of which complete-splits have been executed. Due to the nature of the Transaction Translator periodic pipeline, the complete-splits need to be executed in order. The host controller needs to detect when the complete-splits have not been executed in order. This can only occur due to system hold-offs for which the host controller cannot access the memory-based schedule. C-prog-mask is a simple bit-vector for which the host controller sets a field for each complete-split executed. The bit position is determined by the microframe (USB_n_FRINDEX[2:0]) number in which the complete-split was executed. The host controller always checks the C-prog-mask before executing a complete-split transaction. If the previous complete-splits have not been executed, then it means that one (or more) have been skipped and data has potentially been lost. The system software must initialize this field to 0 before setting the Active field of an siTD to a 1.

If a transaction translator returns with the final data before all of the complete-splits have been executed, the state of the transfer is advanced so that the remaining complete-splits are not executed. See Asynchronous—do complete split for a description on how the state of the transfer is advanced. It is important to note that an IN siTD is retired based solely on the responses from the Transaction Translator to the complete-split transactions. This means, for example, that it is possible for a transaction translator to respond to a complete-split with an MDATA PID. The number of bytes in the MDATA's data payload could cause the siTD field Total Bytes to Transfer to decrement to 0. This response can occur before all of the scheduled complete-splits have been executed. In other interface data structures (for example, HS data streams through QHs), the transition of Total Bytes to Transfer to 0 signals the end of the transfer and results in setting of the Active field to 0. However, in this case, the result has

not been delivered by the Transaction Translator and the host must continue with the next complete-split transaction to extract the residual transaction state. This scenario occurs because of the pipeline rules for a Transaction Translator (see Chapter 11 of the Universal Serial Bus Revision 2.0). In summary, the periodic pipeline rules require that, on a microframe boundary, the Transaction Translator will hold the final two bytes received (if it has not seen an End Of Packet (EOP)) in the FS bus pipe stage and give the remaining bytes to the HS pipeline stage. At the microframe boundary, the Transaction Translator could have received the entire packet (including both CRC bytes) but not received the packet EOP. In the next microframe, the Transaction Translator will respond with an MDATA and send all of the data bytes (with the two CRC bytes being held in the FS pipeline stage). This could cause the siTD to decrement its Total Bytes to Transfer field to 0, indicating it has received all expected data. The host must still execute one more (scheduled) complete-split transaction in order to extract the results of the FS transaction from the Transaction Translator (for example, the Transaction Translator may have detected a CRC failure, and this result must be forwarded to the host).

If the host experiences hold-offs that cause the host controller to skip one or more (but not all) scheduled split transactions for an isochronous OUT, then the protocol to the transaction translator will not be consistent and the transaction translator will detect and react to the problem. Likewise, for host hold-offs that cause the host controller to skip one or more (but not all) scheduled split transactions for an isochronous IN, the C-prog-mask is used by the host controller to detect errors. However, if the host experiences a hold-off that causes it to skip all of an siTD, or an siTD expires during a host hold off (for example, a hold-off occurs and the siTD is no longer reachable by the host controller in order for it to report the hold-off event), then the system software must detect that the siTDs have not been processed by the host controller (that is, state not advanced) and report the appropriate error to the client driver.

55.10.4.3.3 Split transaction execution state machine for isochronous

In the following presentation, all references to microframe are in the context of a microframe within an H-Frame.

If the Active field in the Status byte is a 0, the host controller will ignore the siTD and continue traversing the periodic schedule. Otherwise, the host controller will process the siTD as specified below. A split transaction state machine is used to manage the split-transaction protocol sequence. The host controller uses the fields defined in Tracking split transaction progress for interrupt transfers, plus the variable cMicroFrameBit defined in Section Split transaction execution state machine for interrupt to track the progress of an isochronous split transaction. The figure below illustrates the state machine for managing an siTD through an isochronous split transaction. Bold, dotted circles denote the state of the Active field in the Status field of an siTD. The bold, dotted arcs denote the transitions between these states. Solid circles denote the states of the split transaction state machine and the solid arcs denote the transitions between these states. Dotted arcs and boxes reference actions that take place either as a result of a transition or from being in a state.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-[illegible]

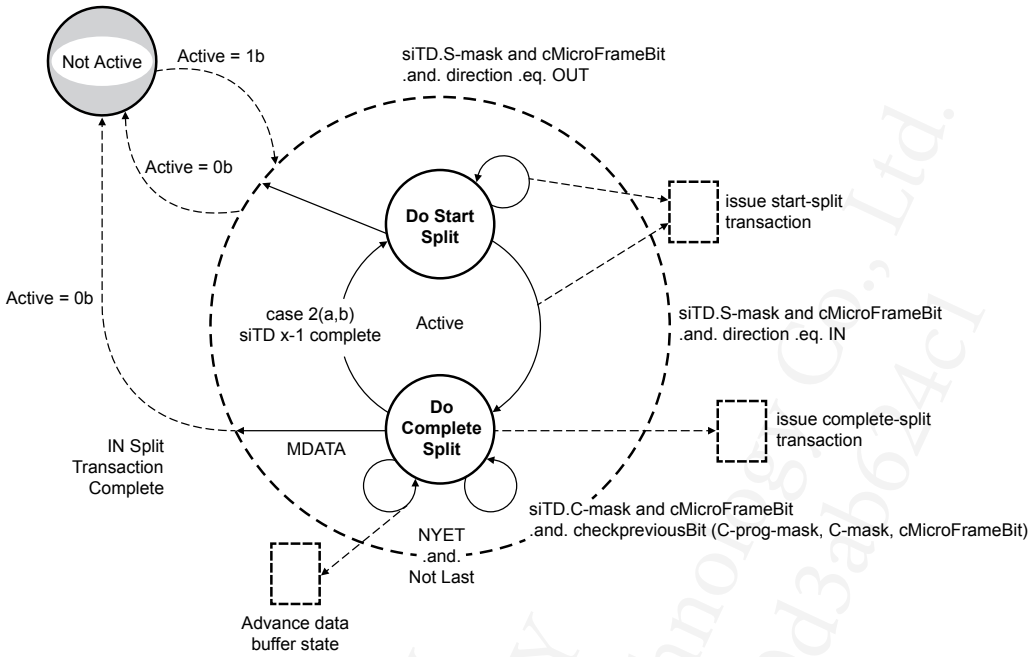


Figure 471. Split transaction state machine for isochronous

55.10.4.3.4 Periodic isochronous - do start split

Isochronous split OUT transactions use only this state.

An siTD for a split-transaction isochronous IN is either initialized to this state, or the siTD transitions to this state from Do Complete Split when a case 2a (IN) or 2b scheduling boundary isochronous split-transaction completes.

Each time the host controller reaches an active siTD in this state, it checks the siTD.S-mask against cMicroFrameBit. If there is a 1 in the appropriate position, the siTD executes a start-split transaction. By definition, the host controller cannot reach an siTD at the wrong time. If the I/O field indicates an IN, then the start-split transaction includes only the extended token plus the FS token. You must initialize the siTD.Total Bytes To Transfer field to the number of bytes expected. This is usually the maximum packet size for the FS endpoint. The host controller exits this state when the start-split transaction is complete.

The remainder of this section is specific to an isochronous OUT endpoint (that is, the I/O field indicates an OUT). When the host controller executes a start-split transaction for an isochronous OUT, it includes a data payload in the start-split transaction. The memory buffer address for the data payload is constructed by concatenating siTD.Current Offset with the page pointer indicated by the page selector field (siTD.P). The value of 0 in this field selects Page 0 and a 1 selects Page 1. During the start-split for an OUT, if the data transfer crosses a page boundary during the transaction, the host controller must detect the page cross, update the siTD.P-bit from a 0 to a 1, and begin using the siTD.Page 1 with siTD.Current Offset as the memory address pointer. The field siTD.TP is used to annotate each start-split transaction with the indication of which part of the split-transaction data the current payload represents (ALL, BEGIN, MID, END). In all cases, the host controller simply uses the value in siTD.TP to mark the start-split with the correct TP code.

T-Count is always initialized to the number of start-splits for the current frame. TP is always initialized to the first required TP identifier. The scheduling boundary case (see Split transaction scheduling mechanisms for isochronous) is used to determine the initial value of TP. The initial cases are summarized in the following table.

Table 448. Initial conditions for OUT siTD's TP and T-Count fields

Case	T-Count	TP	Description
1, 2a	=1	ALL	When the OUT data payload is less than (or equal to) 188 bytes, only one start-split is required to move the data. The one start-split must be marked with an ALL.
1, 2a	!=1	BEGIN	When the OUT data payload is greater than 188 bytes, more than one start-split must be used to move the data. The initial start-split must be marked with a BEGIN.

After each start-split transaction is complete, the host controller updates T-Count and TP appropriately so that the next start-split is correctly annotated.

The table below illustrates all of the TP and T-Count transitions, which must be accomplished by the host controller.

Table 449. TP/T-Count transition table

TP	T-Count next	TP next	Description
ALL	0	N/A	Transition from ALL, to done.
BEGIN	1	END	Transition from BEGIN to END. Occurs when T-Count starts at 2.
BEGIN	!=1	MID	Transition from BEGIN to MID. Occurs when T-Count starts at greater than 2.
MID	!=1	MID	TP stays at MID when T-Count is not equal to 1 (that is, greater than 1). This case can occur for any of the scheduling boundary cases where the T-Count starts greater than 3.
MID	1	END	Transition from MID to END. This case can occur for any of the scheduling boundary cases where the T-Count starts greater than 2.

The start-split transactions do not receive a handshake from the transaction translator, so the host controller always advances the transfer state in the siTD after the bus transaction is complete. To advance the transfer state, the following operations take place:

- The siTD.Total Bytes to Transfer and the siTD.Current Offset fields are adjusted to reflect the number of bytes transferred.
- The siTD.P (page selector) field is updated appropriately.
- The siTD.TP and siTD.T-Count fields are updated appropriately as defined in Table 449.

These fields are then written back to the memory-based siTD. The S-mask is fixed for the life of the current budget. As mentioned above, TP and T-Count are set specifically in each siTD to reflect the data to be sent from this siTD. Therefore, regardless of the value of S-mask, the actual number of start-split transactions depends on T-Count (or equivalently, Total Bytes to Transfer). The host controller must set the Active field to a 0 when it detects that all of the schedule data has been sent to the bus. The preferred method is to detect when T-Count decrements to 0 as a result of a start-split bus transaction. Equivalently, the host controller can detect when Total Bytes to Transfer decrements to 0. Either implementation must ensure that if the initial condition is Total Bytes to Transfer equal to 0 and T-Count is equal to a 1, then the host controller will issue a single start-split, with a zero-length data payload. You must ensure that TP, T-Count, and Total Bytes to Transfer are set to deliver the appropriate number of bus transactions from each siTD. An inconsistent combination will yield undefined behavior.

If the host experiences hold-offs that cause the host controller to skip start-split transactions for an OUT transfer, the state of the transfer will not progress appropriately. The transaction translator will observe protocol violations in the arrival of the start-splits for the OUT endpoint (that is, the TP annotation will be incorrect as received by the transaction translator).

A host controller implementation can optionally track the progress of an OUT split transaction by setting appropriate bits in the siTD.C-prog-mask as it executes each scheduled start-split. The checkPreviousBit() algorithm defined in Periodic isochronous -

do complete split can be used prior to executing each start-split to determine whether start-splits were skipped. The host controller can use this mechanism to detect missed microframes. It can then set the siTD's Active field to 0 and stop execution of this siTD. This saves on both memory and HS bus bandwidth.

55.10.4.3.5 Periodic isochronous - do complete split

The state is only used by a split-transaction isochronous IN endpoint.

This state is entered unconditionally from the Do Start State after a start-split transaction is executed for an IN endpoint. Each time the host controller visits an siTD in this state, it conducts a number of tests to determine whether it should execute a complete-split transaction. The individual tests are listed below. The sequence in which they are applied depends on which microframe the host controller is currently executing, meaning that the tests might not be applied until after the siTD referenced from the back pointer has been fetched.

- Test A. cMicroFrameBit is bit-wise ANDed with the siTD.C-mask field. A non-zero result indicates that you scheduled a complete-split for this endpoint, during this microframe. This test is always applied to a newly-fetched siTD that is in this state.
- Test B. The siTD.C-prog-mask bit vector is checked to determine whether the previous complete splits have been executed. The sequence in which this test is applied depends on the current value of USB_*n*_FRINDEX[2:0]. If USB_*n*_FRINDEX[2:0] is 0 or 1, it is not applied until the back pointer has been used; otherwise, it is applied immediately.

```
Algorithm Boolean CheckPreviousBit(siTD.C-prog-mask, siTD.C-mask, cMicroFrameBit)
Begin
          Boolean rvalue = TRUE;
          previousBit = cMicroFrameBit rotate-right(1)
          -- Bit-wise anding previousBit with C-mask indicates whether there was an intent
          -- to send a complete split in the previous microframe. So, if the
          -- 'previous bit' is set in C-mask, check C-prog-mask to make sure it
          -- happened.
          if previousBit bitAND siTD.C-mask then
                    if not (previousBit bitAND siTD.C-prog-mask) then
                              rvalue = FALSE
                    End if
          End if
          Return rvalue
End Algorithm
```

If Test A is true and the value of USB_n_FRINDEX[2:0] is 0 or 1, then this is a case 2a or 2b scheduling boundary (see Figure 469).

If Test A and Test B evaluate to true, then the host controller executes a complete-split transaction using the transfer state of the current siTD. When the host controller commits to executing the complete-split transaction, it updates QH.C-prog-mask by performing a bit-OR with the cMicroFrameBit. The transfer state is advanced based on the completion status of the complete-split transaction. To advance the transfer state of an IN siTD, the host controller must:

- Decrement the number of bytes received from siTD.Total Bytes to Transfer,
- Adjust siTD.Current Offset by the number of bytes received,
- Adjust the siTD.P (page selector) field if the transfer caused the host controller to use the next page pointer, and
- Set any appropriate bits in the siTD.Status field, depending on the results of the transaction.

Note that if the host controller encounters a condition where siTD.Total Bytes To Transfer is zero, and it receives more data, the host controller must not write the additional data to memory. The siTD.Status.Active bit must be set to 0 and the siTD.Status.Babble Detected bit must be set to a 1. The fields siTD.Total Bytes to Transfer, siTD.Current Offset, and siTD.P (page selector) are not required to be updated as a result of this transaction attempt.

The host controller must accept (assuming good data packet CRC and sufficient room in the buffer as indicated by the value of siTD.Total Bytes To Transfer) MDATA and DATA0/1 data payloads up to and including 192 bytes. A host controller implementation may optionally set siTD.Status Active to 0 and siTD.Status.Babble Detected to 1 when it receives an MDATA or DATA0/1 with a data payload of more than 192 bytes. The following responses have these effects:

- ERR. FS transaction completed with a time-out or bad CRC and this is a reflection of that error to the host. The host controller sets the ERR bit in the siTD.Status field and sets the Active bit to 0.
- Transaction Error (XactErr). The complete-split transaction encounters a Timeout, CRC16 failure, and so on. The siTD.Status field XactErr field is set to a 1 and the complete-split transaction must be retried immediately. The host controller must use an internal error counter to count the number of retries as a counter field is not provided in the siTD data structure. The host controller will not retry more than two times. If the host controller exhausts the retries or the end of the microframe occurs, the Active bit is set to 0.
- DATAx (0 or 1). This response signals that the final data for the split transaction has arrived. The transfer state of the siTD is advanced and the Active bit is set to 0. If the Bytes To Transfer field has not decremented to 0 (including the reception of the data payload in the DATAx response), then less data than was expected, or allowed for, was actually received. This short packet event does not set USBSTS[UI] to 1. The host controller will not detect this condition.
- NYET (and Last). On each NYET response, the host controller also checks to determine whether this is the last complete-split for this split transaction. If it is the last complete-split (with a NYET response), then the transfer state of the siTD is not advanced (never received any data) and the Active bit is set to 0. No bits are set in the Status field because this is essentially a skipped transaction. The transaction translator must have responded to all the scheduled complete-splits with NYETs, meaning that the start-split issued by the host controller was not received. This result must be interpreted by system software as if the transaction was completely skipped. The test for whether this is the last complete split can be performed by XORing C-mask with C-prog-mask. A zero result indicates that all complete-splits have been executed.
- MDATA (and Last). See the above description for testing for Last. This can only occur when there is an error condition. Either there has been a babble condition on the FS link, which delayed the completion of the FS transaction, or you set up the S-mask and/or C-masks incorrectly. The host controller must set the XactErr bit to 1 and the Active bit is set to 0.
- NYET (and not Last). See the above description for testing for Last. The complete-split transaction received a NYET response from the transaction translator. Do not update any transfer state (except for C-prog-mask), and stay in this state.
- MDATA (and not Last). The transaction translator responds with an MDATA when it has partial data for the split transaction. For example, the FS transaction data payload spans from microframe X to X+1, and during microframe X the transaction translator will respond with an MDATA and the data accumulated up to the end of microframe X. The host controller advances the transfer state to reflect the number of bytes received.

If Test A succeeds, but Test B fails, it means that one or more of the complete-splits have been skipped. The host controller sets the Missed Microframe status bit and sets the Active bit to 0.

55.10.4.3.6 Complete-split for scheduling boundary cases 2a, 2b

Boundary cases 2a and 2b (INs only) (see Figure 469) require that the host controller use the transaction state context of the previous siTD to finish the split transaction. The table below enumerates the transaction state fields.

Table 450. Summary of siTD Split Transaction state

Buffer state	Status	Execution progress
Total Bytes to Transfer P (page select) Current Offset TP T-Count	All bits in the status field	C-prog-mask

NOTE

TP and T-Count are used only for Host to Device (OUT) endpoints.

If you have budgeted the schedule of this data stream with a frame wrap case, then it must initialize the siTD.Back Pointer field to reference a valid siTD and will have the siTD.Back Pointer.T-bit in the siTD.Back Pointer field cleared to 0. Otherwise, you

must set the siTD.Back Pointer.T-bit in the siTD.Back Pointer field to 1. The host controller's rules for interpreting when to use the siTD.Back Pointer field are listed below. These rules apply only when the value of siTD's Active field is 1 and the SplitXState is Do Complete Split:

- When the cMicroFrameBit is a 1h and the siTDX.Back Pointer.T-bit is 0
- If the cMicroFrameBit is a 2h and the value of siTDX.S-mask[0] is 0

When either of the above conditions apply, then the host controller must use the transaction state from $siTD_{X-1}$.

To access $siTD_{X-1}$, the host controller reads on-chip the siTD referenced from $siTD_X$.Back Pointer.

The host controller must save the entire state from $siTD_X$ when processing $siTD_{X-1}$. This is to accommodate for Case 2b processing. The host controller must not recursively walk the list of siTD.Back Pointers.

If $siTD_{X-1}$ is active (the Active field is 1 and SplitXStat is Do Complete Split), then both Test A and Test B are applied as described above. If these criteria to execute a complete-split are met, the host controller executes the complete split and evaluates the results as described above. The transaction state (see Table 450) of $siTD_{X-1}$ is appropriately advanced based on the results and written back to memory. If the resultant state of $siTD_{X-1}$'s Active field is 1, then the host controller returns to the context of $siTD_X$, and follows its next pointer to the next schedule item. No updates to $siTD_X$ are necessary.

If $siTD_{X-1}$ is active (the Active field is 1 and SplitXStat is Do Start Split), then the host controller must clear the Active field to 0, set the Missed Microframe status field to 1, and write the resultant status back to memory.

If $siTD_{X-1}$'sActive field is 0 (because it was 0 when the host controller first visited $siTD_{X-1}$ via $siTD_X$'s back pointer, it transitioned to 0 as a result of a detected error, or the results of $siTD_{X-1}$'s complete-split transaction transitioned it to 0), then the host controller returns to the context of $siTD_X$ and transitions its SplitXState to Do Start Split. The host controller then determines whether the Case 2b start split boundary condition exists (that is, if cMicroframeBit is a 1b and $siTD_X$.S-mask[0] is a 1b). If this criterion is met, the host controller immediately executes a start-split transaction and appropriately advances the transaction state of $siTD_X$, then follows $siTD_X$.Next Pointer to the next schedule item. If the criterion is not met, the host controller simply follows $siTD_X$.Next Pointer to the next schedule item. Note that in the case of a 2b boundary case, the split-transaction of $siTD_{X-1}$ will have its Active field cleared to 0 when the host controller returns to the context of $siTD_X$. Also, note that you must not initialize an siTD with C-mask bits 0 and 1 set to 1 and an S-mask with bit 0 set to 1. This scheduling combination is not supported and the resulting behavior of the host controller is undefined.

55.11 Deviations from the EHCI specifications

The Host mode operation of the USB host module is mostly EHCI-compatible, with a few minor differences. For the most part, the modules conform to the data structures and operations described in Section 3, "Data structures," and Section 4, "Operational model," in the EHCI specification. The particulars of the deviations occur in the following areas:

- Embedded transaction translator - Allows direct attachment of HS, FS and LS devices in Host mode without the need for a companion controller.
- Embedded design interface - The module does not have a PCI Interface and therefore the PCI configuration registers described in the EHCI specification are not applicable.

55.11.1 Embedded transaction translator function

The USB Host mode supports directly-connected HS, FS, and LS devices without requiring a companion controller by including the capabilities of a USB 2.0 HS hub transaction translator. Although there is no separate transaction translator block in the system, the transaction translator function normally associated with a HS hub is implemented within the DMA and protocol engine blocks. The embedded transaction translator function is an extension to the EHCI interface, but makes use of the standard data structures and operational models in the EHCI specification to support FS and LS devices.

55.11.1.1 Capability registers

These additions to the capability registers support the embedded Transaction Translator function:

- N_TT added to HCSPARAMS - Host controller structural parameters
- N_PTT added to HCSPARAMS - Host controller structural parameters

See Host Controller Structural Parameters (HCSPARAMS) for usage information.

55.11.1.2 Operational registers

Operational registers are defined in the Universal Serial Bus Specification, Revision 2.0, in the section "Host Controller Operational Registers".

The following additions to the operational registers provide support for the embedded TT:

- Addition of a 2-bit port speed (PSPD) field to the PORTSC*n* register

55.11.1.3 Discovery

In a standard EHCI controller design, the EHCI host controller driver detects an FS or LS device by noting if the port enable field is set after the port reset operation. In a standard EHCI controller implementation, the port enable is set only after the port reset operation and when the host and device negotiate a HS connection (chirp completes successfully).

The module always sets the port enable field after the port reset operation regardless of the host device chirp result, and the resulting port speed is indicated by the PORTSC*n*[PSPD] field. Therefore, the standard EHCI host controller driver requires an alteration to manage directly-connected FS and LS devices or hubs. The change is a fundamental one that is summarized in the next table.

Table 451. Functional differences between EHCI and EHCI with embedded TT

Standard EHCI	EHCI with embedded transaction translator
After the port enable field is set following a connection and reset sequence, the device/hub is assumed to be HS.	After the port enable field is set following a connection and reset sequence, the device/hub speed is noted from PORTSCn.
FS and LS devices are assumed to be downstream from an HS hub. Therefore, all port-level control performs through the hub class to the nearest hub.	FS and LS devices can be downstream from an HS hub or directly attached. When the FS or LS device is downstream from an HS hub, port-level control acts using the hub class through the nearest hub. When an FS or LS device is directly attached, then port-level control is accomplished using PORTSCn.
FS and LS devices are assumed to be downstream from a HS hub with HubAddr equal to X [where HubAddr > 0 and HubAddr is the address of the hub where the bus transitions from HS to FS or LS (split target hub)]	FS and LS devices can be downstream from an HS hub with HubAddr equal to X [HubAddr > 0] or directly attached [where HubAddr equals 0 and HubAddr is the address of the root hub where the bus transitions from HS to FS or LS (split target hub is the root hub)]

55.11.1.4 Data structures

The same data structures used for FS or LS transactions though a HS hub are also used for transactions through the root hub. It is demonstrated here how the hub address and endpoint speed fields must be set for directly-attached FS or LS devices and hubs:

- QH (for direct-attach FS or LS) - asynchronous (bulk/control endpoints) periodic (interrupt)
  - Hub address equals 0
  - Transactions to direct-attached device/hub.
    - QH.EPS equals port speed
  - Transactions to a device downstream from direct-attached FS hub.
    - QH.EPS equals downstream device speed

NOTE

When QH.EPS equals 01 (LS) and PORTSC*n*[PSPD] equals 00 (FS), a LS-pre-PID is sent before transmitting LS traffic.

Maximum packet size must equal 64 or less to prevent undefined behavior.

- siTD (for direct-attach FS) - Periodic (ISO endpoint)
  - All FS ISO transactions:
    - Hub address equals 0
    - siTD.EPS equals 00 (FS)

  Maximum packet size must equal 1023 or less to prevent undefined behavior.

55.11.1.5 Operational model

The operational models are well-defined for the behavior of the transaction translator (see the USB 2.0 specification) and for the EHCI controller moving packets between system memory and a USB-HS hub. Because the embedded transaction translator exists within the USB host controller, no physical bus exists between the EHCI host controller driver and the USB FS or LS bus. These sections briefly discuss the operational model for how the EHCI and transaction translator operational models combine without the physical bus between them. The following sections assume that you are familiar with the EHCI and USB 2.0 transaction translator operational models.

55.11.1.5.1 Microframe pipeline

The EHCI operational model uses the concept of H-frames and B-frames to describe the pipeline between the host (H) and the bus (B). The embedded transaction translator uses the same pipeline algorithms specified in the USB 2.0 specification for a hub-based transaction translator.

All periodic transfers always begin at B-frame 0 (after SOF) and continue until the stored periodic transfers are complete. As an example of the microframe pipeline implemented in the embedded transaction translator, all periodic transfers that are tagged in EHCI to execute in H-frame 0 are ready to execute on the bus in B-frame 0.

When programming the S-mask and C-masks in the EHCI data structures to schedule periodic transfers for the embedded transaction translator, the EHCI host controller driver must follow the same rules specified in EHCI for programming the S-mask and C-mask for downstream hub-based transaction translators.

After periodic transfers are exhausted, any stored asynchronous transfer is moved. Asynchronous transfers are opportunistic because they execute when possible and their operation is not tied to H-frame and B-frame boundaries with the exception that an asynchronous transfer cannot babble through the SOF (start of B-frame 0.)

55.11.1.5.2 Split-state machines

The start and complete-split operational model differs from EHCI slightly because there is no bus medium between the EHCI controller and the embedded transaction translator. Where a start or complete-split operation would occur by requesting the split to the HS hub, the start/complete-split operation is a simple internal operation to the embedded transaction translator. The next table summarizes the conditions where handshakes are emulated from internal states instead of actual handshakes to HS-split bus traffic.

Table 452. Emulated handshakes

Condition	Emulated TT response
Start-Split: All asynchronous buffers are full	NAK
Start-Split: All periodic buffers are full	ERR

Table continues on the next page...

Table 452. Emulated handshakes (continued)

Condition	Emulated TT response
Start-Split: Success for start of asynchronous transaction	ACK
Start-Split: Start periodic transaction	No handshake (Okay)
Complete-Split: Failed to find transaction in queue	Bus time-out
Complete-Split: Transaction in queue is busy	NYET
Complete-Split: Transaction in queue is complete	Actual handshake from FS or LS device

55.11.1.5.3 Asynchronous transaction scheduling and buffer management

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

- USB 2.0 - 11.17.3
  - Sequencing is provided and a packet length estimator ensures no FS or LS packet babbles into SOF time.
- USB 2.0 - 11.17.4
  - Transaction tracking for two data pipes.
- USB 2.0 - 11.17.5
  - Clear_TT_Buffer capability

55.11.1.5.4 Periodic transaction scheduling and buffer management

The following USB 2.0 specification items are implemented in the embedded transaction translator:

- USB 2.0 - 11.18.6.[1-2]
  - Abort of pending start-splits
    - EOF (and not started in microframes 6)
    - Idle for more than four microframes
  - Abort of pending complete-splits
    - EOF
    - Idle for more than four microframes
- USB 2.0 - 11.18.[7-8]
  - Transaction tracking for up to four data pipes
    - No more than four periodic transactions (interrupt/isochronous) can be scheduled through the embedded TT per frame.
  - Complete-split transaction searching

NOTE

There is no data schedule mechanism for these transactions other than the microframe pipeline. The embedded TT assumes the number of packets scheduled in a frame does not exceed the frame duration (1 ms) or else undefined behavior may result.

processor NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c62e54-821-48a8-b251-390a8a6242c1

55.11.1.5.5 Multiple transaction translators

The maximum number of embedded transaction translators currently supported is one, as indicated by HCSPARAMS[N_TT]. See Host Controller Structural Parameters (HCSPARAMS) for more information.

55.11.2 Non-zero fields in the register file

For the reserved fields and reserved addresses in the capability registers, you need to adhere to these steps:

- Write operations to all EHCI reserved fields in the operation registers must always write a value of 0.
- Read operations by the module must properly mask EHCI reserved fields.

55.11.3 SOF interrupt

The SOF interrupt is a free-running 125 µs or 1 ms interrupt for Host mode. EHCI does not specify this interrupt, but it has been added for convenience and as a potential software time base. The free running interrupt is shared with the Device mode start-of-frame interrupt. See USB Status (USBSTS) and Interrupt Enable (USBINTR) for more information.

55.11.4 Embedded design

This is an embedded USB host controller as defined by the EHCI specification; therefore, it does not implement the PCI configuration registers.

55.11.4.1 Frame adjust register

Given that the optional PCI configuration registers are not included in this implementation, there are no corresponding bit-level timing adjustments like those provided by the frame adjust register in the PCI configuration registers. Starts of microframes are timed precisely to 125 µs using the transceiver clock as a reference clock or a 60 MHz transceiver clock for 8-bit physical interfaces and FS serial interfaces.

55.11.5 Miscellaneous variations from EHCI

55.11.5.1 Programmable physical interface behavior

The modules support multiple physical interfaces that can operate in different modes when the module is configured with the software-programmable physical interface modes. The control bits for selecting the PHY operating mode are added to PORTSCn, providing a capability not defined by the EHCI specification.

55.11.5.2 Discovery

55.11.5.2.1 Port reset

The port connect methods specified by EHCI require setting PORTSC*n*[PR] to 1 for a duration of 10 ms. Due to the complexity required for supporting the attachment of devices that are not HS, a counter is present in the design that can count the 10 ms reset pulse to alleviate the requirement for you to measure this duration. Therefore, the basic connection is summarized as:

- Port change interrupt - Port connect change occurs to notify the host controller driver that a device has attached.
- You write 1 to PORTSC*n*[PR] to reset the device.
- You write 0 to PORTSC*n*[PR] after 10 ms.
  - This step, necessary in a standard EHCI design, may be omitted with this implementation. Should the EHCI host controller driver attempt to write a 0 to the reset field when a reset is in progress, the write is ignored and the reset continues until completion.
- Port change interrupt - Port enable change occurs to notify the host controller that the device is now operational and at this point the port speed is determined.

55.11.5.2.2 Port speed detection

After the port change interrupt indicates that a port is enabled, the EHCI stack must determine the port speed. Unlike the EHCI implementation, which reassigns the port owner for any device that does not connect at HS, this host controller supports direct-attach of non-HS devices. Therefore, the following differences are important regarding port speed detection:

- Port owner hand-off is not implemented. Therefore, PORTSC*n*[PO] is read-only and always reads 0.
- A 2-bit port speed indicator field has been added to PORTSCn to provide the current operating speed of the port to the host controller driver.
- A 1-bit HS indicator bit has been added to PORTSCn to signify that the port is in HS versus FS or LS.
  - This information is redundant with the 2-bit port speed indicator field above.

55.11.5.3 Port Test mode

Port Test Control mode behaves fully as described in EHCI since the release of revision 3.2.1. In earlier product revisions, the test packet mode was not EHCI compatible. An alternate host controller driver procedure is no longer necessary or supported.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 56
FlexRay Communication Controller (FlexRay)

56.1 Introduction

This module implements the FlexRay Communications System Protocol Specification, Version 2.1 Rev A.

To distinguish between the FlexRay protocol and the FlexRay module on this chip, this chapter also uses the term CC to refer to the module.

56.1.1 References

The FlexRay specifications have been developed for automotive applications. These specifications have not been developed or tested for non-automotive applications.

- *FlexRay Communications System Protocol Specification, Version 2.1 Rev A*

  See this document for all of the FlexRay-related information including the configuration parameters and the allowed parameter ranges.

- *FlexRay Communications System Electrical Physical Layer Specification, Version 3.0*

56.1.2 Overview

CC has three main components:

- CHI
- PE
- CDC

The following figure represents a block diagram of CC and its surrounding modules.

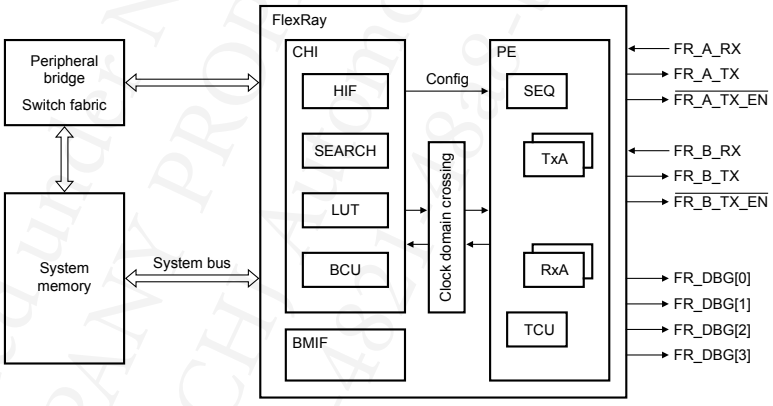


Figure 472. FlexRay block diagram

The protocol engine has two transmitter units, TxA and TxB, and two receiver units, RxA and RxB, for sending and receiving frames through the two FlexRay channels. TCU is responsible for maintaining global clock synchronization to the FlexRay network. SEQ controls the overall PE activity.

CHI provides host access to the module's configuration, control, and status registers, as well as to the MB configuration, control, and status registers.

CDC implements signal crossing between the CHI and PE clock domains, to allow for asynchronous PE and CHI clock domains.

CC stores the frame header and payload data of frames received or of frames to be transmitted in the FlexRay memory area. The application accesses the FlexRay memory area to retrieve and provide the frames for CC to process. In addition

to the frame header and payload data, CC stores the synchronization frame related tables in the FlexRay memory area for application processing.

The FlexRay memory area is located in the MCU system memory. CC has access to the FlexRay memory area via its BMIF. The host provides the start address of the FlexRay memory area within the system memory by programming System Memory Base Address High (SYMBADHR) and System Memory Base Address Low (SYMBADLR). the System Memory Base Address Register (SYMBADHR and SYMBADLR). All offsets related to the FlexRay memory area are stored in offset registers. The physical address pointer into the FlexRay memory area is calculated using the offset values and the FlexRay memory base address.

CC does not provide a memory protection scheme for the FlexRay memory area.

56.1.3 Features

- Implements a protocol that complies with the protocol specification in References
- Implements a bus driver interface that complies with the physical layer specification in References
- Single-channel support, in which FlexRay port A can connect to physical FlexRay channel A or B
- Dual-channel support
- Supports FlexRay bus data rates of:
  - 10 Mbit/s
  - 8 Mbit/s
  - 5 Mbit/s
  - 2.5 Mbit/s
- Supports 256 configurable MBs with individual:
  - Frame ID filtering
  - Channel ID filtering
  - Cycle counter filtering
- MB header, status, and payload data stored in dedicated FlexRay memory area
  - Allows for flexible and efficient MB implementation
  - Ensures consistent data access via buffer locking scheme
  - Allows an application to lock multiple buffers simultaneously
- Size of MB payload data section configurable from 0 up to 254 bytes
- Two independent MB segments with configurable size of payload data section:
  - Each segment's payload data section has a configurable size
  - Each segment can contain MBs assigned to the static segment and MBs assigned to the dynamic segment simultaneously
- Zero padding for transmit MBs in the static segment, applied when the frame payload length exceeds the size of the MB data section
- Transmit MBs configurable with state/event semantics
- MBs that can be configured as receive and transmit MBs
- Supports individual MB reconfiguration, allowing you to:
  - Disable individual MBs safely
  - Reconfigure disabled MBs
- Two independent receive FIFOs

Provided under NDA
COMPANY PROPRIETARY
Beijing CIIC Automotive Technology Co., Ltd.
c8c20e3-43b1-48a8-b251-390d3ab624c1

  - One receive FIFO per channel
  - Up to 255 entries for each FIFO
  - Global frame ID filtering, based on both value/mask filters and range filters
  - Global channel ID filtering
  - Global message ID filtering for the dynamic segment
- Four configurable slot error counters
- Four dedicated slot status indicators, used to observe slots without using receive MBs
- Measured value indicators for the clock synchronization; the internal synchronization frame ID and synchronization frame measurement tables can be copied into the FlexRay memory area
- Supports fractional MTs for clock correction
- Maskable interrupt sources provided via individual and combined interrupt lines
- One absolute timer
- One timer that can be configured as absolute or relative
- Error correction and error detection (SECDED ECC) for PE data RAM
- Error detection (SECDED ECC) for CHI LRAM

56.1.4 Modes of operation

This section describes the basic CC operational power modes.

56.1.4.1 Disabled mode

CC enters Disabled mode during hard reset or when the host writes 0 to MCR[MEN]. The host can write 0 to this field only when the module is in the POC:default config mode. You can check whether CC is in Disabled mode by reading MCR[MEN].

No communication is performed on the FlexRay bus.

All registers with the write access conditions *Any Time* and *Disabled Mode* can be accessed for writing as stated in Register descriptions.

You configure CC by accessing the configuration fields in Module Configuration (MCR) as described in Module Initialization.

56.1.4.1.1 Leave Disabled mode

CC leaves Disabled mode and enters Normal mode when you write 1 to MCR[MEN].

After you enable CC, you can disable it only during POC:default config.

56.1.4.2 Normal mode

In this mode, CC is fully functional. CC indicates that it is in Normal mode by writing 1 to MCR[MEN].

56.1.4.2.1 Enter Normal mode

CC enters this mode when you request CC to leave Disabled mode. If CC enters Normal mode by leaving Disabled mode, you must perform the protocol initialization described in Protocol Initialization to achieve full FlexRay functionality.

Depending on the values of the SCM, CHA, and CHB fields in Module Configuration (MCR), the corresponding FlexRay bus driver ports are enabled and driven.

56.2 External signal description

This section lists and describes the CC signals connected to external pins. These signals are summarized in the following table.

For the exact signal names on this chip, see the IOMUX file attached to this document. The off-chip signals FR_A_RX, FR_A_TX, and FR_A_TX_EN_b are available in each package option. The availability of the other off-chip signals depends on the package option and is specific to the chip.

Table 453. External signal properties

Name	Direction	Active	Reset	Function	Details
FR_A_RX	Input	—	—	Receive data channel A	Carries the receive data for channel A from the corresponding FlexRay bus driver
FR_A_TX	Output	—	1	Transmit data channel A	Carries the transmit data for channel A to the corresponding FlexRay bus driver
FR_A_TX_EN_b	Output	Low	1	Transmit enable channel A	Indicates to the FlexRay bus driver that CC is attempting to transmit data on channel A
FR_B_RX	Input	—	—	Receive data channel B	Carries the receive data for channel B from the corresponding FlexRay bus driver
FR_B_TX	Output	—	1	Transmit data channel B	Carries the transmit data for channel B to the corresponding FlexRay bus driver
FR_B_TX_EN_b	Output	Low	1	Transmit enable channel B	Indicates to the FlexRay bus driver that CC is attempting to transmit data on channel B
FR_DBG[0]	Output	—	0	Debug strobe 0	Provide the selected debug strobe signals For details of the debug strobe signal selection, see Strobe Signal Support.
FR_DBG[1]	Output	—	0	Debug strobe 1	
FR_DBG[2]	Output	—	0	Debug strobe 2	
FR_DBG[3]	Output	—	0	Debug strobe 3	

56.3 PE clocking

PE is clocked by an internal crystal oscillator, a crystal or CMOS compatible clock must be connected to the oscillator pins. The crystal or clock must fulfill the requirements given by the *FlexRay Communications System Protocol Specification, Version*2.1 Rev A.

56.4 Register descriptions

This section provides detailed descriptions of all registers in ascending address order, presented as 16-bit wide entities.

Table 454. Register access conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the field is not readable or not writeable.
R*	Reserved field; will not be changed. You must not write any value different from the reset value.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.

Table 455. Register field types

Convention	Description
rwm	A read/write field that hardware can modify in some fashion other than by a reset.
w1c	Write one to clear. A flag field that you can read and return to 0 by writing 1. Writing 0 has no effect.
Reset value	
0	Resets to zero.
1	Resets to one.
-	Not defined after reset and not affected by reset.

56.4.1 Register reset

After a system reset, all registers except the following reset to their reset value:

- MB Cycle Counter Filter (MBCCFR0 - MBCCFR255)
- MB Frame ID (MBFIDR0 - MBFIDR255)
- MB Index (MBIDXR0 - MBIDXR255)

The aforementioned registers are located in physical memory blocks and are therefore unaffected by reset.

For some register fields, additional reset conditions exist. The following table explains them.

Table 456. Additional register reset conditions

Condition	Description
Protocol RUN command	The field resets when you write 0101b (corresponding to the RUN command) to POCR[POCCMD].
MB disable	The field resets when you have disabled the MB. This happens when you write 1 to MBCCSR*n*[EDT] while the MB is enabled (MBCCSR*n*[EDS] = 1) and CC grants you the disabling by writing 0 to MBCCSR*n*[EDS].

56.4.2 Register write access

This section describes the write access restriction terms that apply to all registers.

56.4.2.1 Register write access restriction

The detailed register descriptions specify the write access conditions for each field. The table below describes the write access conditions. If none of the given write access conditions are fulfilled for a specific field, any write attempt to this field is ignored without any notification. The values of the fields are not changed. The condition term [A or B] indicates that the register or field can be written to if at least one of the conditions is fulfilled. The condition term [A and B] indicates that the register or field can be written to if both conditions are fulfilled.

Table 457. Register write access restrictions

Condition	Indication	Description
Any time	—	No write access restriction
Disabled mode	MCR[MEN] = 0	Write access only when CC is in Disabled mode
Normal mode	MCR[MEN] = 1	Write access only when CC is in Normal Mode
POC:config	PSR0[PROTSTATE] = 1b (representing POC:config)	Write access only when the protocol is in the POC:config state
MB_DIS	MBCCSR*n*[EDS] = 0	Write access only when related MB is disabled
MB_LCK	MBCCSR*n*[LCKS] = 1	Write access only when related MB is locked
IDL	EERICR[BSY] = 0	Write access only when ECC configuration is idle

56.4.2.2 Register write access requirements

All registers can be accessed with 8-bit, 16-bit, and 32-bit wide operations.

For some of the registers, at least a 16-bit wide write access is required to ensure correct operation. This write access requirement is described in FlexRay register descriptions. If an 8-bit wide write access is performed to any of these registers, this access is ignored without notification.

56.4.2.3 Internal register access

The following memory mapped registers are used to access multiple internal registers.

- Strobe Signal Control (STBSCR)
- Slot Status Selection (SSSR)
- Slot Status Counter Condition (SSCCR)
- Receive Shadow Buffer Configuration Data

Each of these memory mapped registers provides a SEL and a WMD field. The SEL field is used to select the internal register. The WMD field controls the write mode. If you write 0 to WMD during the write access, all fields of the internal register are updated. If you write 1 to WMD, only the SEL field is changed. All other fields of the internal register remain unchanged. This allows for reading back the values of the selected internal register in a subsequent read access.

56.4.3 PCR*n* fields

Table 458. PCR*n* fields

Name	Description	Min	Max	Unit	PCR #
coldstart_attempts	gColdstartAttempts			number	3
action_point_offset	gdActionPointOffset - 1			MT	0
cas_rx_low_max	gdCASRxLowMax - 1			gdBit	4

Table continues on the next page...

Table 458. PCR*n* fields (continued)

Name	Description	Min	Max	Unit	PCR #
dynamic_slot_idle_phase	gdDynamicSlotIdlePhase			minislot	28
minislot_action_point_offset	gdMinislotActionPointOffset - 1			MT	3
minislot_after_action_point	gdMinislot - gdMinislotActionPointOffset - 1			MT	2
static_slot_length	gdStaticSlot			MT	0
static_slot_after_action_point	gdStaticSlot - gdActionPointOffset - 1			MT	13
symbol_window_exists	gdSymbolWindow!=0	0	1	bool	9
symbol_window_after_action_point	gdSymbolWindow - gdActionPointOffset - 1			MT	6
tss_transmitter	gdTSSTransmitter			gdBit	5
wakeup_symbol_rx_idle	gdWakeupSymbolRxIdle			gdBit	5
wakeup_symbol_rx_low	gdWakeupSymbolRxLow			gdBit	3
wakeup_symbol_rx_window	gdWakeupSymbolRxWindow			gdBit	4
wakeup_symbol_tx_idle	gdWakeupSymbolTxIdle			gdBit	8
wakeup_symbol_tx_low	gdWakeupSymbolTxLow			gdBit	5
noise_listen_timeout	(gListenNoise * pdListenTimeout) - 1			µT	16/17
macro_initial_offset_a	pMacroInitialOffset[A]			MT	6
macro_initial_offset_b	pMacroInitialOffset[B]			MT	16
macro_per_cycle	gMacroPerCycle			MT	10
macro_after_first_static_slot	gMacroPerCycle - gdStaticSlot			MT	1
macro_after_offset_correction	gMacroPerCycle - gOffsetCorrectionStart			MT	28
max_without_clock_correction_fatal	gMaxWithoutClockCorrectionFatal			cyclepairs	8
max_without_clock_correction_passive	gMaxWithoutClockCorrectionPassive			cyclepairs	8
minislot_exists	gNumberOfMinislots!=0	0	1	bool	9
minislots_max	gNumberOfMinislots - 1			minislot	29
number_of_static_slots	gNumberOfStaticSlots			static slot	2
offset_correction_start	gOffsetCorrectionStart			MT	11

Table continues on the next page...

Table 458. PCR*n* fields (continued)

Name	Description	Min	Max	Unit	PCR #
payload_length_static	gPayloadLengthStatic			2 bytes	19
max_payload_length_dynamic	pPayloadLengthDynMax			2 bytes	24
first_minislot_action_point_offset	max(gdActionPointOffset, gdMinislotActionPointOffset) - 1			MT	13
allow_halt_due_to_clock	pAllowHaltDueToClock			bool	26
allow_passive_to_active	pAllowPassiveToActive			cyclepairs	12
cluster_drift_damping	pClusterDriftDamping			µT	24
comp_accepted_startup_range_a	pdAcceptedStartupRange - pDelayCompensation[A]			µT	22
comp_accepted_startup_range_b	pdAcceptedStartupRange - pDelayCompensation[B]			µT	26
listen_timeout	pdListenTimeout - 1			µT	14/15
key_slot_id	pKeySlotId			number	18
key_slot_used_for_startup	pKeySlotUsedForStartup			bool	11
key_slot_used_for_sync	pKeySlotUsedForSync			bool	11
latest_tx	gNumberOfMinislots - pLatestTx			minislot	21
sync_node_max	gSyncNodeMax			number	30
micro_initial_offset_a	pMicroInitialOffset[A]			µT	20
micro_initial_offset_b	pMicroInitialOffset[B]			µT	20
micro_per_cycle	pMicroPerCycle			µT	22/23
micro_per_cycle_min	pMicroPerCycle - pdMaxDrift			µT	24/25
micro_per_cycle_max	pMicroPerCycle + pdMaxDrift			µT	26/27
micro_per_macro_nom_half	round(pMicroPerMacroNom / 2)			µT	7
offset_correction_out	pOffsetCorrectionOut			µT	9
rate_correction_out	pRateCorrectionOut			µT	14
single_slot_enabled	pSingleSlotEnabled			bool	10
wakeup_channel	pWakeupChannel	See Table 459.			10

Table continues on the next page...

Table 458. PCR*n* fields (continued)

Name	Description	Min	Max	Unit	PCR #
wakeup_pattern	pWakeupPattern			number	18
decoding_correction_a	pDecodingCorrection + pDelayCompensation[A] + 2			µT	19
decoding_correction_b	pDecodingCorrection + pDelayCompensation[B] + 2			µT	7
key_slot_header_crc	Header CRC for key slot	0h	7FFh	number	12
extern_offset_correction	pExternOffsetCorrection			µT	29
extern_rate_correction	pExternRateCorrection			µT	21

Table 459. Wakeup channel selection

wakeup_channel	Wakeup channel
0	A
1	B

56.5 FlexRay register descriptions

CC occupies 8 KB (8192 bytes) of address space starting at the CC base address defined by the chip memory map.

Do not access these address offsets:

- Ah

because corresponding features are unavailable.

You can write to the following registers only with 16-bit writes:

- Strobe Signal Control (STBSCR)
- PE DRAM Access (PEDRAR)
- PE DRAM Data (PEDRDR)
- Sync Frame ID Rejection Filter (SFIDRFR)
- Slot Status Selection (SSSR)
- Slot Status Counter Condition (SSCCR)
- Receive Shadow Buffer Index (RSBIR)
- Receive FIFO Range Filter Configuration (RFRFCFR)
- MB Cycle Counter Filter (MBCCFR0 - MBCCFR255)
- MB Frame ID (MBFIDR0 - MBFIDR255)
- MB Index (MBIDXR0 - MBIDXR255)
- MB Data Field Offset (MBDOR0 - MBDOR259)
- LRAM ECC Error Test Register (LEETR0 - LEETR5)

56.5.1 FlexRay memory map

FR_0 base address: 402F_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Version (MVR)	16	RO	A568h
2h	Module Configuration (MCR)	16	RW	0000h
4h	System Memory Base Address High (SYMBADHR)	16	RW	0000h
6h	System Memory Base Address Low (SYMBADLR)	16	RW	0000h
8h	Strobe Signal Control (STBSCR)	16	RW	0000h
Ch	MB Data Size (MBDSR)	16	RW	0000h
Eh	MB Segment Size and Utilization (MBSSUTR)	16	RW	FFFFh
10h	PE DRAM Access (PEDRAR)	16	RW	0000h
12h	PE DRAM Data (PEDRDR)	16	RW	0000h
14h	Protocol Operation Control Register (POCR)	16	RW	0000h
16h	Global Interrupt Flag And Enable (GIFER)	16	RW	0000h
18h	Protocol Interrupt Flag 0 (PIFR0)	16	W1C	0000h
1Ah	Protocol Interrupt Flag 1 (PIFR1)	16	W1C	0000h
1Ch	Protocol Interrupt Enable 0 (PIER0)	16	RW	0000h
1Eh	Protocol Interrupt Enable 1 (PIER1)	16	RW	0000h
20h	CHI Error Flag (CHIERFR)	16	W1C	0000h
22h	MB Interrupt Vector (MBIVEC)	16	RO	0000h
24h	Channel A Status Error Counter Register (CASERCR)	16	RO	0000h
26h	Channel B Status Error Counter (CBSERCR)	16	RO	0000h
28h	Protocol Status 0 (PSR0)	16	RO	0000h
2Ah	Protocol Status 1 (PSR1)	16	W1C	0000h
2Ch	Protocol Status Register 2 (PSR2)	16	RO	0000h
2Eh	Protocol Status 3 (PSR3)	16	W1C	0000h
30h	MT Counter (MTCTR)	16	RO	0000h
32h	Cycle Counter (CYCTR)	16	RO	0000h
34h	Slot Counter Channel A (SLTCTAR)	16	RO	0000h
36h	Slot Counter Channel B (SLTCTBR)	16	RO	0000h
38h	Rate Correction Value (RTCORVR)	16	RO	0000h
3Ah	Offset Correction Value (OFCORVR)	16	RO	0000h
3Ch	Combined Interrupt Flag (CIFR)	16	RO	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3Eh	System Memory Access Timeout (SYMATOR)	16	RW	0006h
40h	Sync Frame Counter (SFCNTR)	16	RO	0000h
42h	Sync Frame Table Offset (SFTOR)	16	RW	0000h
44h	Sync Frame Table Configuration Control Status (SFTCCSR)	16	RW	0000h
46h	Sync Frame ID Rejection Filter (SFIDRFR)	16	RW	0000h
48h	Sync Frame ID Acceptance Filter Value (SFIDAFVR)	16	RW	0000h
4Ah	Sync Frame ID Acceptance Filter Mask (SFIDAFMR)	16	RW	0000h
4Ch - 56h	NMV n (NMVR0 - NMVR5)	16	RO	0000h
58h	Network Management Vector Length Register (NMVLR)	16	RW	0000h
5Ah	Timer Configuration And Control (TICCR)	16	RW	0000h
5Ch	Timer 1 Cycle Set Register (TI1CYSR)	16	RW	0000h
5Eh	Timer 1 MT Offset (TI1MTOR)	16	RW	0000h
60h	Timer 2 Configuration 0 (Absolute Timer Configuration) (TI2CR0_ABS)	16	RW	0000h
60h	Timer 2 Configuration 0 (Relative Timer Configuration) (TI2CR0_REL)	16	RW	0000h
62h	Timer 2 Configuration 1 (Absolute Timer Configuration) (TI2CR1_ABS)	16	RW	0000h
62h	Timer 2 Configuration 1 (Relative Timer Configuration) (TI2CR1_REL)	16	RW	0000h
64h	Slot Status Selection (SSSR)	16	RW	0000h
66h	Slot Status Counter Condition (SSCCR)	16	RW	0000h
68h - 76h	Slot Status (SSR0 - SSR7)	16	RO	0000h
78h - 7Eh	Slot Status Counter (SSCR0 - SSCR3)	16	RO	0000h
80h	MTS A Configuration (MTSACFR)	16	RW	0000h
82h	MTS B Configuration (MTSBCFR)	16	RW	0000h
84h	Receive Shadow Buffer Index (RSBIR)	16	RW	0000h
86h	Receive FIFO Watermark And Selection (RFWMSR)	16	RW	0000h
88h	Receive FIFO Start Index (RFSIR)	16	RW	0000h
8Ah	Receive FIFO Depth And Size (RFDSR)	16	RW	0000h
8Ch	Receive FIFO A Read Index (RFARIR)	16	RO	0000h
8Eh	Receive FIFO B Read Index (RFBRIR)	16	RO	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
90h	Receive FIFO Message ID Acceptance Filter Value (RFMIDAFVR)	16	RW	0000h
92h	Receive FIFO Message ID Acceptance Filter Mask (RFMIDAFMR)	16	RW	0000h
94h	Receive FIFO Frame ID Rejection Filter Value (RFFIDRFVR)	16	RW	0000h
96h	Receive FIFO Frame ID Rejection Filter Mask (RFFIDRFMR)	16	RW	0000h
98h	Receive FIFO Range Filter Configuration (RFRFCFR)	16	RW	0000h
9Ah	Receive FIFO Range Filter Control (RFRFCTR)	16	RW	0000h
9Ch	Last Dynamic Transmit Slot Channel A (LDTXSLAR)	16	RO	0000h
9Eh	Last Dynamic Transmit Slot Channel B (LDTXSLBR)	16	RO	0000h
A0h	Protocol Configuration 0 (PCR0)	16	RW	0000h
A2h	Protocol Configuration 1 (PCR1)	16	RW	0000h
A4h	Protocol Configuration 2 (PCR2)	16	RW	0000h
A6h	Protocol Configuration 3 (PCR3)	16	RW	0000h
A8h	Protocol Configuration 4 (PCR4)	16	RW	0000h
AAh	Protocol Configuration 5 (PCR5)	16	RW	0000h
ACh	Protocol Configuration 6 (PCR6)	16	RW	0000h
AEh	Protocol Configuration 7 (PCR7)	16	RW	0000h
B0h	Protocol Configuration 8 (PCR8)	16	RW	0000h
B2h	Protocol Configuration 9 (PCR9)	16	RW	0000h
B4h	Protocol Configuration 10 (PCR10)	16	RW	0000h
B6h	Protocol Configuration 11 (PCR11)	16	RW	0000h
B8h	Protocol Configuration 12 (PCR12)	16	RW	0000h
BAh	Protocol Configuration 13 (PCR13)	16	RW	0000h
BCh	Protocol Configuration 14 (PCR14)	16	RW	0000h
BEh	Protocol Configuration 15 (PCR15)	16	RW	0000h
C0h	Protocol Configuration 16 (PCR16)	16	RW	0000h
C2h	Protocol Configuration 17 (PCR17)	16	RW	0000h
C4h	Protocol Configuration 18 (PCR18)	16	RW	0000h
C6h	Protocol Configuration 19 (PCR19)	16	RW	0000h
C8h	Protocol Configuration 20 (PCR20)	16	RW	0000h
CAh	Protocol Configuration 21 (PCR21)	16	RW	0000h
CCh	Protocol Configuration 22 (PCR22)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
CEh	Protocol Configuration 23 (PCR23)	16	RW	0000h
D0h	Protocol Configuration 24 (PCR24)	16	RW	0000h
D2h	Protocol Configuration 25 (PCR25)	16	RW	0000h
D4h	Protocol Configuration 26 (PCR26)	16	RW	0000h
D6h	Protocol Configuration 27 (PCR27)	16	RW	0000h
D8h	Protocol Configuration 28 (PCR28)	16	RW	0000h
DAh	Protocol Configuration 29 (PCR29)	16	RW	0000h
DCh	Protocol Configuration 30 (PCR30)	16	RW	0000h
DEh	Stopwatch Count High (STPWHR)	16	R2C	0000h
E0h	Stopwatch Count Low (STPWLR)	16	R2C	0000h
E2h	Protocol Event Output Enable And Stopwatch Control (PEOER)	16	RW	0000h
E6h	Receive FIFO Start Data Offset (RFSDOR)	16	RW	0000h
E8h	Receive FIFO System Memory Base Address High (RFSYMBADHR)	16	RW	0000h
EAh	Receive FIFO System Memory Base Address Low (RFSYMBADLR)	16	RW	0000h
ECh	Receive FIFO Periodic Timer (RFPTR)	16	RW	0000h
EEh	Receive FIFO Fill Level and Pop Count (RFFLPCR)	16	RW	0000h
F0h	ECC Error Interrupt Flag And Enable (EEIFER)	16	RW	0000h
F2h	ECC Error Report And Injection Control (EERICR)	16	RW	0000h
F4h	ECC Error Report Address (EERAR)	16	RO	7000h
F6h	ECC Error Report Data (EERDR)	16	RO	0000h
F8h	ECC Error Report Code (EERCR)	16	RO	0000h
FAh	ECC Error Injection Address Register (EEIAR)	16	RW	0000h
FCh	ECC Error Injection Data (EEIDR)	16	RW	0000h
FEh	ECC Error Injection Code (EEICR)	16	RW	0000h
800h	MB Configuration Control Status (MBCCSR0)	16	RW	0000h
802h	MB Cycle Counter Filter (MBCCFR0)	16	RW	0000h
804h	MB Frame ID (MBFIDR0)	16	RW	0000h
806h	MB Index (MBIDXR0)	16	RW	0000h
808h	MB Configuration Control Status (MBCCSR1)	16	RW	0000h
80Ah	MB Cycle Counter Filter (MBCCFR1)	16	RW	0000h
80Ch	MB Frame ID (MBFIDR1)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
80Eh	MB Index (MBIDXR1)	16	RW	0000h
810h	MB Configuration Control Status (MBCCSR2)	16	RW	0000h
812h	MB Cycle Counter Filter (MBCCFR2)	16	RW	0000h
814h	MB Frame ID (MBFIDR2)	16	RW	0000h
816h	MB Index (MBIDXR2)	16	RW	0000h
818h	MB Configuration Control Status (MBCCSR3)	16	RW	0000h
81Ah	MB Cycle Counter Filter (MBCCFR3)	16	RW	0000h
81Ch	MB Frame ID (MBFIDR3)	16	RW	0000h
81Eh	MB Index (MBIDXR3)	16	RW	0000h
820h	MB Configuration Control Status (MBCCSR4)	16	RW	0000h
822h	MB Cycle Counter Filter (MBCCFR4)	16	RW	0000h
824h	MB Frame ID (MBFIDR4)	16	RW	0000h
826h	MB Index (MBIDXR4)	16	RW	0000h
828h	MB Configuration Control Status (MBCCSR5)	16	RW	0000h
82Ah	MB Cycle Counter Filter (MBCCFR5)	16	RW	0000h
82Ch	MB Frame ID (MBFIDR5)	16	RW	0000h
82Eh	MB Index (MBIDXR5)	16	RW	0000h
830h	MB Configuration Control Status (MBCCSR6)	16	RW	0000h
832h	MB Cycle Counter Filter (MBCCFR6)	16	RW	0000h
834h	MB Frame ID (MBFIDR6)	16	RW	0000h
836h	MB Index (MBIDXR6)	16	RW	0000h
838h	MB Configuration Control Status (MBCCSR7)	16	RW	0000h
83Ah	MB Cycle Counter Filter (MBCCFR7)	16	RW	0000h
83Ch	MB Frame ID (MBFIDR7)	16	RW	0000h
83Eh	MB Index (MBIDXR7)	16	RW	0000h
840h	MB Configuration Control Status (MBCCSR8)	16	RW	0000h
842h	MB Cycle Counter Filter (MBCCFR8)	16	RW	0000h
844h	MB Frame ID (MBFIDR8)	16	RW	0000h
846h	MB Index (MBIDXR8)	16	RW	0000h
848h	MB Configuration Control Status (MBCCSR9)	16	RW	0000h
84Ah	MB Cycle Counter Filter (MBCCFR9)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
84Ch	MB Frame ID (MBFIDR9)	16	RW	0000h
84Eh	MB Index (MBIDXR9)	16	RW	0000h
850h	MB Configuration Control Status (MBCCSR10)	16	RW	0000h
852h	MB Cycle Counter Filter (MBCCFR10)	16	RW	0000h
854h	MB Frame ID (MBFIDR10)	16	RW	0000h
856h	MB Index (MBIDXR10)	16	RW	0000h
858h	MB Configuration Control Status (MBCCSR11)	16	RW	0000h
85Ah	MB Cycle Counter Filter (MBCCFR11)	16	RW	0000h
85Ch	MB Frame ID (MBFIDR11)	16	RW	0000h
85Eh	MB Index (MBIDXR11)	16	RW	0000h
860h	MB Configuration Control Status (MBCCSR12)	16	RW	0000h
862h	MB Cycle Counter Filter (MBCCFR12)	16	RW	0000h
864h	MB Frame ID (MBFIDR12)	16	RW	0000h
866h	MB Index (MBIDXR12)	16	RW	0000h
868h	MB Configuration Control Status (MBCCSR13)	16	RW	0000h
86Ah	MB Cycle Counter Filter (MBCCFR13)	16	RW	0000h
86Ch	MB Frame ID (MBFIDR13)	16	RW	0000h
86Eh	MB Index (MBIDXR13)	16	RW	0000h
870h	MB Configuration Control Status (MBCCSR14)	16	RW	0000h
872h	MB Cycle Counter Filter (MBCCFR14)	16	RW	0000h
874h	MB Frame ID (MBFIDR14)	16	RW	0000h
876h	MB Index (MBIDXR14)	16	RW	0000h
878h	MB Configuration Control Status (MBCCSR15)	16	RW	0000h
87Ah	MB Cycle Counter Filter (MBCCFR15)	16	RW	0000h
87Ch	MB Frame ID (MBFIDR15)	16	RW	0000h
87Eh	MB Index (MBIDXR15)	16	RW	0000h
880h	MB Configuration Control Status (MBCCSR16)	16	RW	0000h
882h	MB Cycle Counter Filter (MBCCFR16)	16	RW	0000h
884h	MB Frame ID (MBFIDR16)	16	RW	0000h
886h	MB Index (MBIDXR16)	16	RW	0000h
888h	MB Configuration Control Status (MBCCSR17)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
88Ah	MB Cycle Counter Filter (MBCCFR17)	16	RW	0000h
88Ch	MB Frame ID (MBFIDR17)	16	RW	0000h
88Eh	MB Index (MBIDXR17)	16	RW	0000h
890h	MB Configuration Control Status (MBCCSR18)	16	RW	0000h
892h	MB Cycle Counter Filter (MBCCFR18)	16	RW	0000h
894h	MB Frame ID (MBFIDR18)	16	RW	0000h
896h	MB Index (MBIDXR18)	16	RW	0000h
898h	MB Configuration Control Status (MBCCSR19)	16	RW	0000h
89Ah	MB Cycle Counter Filter (MBCCFR19)	16	RW	0000h
89Ch	MB Frame ID (MBFIDR19)	16	RW	0000h
89Eh	MB Index (MBIDXR19)	16	RW	0000h
8A0h	MB Configuration Control Status (MBCCSR20)	16	RW	0000h
8A2h	MB Cycle Counter Filter (MBCCFR20)	16	RW	0000h
8A4h	MB Frame ID (MBFIDR20)	16	RW	0000h
8A6h	MB Index (MBIDXR20)	16	RW	0000h
8A8h	MB Configuration Control Status (MBCCSR21)	16	RW	0000h
8AAh	MB Cycle Counter Filter (MBCCFR21)	16	RW	0000h
8ACh	MB Frame ID (MBFIDR21)	16	RW	0000h
8AEh	MB Index (MBIDXR21)	16	RW	0000h
8B0h	MB Configuration Control Status (MBCCSR22)	16	RW	0000h
8B2h	MB Cycle Counter Filter (MBCCFR22)	16	RW	0000h
8B4h	MB Frame ID (MBFIDR22)	16	RW	0000h
8B6h	MB Index (MBIDXR22)	16	RW	0000h
8B8h	MB Configuration Control Status (MBCCSR23)	16	RW	0000h
8BAh	MB Cycle Counter Filter (MBCCFR23)	16	RW	0000h
8BCh	MB Frame ID (MBFIDR23)	16	RW	0000h
8BEh	MB Index (MBIDXR23)	16	RW	0000h
8C0h	MB Configuration Control Status (MBCCSR24)	16	RW	0000h
8C2h	MB Cycle Counter Filter (MBCCFR24)	16	RW	0000h
8C4h	MB Frame ID (MBFIDR24)	16	RW	0000h
8C6h	MB Index (MBIDXR24)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
8C8h	MB Configuration Control Status (MBCCSR25)	16	RW	0000h
8CAh	MB Cycle Counter Filter (MBCCFR25)	16	RW	0000h
8CCh	MB Frame ID (MBFIDR25)	16	RW	0000h
8CEh	MB Index (MBIDXR25)	16	RW	0000h
8D0h	MB Configuration Control Status (MBCCSR26)	16	RW	0000h
8D2h	MB Cycle Counter Filter (MBCCFR26)	16	RW	0000h
8D4h	MB Frame ID (MBFIDR26)	16	RW	0000h
8D6h	MB Index (MBIDXR26)	16	RW	0000h
8D8h	MB Configuration Control Status (MBCCSR27)	16	RW	0000h
8DAh	MB Cycle Counter Filter (MBCCFR27)	16	RW	0000h
8DCh	MB Frame ID (MBFIDR27)	16	RW	0000h
8DEh	MB Index (MBIDXR27)	16	RW	0000h
8E0h	MB Configuration Control Status (MBCCSR28)	16	RW	0000h
8E2h	MB Cycle Counter Filter (MBCCFR28)	16	RW	0000h
8E4h	MB Frame ID (MBFIDR28)	16	RW	0000h
8E6h	MB Index (MBIDXR28)	16	RW	0000h
8E8h	MB Configuration Control Status (MBCCSR29)	16	RW	0000h
8EAh	MB Cycle Counter Filter (MBCCFR29)	16	RW	0000h
8ECh	MB Frame ID (MBFIDR29)	16	RW	0000h
8EEh	MB Index (MBIDXR29)	16	RW	0000h
8F0h	MB Configuration Control Status (MBCCSR30)	16	RW	0000h
8F2h	MB Cycle Counter Filter (MBCCFR30)	16	RW	0000h
8F4h	MB Frame ID (MBFIDR30)	16	RW	0000h
8F6h	MB Index (MBIDXR30)	16	RW	0000h
8F8h	MB Configuration Control Status (MBCCSR31)	16	RW	0000h
8FAh	MB Cycle Counter Filter (MBCCFR31)	16	RW	0000h
8FCh	MB Frame ID (MBFIDR31)	16	RW	0000h
8FEh	MB Index (MBIDXR31)	16	RW	0000h
900h	MB Configuration Control Status (MBCCSR32)	16	RW	0000h
902h	MB Cycle Counter Filter (MBCCFR32)	16	RW	0000h
904h	MB Frame ID (MBFIDR32)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
906h	MB Index (MBIDXR32)	16	RW	0000h
908h	MB Configuration Control Status (MBCCSR33)	16	RW	0000h
90Ah	MB Cycle Counter Filter (MBCCFR33)	16	RW	0000h
90Ch	MB Frame ID (MBFIDR33)	16	RW	0000h
90Eh	MB Index (MBIDXR33)	16	RW	0000h
910h	MB Configuration Control Status (MBCCSR34)	16	RW	0000h
912h	MB Cycle Counter Filter (MBCCFR34)	16	RW	0000h
914h	MB Frame ID (MBFIDR34)	16	RW	0000h
916h	MB Index (MBIDXR34)	16	RW	0000h
918h	MB Configuration Control Status (MBCCSR35)	16	RW	0000h
91Ah	MB Cycle Counter Filter (MBCCFR35)	16	RW	0000h
91Ch	MB Frame ID (MBFIDR35)	16	RW	0000h
91Eh	MB Index (MBIDXR35)	16	RW	0000h
920h	MB Configuration Control Status (MBCCSR36)	16	RW	0000h
922h	MB Cycle Counter Filter (MBCCFR36)	16	RW	0000h
924h	MB Frame ID (MBFIDR36)	16	RW	0000h
926h	MB Index (MBIDXR36)	16	RW	0000h
928h	MB Configuration Control Status (MBCCSR37)	16	RW	0000h
92Ah	MB Cycle Counter Filter (MBCCFR37)	16	RW	0000h
92Ch	MB Frame ID (MBFIDR37)	16	RW	0000h
92Eh	MB Index (MBIDXR37)	16	RW	0000h
930h	MB Configuration Control Status (MBCCSR38)	16	RW	0000h
932h	MB Cycle Counter Filter (MBCCFR38)	16	RW	0000h
934h	MB Frame ID (MBFIDR38)	16	RW	0000h
936h	MB Index (MBIDXR38)	16	RW	0000h
938h	MB Configuration Control Status (MBCCSR39)	16	RW	0000h
93Ah	MB Cycle Counter Filter (MBCCFR39)	16	RW	0000h
93Ch	MB Frame ID (MBFIDR39)	16	RW	0000h
93Eh	MB Index (MBIDXR39)	16	RW	0000h
940h	MB Configuration Control Status (MBCCSR40)	16	RW	0000h
942h	MB Cycle Counter Filter (MBCCFR40)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
944h	MB Frame ID (MBFIDR40)	16	RW	0000h
946h	MB Index (MBIDXR40)	16	RW	0000h
948h	MB Configuration Control Status (MBCCSR41)	16	RW	0000h
94Ah	MB Cycle Counter Filter (MBCCFR41)	16	RW	0000h
94Ch	MB Frame ID (MBFIDR41)	16	RW	0000h
94Eh	MB Index (MBIDXR41)	16	RW	0000h
950h	MB Configuration Control Status (MBCCSR42)	16	RW	0000h
952h	MB Cycle Counter Filter (MBCCFR42)	16	RW	0000h
954h	MB Frame ID (MBFIDR42)	16	RW	0000h
956h	MB Index (MBIDXR42)	16	RW	0000h
958h	MB Configuration Control Status (MBCCSR43)	16	RW	0000h
95Ah	MB Cycle Counter Filter (MBCCFR43)	16	RW	0000h
95Ch	MB Frame ID (MBFIDR43)	16	RW	0000h
95Eh	MB Index (MBIDXR43)	16	RW	0000h
960h	MB Configuration Control Status (MBCCSR44)	16	RW	0000h
962h	MB Cycle Counter Filter (MBCCFR44)	16	RW	0000h
964h	MB Frame ID (MBFIDR44)	16	RW	0000h
966h	MB Index (MBIDXR44)	16	RW	0000h
968h	MB Configuration Control Status (MBCCSR45)	16	RW	0000h
96Ah	MB Cycle Counter Filter (MBCCFR45)	16	RW	0000h
96Ch	MB Frame ID (MBFIDR45)	16	RW	0000h
96Eh	MB Index (MBIDXR45)	16	RW	0000h
970h	MB Configuration Control Status (MBCCSR46)	16	RW	0000h
972h	MB Cycle Counter Filter (MBCCFR46)	16	RW	0000h
974h	MB Frame ID (MBFIDR46)	16	RW	0000h
976h	MB Index (MBIDXR46)	16	RW	0000h
978h	MB Configuration Control Status (MBCCSR47)	16	RW	0000h
97Ah	MB Cycle Counter Filter (MBCCFR47)	16	RW	0000h
97Ch	MB Frame ID (MBFIDR47)	16	RW	0000h
97Eh	MB Index (MBIDXR47)	16	RW	0000h
980h	MB Configuration Control Status (MBCCSR48)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
982h	MB Cycle Counter Filter (MBCCFR48)	16	RW	0000h
984h	MB Frame ID (MBFIDR48)	16	RW	0000h
986h	MB Index (MBIDXR48)	16	RW	0000h
988h	MB Configuration Control Status (MBCCSR49)	16	RW	0000h
98Ah	MB Cycle Counter Filter (MBCCFR49)	16	RW	0000h
98Ch	MB Frame ID (MBFIDR49)	16	RW	0000h
98Eh	MB Index (MBIDXR49)	16	RW	0000h
990h	MB Configuration Control Status (MBCCSR50)	16	RW	0000h
992h	MB Cycle Counter Filter (MBCCFR50)	16	RW	0000h
994h	MB Frame ID (MBFIDR50)	16	RW	0000h
996h	MB Index (MBIDXR50)	16	RW	0000h
998h	MB Configuration Control Status (MBCCSR51)	16	RW	0000h
99Ah	MB Cycle Counter Filter (MBCCFR51)	16	RW	0000h
99Ch	MB Frame ID (MBFIDR51)	16	RW	0000h
99Eh	MB Index (MBIDXR51)	16	RW	0000h
9A0h	MB Configuration Control Status (MBCCSR52)	16	RW	0000h
9A2h	MB Cycle Counter Filter (MBCCFR52)	16	RW	0000h
9A4h	MB Frame ID (MBFIDR52)	16	RW	0000h
9A6h	MB Index (MBIDXR52)	16	RW	0000h
9A8h	MB Configuration Control Status (MBCCSR53)	16	RW	0000h
9AAh	MB Cycle Counter Filter (MBCCFR53)	16	RW	0000h
9ACh	MB Frame ID (MBFIDR53)	16	RW	0000h
9AEh	MB Index (MBIDXR53)	16	RW	0000h
9B0h	MB Configuration Control Status (MBCCSR54)	16	RW	0000h
9B2h	MB Cycle Counter Filter (MBCCFR54)	16	RW	0000h
9B4h	MB Frame ID (MBFIDR54)	16	RW	0000h
9B6h	MB Index (MBIDXR54)	16	RW	0000h
9B8h	MB Configuration Control Status (MBCCSR55)	16	RW	0000h
9BAh	MB Cycle Counter Filter (MBCCFR55)	16	RW	0000h
9BCh	MB Frame ID (MBFIDR55)	16	RW	0000h
9BEh	MB Index (MBIDXR55)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9C0h	MB Configuration Control Status (MBCCSR56)	16	RW	0000h
9C2h	MB Cycle Counter Filter (MBCCFR56)	16	RW	0000h
9C4h	MB Frame ID (MBFIDR56)	16	RW	0000h
9C6h	MB Index (MBIDXR56)	16	RW	0000h
9C8h	MB Configuration Control Status (MBCCSR57)	16	RW	0000h
9CAh	MB Cycle Counter Filter (MBCCFR57)	16	RW	0000h
9CCh	MB Frame ID (MBFIDR57)	16	RW	0000h
9CEh	MB Index (MBIDXR57)	16	RW	0000h
9D0h	MB Configuration Control Status (MBCCSR58)	16	RW	0000h
9D2h	MB Cycle Counter Filter (MBCCFR58)	16	RW	0000h
9D4h	MB Frame ID (MBFIDR58)	16	RW	0000h
9D6h	MB Index (MBIDXR58)	16	RW	0000h
9D8h	MB Configuration Control Status (MBCCSR59)	16	RW	0000h
9DAh	MB Cycle Counter Filter (MBCCFR59)	16	RW	0000h
9DCh	MB Frame ID (MBFIDR59)	16	RW	0000h
9DEh	MB Index (MBIDXR59)	16	RW	0000h
9E0h	MB Configuration Control Status (MBCCSR60)	16	RW	0000h
9E2h	MB Cycle Counter Filter (MBCCFR60)	16	RW	0000h
9E4h	MB Frame ID (MBFIDR60)	16	RW	0000h
9E6h	MB Index (MBIDXR60)	16	RW	0000h
9E8h	MB Configuration Control Status (MBCCSR61)	16	RW	0000h
9EAh	MB Cycle Counter Filter (MBCCFR61)	16	RW	0000h
9ECh	MB Frame ID (MBFIDR61)	16	RW	0000h
9EEh	MB Index (MBIDXR61)	16	RW	0000h
9F0h	MB Configuration Control Status (MBCCSR62)	16	RW	0000h
9F2h	MB Cycle Counter Filter (MBCCFR62)	16	RW	0000h
9F4h	MB Frame ID (MBFIDR62)	16	RW	0000h
9F6h	MB Index (MBIDXR62)	16	RW	0000h
9F8h	MB Configuration Control Status (MBCCSR63)	16	RW	0000h
9FAh	MB Cycle Counter Filter (MBCCFR63)	16	RW	0000h
9FCh	MB Frame ID (MBFIDR63)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
9FEh	MB Index (MBIDXR63)	16	RW	0000h
A00h	MB Configuration Control Status (MBCCSR64)	16	RW	0000h
A02h	MB Cycle Counter Filter (MBCCFR64)	16	RW	0000h
A04h	MB Frame ID (MBFIDR64)	16	RW	0000h
A06h	MB Index (MBIDXR64)	16	RW	0000h
A08h	MB Configuration Control Status (MBCCSR65)	16	RW	0000h
A0Ah	MB Cycle Counter Filter (MBCCFR65)	16	RW	0000h
A0Ch	MB Frame ID (MBFIDR65)	16	RW	0000h
A0Eh	MB Index (MBIDXR65)	16	RW	0000h
A10h	MB Configuration Control Status (MBCCSR66)	16	RW	0000h
A12h	MB Cycle Counter Filter (MBCCFR66)	16	RW	0000h
A14h	MB Frame ID (MBFIDR66)	16	RW	0000h
A16h	MB Index (MBIDXR66)	16	RW	0000h
A18h	MB Configuration Control Status (MBCCSR67)	16	RW	0000h
A1Ah	MB Cycle Counter Filter (MBCCFR67)	16	RW	0000h
A1Ch	MB Frame ID (MBFIDR67)	16	RW	0000h
A1Eh	MB Index (MBIDXR67)	16	RW	0000h
A20h	MB Configuration Control Status (MBCCSR68)	16	RW	0000h
A22h	MB Cycle Counter Filter (MBCCFR68)	16	RW	0000h
A24h	MB Frame ID (MBFIDR68)	16	RW	0000h
A26h	MB Index (MBIDXR68)	16	RW	0000h
A28h	MB Configuration Control Status (MBCCSR69)	16	RW	0000h
A2Ah	MB Cycle Counter Filter (MBCCFR69)	16	RW	0000h
A2Ch	MB Frame ID (MBFIDR69)	16	RW	0000h
A2Eh	MB Index (MBIDXR69)	16	RW	0000h
A30h	MB Configuration Control Status (MBCCSR70)	16	RW	0000h
A32h	MB Cycle Counter Filter (MBCCFR70)	16	RW	0000h
A34h	MB Frame ID (MBFIDR70)	16	RW	0000h
A36h	MB Index (MBIDXR70)	16	RW	0000h
A38h	MB Configuration Control Status (MBCCSR71)	16	RW	0000h
A3Ah	MB Cycle Counter Filter (MBCCFR71)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A3Ch	MB Frame ID (MBFIDR71)	16	RW	0000h
A3Eh	MB Index (MBIDXR71)	16	RW	0000h
A40h	MB Configuration Control Status (MBCCSR72)	16	RW	0000h
A42h	MB Cycle Counter Filter (MBCCFR72)	16	RW	0000h
A44h	MB Frame ID (MBFIDR72)	16	RW	0000h
A46h	MB Index (MBIDXR72)	16	RW	0000h
A48h	MB Configuration Control Status (MBCCSR73)	16	RW	0000h
A4Ah	MB Cycle Counter Filter (MBCCFR73)	16	RW	0000h
A4Ch	MB Frame ID (MBFIDR73)	16	RW	0000h
A4Eh	MB Index (MBIDXR73)	16	RW	0000h
A50h	MB Configuration Control Status (MBCCSR74)	16	RW	0000h
A52h	MB Cycle Counter Filter (MBCCFR74)	16	RW	0000h
A54h	MB Frame ID (MBFIDR74)	16	RW	0000h
A56h	MB Index (MBIDXR74)	16	RW	0000h
A58h	MB Configuration Control Status (MBCCSR75)	16	RW	0000h
A5Ah	MB Cycle Counter Filter (MBCCFR75)	16	RW	0000h
A5Ch	MB Frame ID (MBFIDR75)	16	RW	0000h
A5Eh	MB Index (MBIDXR75)	16	RW	0000h
A60h	MB Configuration Control Status (MBCCSR76)	16	RW	0000h
A62h	MB Cycle Counter Filter (MBCCFR76)	16	RW	0000h
A64h	MB Frame ID (MBFIDR76)	16	RW	0000h
A66h	MB Index (MBIDXR76)	16	RW	0000h
A68h	MB Configuration Control Status (MBCCSR77)	16	RW	0000h
A6Ah	MB Cycle Counter Filter (MBCCFR77)	16	RW	0000h
A6Ch	MB Frame ID (MBFIDR77)	16	RW	0000h
A6Eh	MB Index (MBIDXR77)	16	RW	0000h
A70h	MB Configuration Control Status (MBCCSR78)	16	RW	0000h
A72h	MB Cycle Counter Filter (MBCCFR78)	16	RW	0000h
A74h	MB Frame ID (MBFIDR78)	16	RW	0000h
A76h	MB Index (MBIDXR78)	16	RW	0000h
A78h	MB Configuration Control Status (MBCCSR79)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
A7Ah	MB Cycle Counter Filter (MBCCFR79)	16	RW	0000h
A7Ch	MB Frame ID (MBFIDR79)	16	RW	0000h
A7Eh	MB Index (MBIDXR79)	16	RW	0000h
A80h	MB Configuration Control Status (MBCCSR80)	16	RW	0000h
A82h	MB Cycle Counter Filter (MBCCFR80)	16	RW	0000h
A84h	MB Frame ID (MBFIDR80)	16	RW	0000h
A86h	MB Index (MBIDXR80)	16	RW	0000h
A88h	MB Configuration Control Status (MBCCSR81)	16	RW	0000h
A8Ah	MB Cycle Counter Filter (MBCCFR81)	16	RW	0000h
A8Ch	MB Frame ID (MBFIDR81)	16	RW	0000h
A8Eh	MB Index (MBIDXR81)	16	RW	0000h
A90h	MB Configuration Control Status (MBCCSR82)	16	RW	0000h
A92h	MB Cycle Counter Filter (MBCCFR82)	16	RW	0000h
A94h	MB Frame ID (MBFIDR82)	16	RW	0000h
A96h	MB Index (MBIDXR82)	16	RW	0000h
A98h	MB Configuration Control Status (MBCCSR83)	16	RW	0000h
A9Ah	MB Cycle Counter Filter (MBCCFR83)	16	RW	0000h
A9Ch	MB Frame ID (MBFIDR83)	16	RW	0000h
A9Eh	MB Index (MBIDXR83)	16	RW	0000h
AA0h	MB Configuration Control Status (MBCCSR84)	16	RW	0000h
AA2h	MB Cycle Counter Filter (MBCCFR84)	16	RW	0000h
AA4h	MB Frame ID (MBFIDR84)	16	RW	0000h
AA6h	MB Index (MBIDXR84)	16	RW	0000h
AA8h	MB Configuration Control Status (MBCCSR85)	16	RW	0000h
AAAh	MB Cycle Counter Filter (MBCCFR85)	16	RW	0000h
AACh	MB Frame ID (MBFIDR85)	16	RW	0000h
AAEh	MB Index (MBIDXR85)	16	RW	0000h
AB0h	MB Configuration Control Status (MBCCSR86)	16	RW	0000h
AB2h	MB Cycle Counter Filter (MBCCFR86)	16	RW	0000h
AB4h	MB Frame ID (MBFIDR86)	16	RW	0000h
AB6h	MB Index (MBIDXR86)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
AB8h	MB Configuration Control Status (MBCCSR87)	16	RW	0000h
ABAh	MB Cycle Counter Filter (MBCCFR87)	16	RW	0000h
ABCh	MB Frame ID (MBFIDR87)	16	RW	0000h
ABEh	MB Index (MBIDXR87)	16	RW	0000h
AC0h	MB Configuration Control Status (MBCCSR88)	16	RW	0000h
AC2h	MB Cycle Counter Filter (MBCCFR88)	16	RW	0000h
AC4h	MB Frame ID (MBFIDR88)	16	RW	0000h
AC6h	MB Index (MBIDXR88)	16	RW	0000h
AC8h	MB Configuration Control Status (MBCCSR89)	16	RW	0000h
ACAh	MB Cycle Counter Filter (MBCCFR89)	16	RW	0000h
ACCh	MB Frame ID (MBFIDR89)	16	RW	0000h
ACEh	MB Index (MBIDXR89)	16	RW	0000h
AD0h	MB Configuration Control Status (MBCCSR90)	16	RW	0000h
AD2h	MB Cycle Counter Filter (MBCCFR90)	16	RW	0000h
AD4h	MB Frame ID (MBFIDR90)	16	RW	0000h
AD6h	MB Index (MBIDXR90)	16	RW	0000h
AD8h	MB Configuration Control Status (MBCCSR91)	16	RW	0000h
ADAh	MB Cycle Counter Filter (MBCCFR91)	16	RW	0000h
ADCh	MB Frame ID (MBFIDR91)	16	RW	0000h
ADEh	MB Index (MBIDXR91)	16	RW	0000h
AE0h	MB Configuration Control Status (MBCCSR92)	16	RW	0000h
AE2h	MB Cycle Counter Filter (MBCCFR92)	16	RW	0000h
AE4h	MB Frame ID (MBFIDR92)	16	RW	0000h
AE6h	MB Index (MBIDXR92)	16	RW	0000h
AE8h	MB Configuration Control Status (MBCCSR93)	16	RW	0000h
AEAh	MB Cycle Counter Filter (MBCCFR93)	16	RW	0000h
AECh	MB Frame ID (MBFIDR93)	16	RW	0000h
AEEh	MB Index (MBIDXR93)	16	RW	0000h
AF0h	MB Configuration Control Status (MBCCSR94)	16	RW	0000h
AF2h	MB Cycle Counter Filter (MBCCFR94)	16	RW	0000h
AF4h	MB Frame ID (MBFIDR94)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
AF6h	MB Index (MBIDXR94)	16	RW	0000h
AF8h	MB Configuration Control Status (MBCCSR95)	16	RW	0000h
AFAh	MB Cycle Counter Filter (MBCCFR95)	16	RW	0000h
AFCh	MB Frame ID (MBFIDR95)	16	RW	0000h
AFEh	MB Index (MBIDXR95)	16	RW	0000h
B00h	MB Configuration Control Status (MBCCSR96)	16	RW	0000h
B02h	MB Cycle Counter Filter (MBCCFR96)	16	RW	0000h
B04h	MB Frame ID (MBFIDR96)	16	RW	0000h
B06h	MB Index (MBIDXR96)	16	RW	0000h
B08h	MB Configuration Control Status (MBCCSR97)	16	RW	0000h
B0Ah	MB Cycle Counter Filter (MBCCFR97)	16	RW	0000h
B0Ch	MB Frame ID (MBFIDR97)	16	RW	0000h
B0Eh	MB Index (MBIDXR97)	16	RW	0000h
B10h	MB Configuration Control Status (MBCCSR98)	16	RW	0000h
B12h	MB Cycle Counter Filter (MBCCFR98)	16	RW	0000h
B14h	MB Frame ID (MBFIDR98)	16	RW	0000h
B16h	MB Index (MBIDXR98)	16	RW	0000h
B18h	MB Configuration Control Status (MBCCSR99)	16	RW	0000h
B1Ah	MB Cycle Counter Filter (MBCCFR99)	16	RW	0000h
B1Ch	MB Frame ID (MBFIDR99)	16	RW	0000h
B1Eh	MB Index (MBIDXR99)	16	RW	0000h
B20h	MB Configuration Control Status (MBCCSR100)	16	RW	0000h
B22h	MB Cycle Counter Filter (MBCCFR100)	16	RW	0000h
B24h	MB Frame ID (MBFIDR100)	16	RW	0000h
B26h	MB Index (MBIDXR100)	16	RW	0000h
B28h	MB Configuration Control Status (MBCCSR101)	16	RW	0000h
B2Ah	MB Cycle Counter Filter (MBCCFR101)	16	RW	0000h
B2Ch	MB Frame ID (MBFIDR101)	16	RW	0000h
B2Eh	MB Index (MBIDXR101)	16	RW	0000h
B30h	MB Configuration Control Status (MBCCSR102)	16	RW	0000h
B32h	MB Cycle Counter Filter (MBCCFR102)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
B34h	MB Frame ID (MBFIDR102)	16	RW	0000h
B36h	MB Index (MBIDXR102)	16	RW	0000h
B38h	MB Configuration Control Status (MBCCSR103)	16	RW	0000h
B3Ah	MB Cycle Counter Filter (MBCCFR103)	16	RW	0000h
B3Ch	MB Frame ID (MBFIDR103)	16	RW	0000h
B3Eh	MB Index (MBIDXR103)	16	RW	0000h
B40h	MB Configuration Control Status (MBCCSR104)	16	RW	0000h
B42h	MB Cycle Counter Filter (MBCCFR104)	16	RW	0000h
B44h	MB Frame ID (MBFIDR104)	16	RW	0000h
B46h	MB Index (MBIDXR104)	16	RW	0000h
B48h	MB Configuration Control Status (MBCCSR105)	16	RW	0000h
B4Ah	MB Cycle Counter Filter (MBCCFR105)	16	RW	0000h
B4Ch	MB Frame ID (MBFIDR105)	16	RW	0000h
B4Eh	MB Index (MBIDXR105)	16	RW	0000h
B50h	MB Configuration Control Status (MBCCSR106)	16	RW	0000h
B52h	MB Cycle Counter Filter (MBCCFR106)	16	RW	0000h
B54h	MB Frame ID (MBFIDR106)	16	RW	0000h
B56h	MB Index (MBIDXR106)	16	RW	0000h
B58h	MB Configuration Control Status (MBCCSR107)	16	RW	0000h
B5Ah	MB Cycle Counter Filter (MBCCFR107)	16	RW	0000h
B5Ch	MB Frame ID (MBFIDR107)	16	RW	0000h
B5Eh	MB Index (MBIDXR107)	16	RW	0000h
B60h	MB Configuration Control Status (MBCCSR108)	16	RW	0000h
B62h	MB Cycle Counter Filter (MBCCFR108)	16	RW	0000h
B64h	MB Frame ID (MBFIDR108)	16	RW	0000h
B66h	MB Index (MBIDXR108)	16	RW	0000h
B68h	MB Configuration Control Status (MBCCSR109)	16	RW	0000h
B6Ah	MB Cycle Counter Filter (MBCCFR109)	16	RW	0000h
B6Ch	MB Frame ID (MBFIDR109)	16	RW	0000h
B6Eh	MB Index (MBIDXR109)	16	RW	0000h
B70h	MB Configuration Control Status (MBCCSR110)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
B72h	MB Cycle Counter Filter (MBCCFR110)	16	RW	0000h
B74h	MB Frame ID (MBFIDR110)	16	RW	0000h
B76h	MB Index (MBIDXR110)	16	RW	0000h
B78h	MB Configuration Control Status (MBCCSR111)	16	RW	0000h
B7Ah	MB Cycle Counter Filter (MBCCFR111)	16	RW	0000h
B7Ch	MB Frame ID (MBFIDR111)	16	RW	0000h
B7Eh	MB Index (MBIDXR111)	16	RW	0000h
B80h	MB Configuration Control Status (MBCCSR112)	16	RW	0000h
B82h	MB Cycle Counter Filter (MBCCFR112)	16	RW	0000h
B84h	MB Frame ID (MBFIDR112)	16	RW	0000h
B86h	MB Index (MBIDXR112)	16	RW	0000h
B88h	MB Configuration Control Status (MBCCSR113)	16	RW	0000h
B8Ah	MB Cycle Counter Filter (MBCCFR113)	16	RW	0000h
B8Ch	MB Frame ID (MBFIDR113)	16	RW	0000h
B8Eh	MB Index (MBIDXR113)	16	RW	0000h
B90h	MB Configuration Control Status (MBCCSR114)	16	RW	0000h
B92h	MB Cycle Counter Filter (MBCCFR114)	16	RW	0000h
B94h	MB Frame ID (MBFIDR114)	16	RW	0000h
B96h	MB Index (MBIDXR114)	16	RW	0000h
B98h	MB Configuration Control Status (MBCCSR115)	16	RW	0000h
B9Ah	MB Cycle Counter Filter (MBCCFR115)	16	RW	0000h
B9Ch	MB Frame ID (MBFIDR115)	16	RW	0000h
B9Eh	MB Index (MBIDXR115)	16	RW	0000h
BA0h	MB Configuration Control Status (MBCCSR116)	16	RW	0000h
BA2h	MB Cycle Counter Filter (MBCCFR116)	16	RW	0000h
BA4h	MB Frame ID (MBFIDR116)	16	RW	0000h
BA6h	MB Index (MBIDXR116)	16	RW	0000h
BA8h	MB Configuration Control Status (MBCCSR117)	16	RW	0000h
BAAh	MB Cycle Counter Filter (MBCCFR117)	16	RW	0000h
BACh	MB Frame ID (MBFIDR117)	16	RW	0000h
BAEh	MB Index (MBIDXR117)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
BB0h	MB Configuration Control Status (MBCCSR118)	16	RW	0000h
BB2h	MB Cycle Counter Filter (MBCCFR118)	16	RW	0000h
BB4h	MB Frame ID (MBFIDR118)	16	RW	0000h
BB6h	MB Index (MBIDXR118)	16	RW	0000h
BB8h	MB Configuration Control Status (MBCCSR119)	16	RW	0000h
BBAh	MB Cycle Counter Filter (MBCCFR119)	16	RW	0000h
BBCh	MB Frame ID (MBFIDR119)	16	RW	0000h
BBEh	MB Index (MBIDXR119)	16	RW	0000h
BC0h	MB Configuration Control Status (MBCCSR120)	16	RW	0000h
BC2h	MB Cycle Counter Filter (MBCCFR120)	16	RW	0000h
BC4h	MB Frame ID (MBFIDR120)	16	RW	0000h
BC6h	MB Index (MBIDXR120)	16	RW	0000h
BC8h	MB Configuration Control Status (MBCCSR121)	16	RW	0000h
BCAh	MB Cycle Counter Filter (MBCCFR121)	16	RW	0000h
BCCh	MB Frame ID (MBFIDR121)	16	RW	0000h
BCEh	MB Index (MBIDXR121)	16	RW	0000h
BD0h	MB Configuration Control Status (MBCCSR122)	16	RW	0000h
BD2h	MB Cycle Counter Filter (MBCCFR122)	16	RW	0000h
BD4h	MB Frame ID (MBFIDR122)	16	RW	0000h
BD6h	MB Index (MBIDXR122)	16	RW	0000h
BD8h	MB Configuration Control Status (MBCCSR123)	16	RW	0000h
BDAh	MB Cycle Counter Filter (MBCCFR123)	16	RW	0000h
BDCh	MB Frame ID (MBFIDR123)	16	RW	0000h
BDEh	MB Index (MBIDXR123)	16	RW	0000h
BE0h	MB Configuration Control Status (MBCCSR124)	16	RW	0000h
BE2h	MB Cycle Counter Filter (MBCCFR124)	16	RW	0000h
BE4h	MB Frame ID (MBFIDR124)	16	RW	0000h
BE6h	MB Index (MBIDXR124)	16	RW	0000h
BE8h	MB Configuration Control Status (MBCCSR125)	16	RW	0000h
BEAh	MB Cycle Counter Filter (MBCCFR125)	16	RW	0000h
BECh	MB Frame ID (MBFIDR125)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
BEEh	MB Index (MBIDXR125)	16	RW	0000h
BF0h	MB Configuration Control Status (MBCCSR126)	16	RW	0000h
BF2h	MB Cycle Counter Filter (MBCCFR126)	16	RW	0000h
BF4h	MB Frame ID (MBFIDR126)	16	RW	0000h
BF6h	MB Index (MBIDXR126)	16	RW	0000h
BF8h	MB Configuration Control Status (MBCCSR127)	16	RW	0000h
BFAh	MB Cycle Counter Filter (MBCCFR127)	16	RW	0000h
BFCh	MB Frame ID (MBFIDR127)	16	RW	0000h
BFEh	MB Index (MBIDXR127)	16	RW	0000h
C00h	MB Configuration Control Status (MBCCSR128)	16	RW	0000h
C02h	MB Cycle Counter Filter (MBCCFR128)	16	RW	0000h
C04h	MB Frame ID (MBFIDR128)	16	RW	0000h
C06h	MB Index (MBIDXR128)	16	RW	0000h
C08h	MB Configuration Control Status (MBCCSR129)	16	RW	0000h
C0Ah	MB Cycle Counter Filter (MBCCFR129)	16	RW	0000h
C0Ch	MB Frame ID (MBFIDR129)	16	RW	0000h
C0Eh	MB Index (MBIDXR129)	16	RW	0000h
C10h	MB Configuration Control Status (MBCCSR130)	16	RW	0000h
C12h	MB Cycle Counter Filter (MBCCFR130)	16	RW	0000h
C14h	MB Frame ID (MBFIDR130)	16	RW	0000h
C16h	MB Index (MBIDXR130)	16	RW	0000h
C18h	MB Configuration Control Status (MBCCSR131)	16	RW	0000h
C1Ah	MB Cycle Counter Filter (MBCCFR131)	16	RW	0000h
C1Ch	MB Frame ID (MBFIDR131)	16	RW	0000h
C1Eh	MB Index (MBIDXR131)	16	RW	0000h
C20h	MB Configuration Control Status (MBCCSR132)	16	RW	0000h
C22h	MB Cycle Counter Filter (MBCCFR132)	16	RW	0000h
C24h	MB Frame ID (MBFIDR132)	16	RW	0000h
C26h	MB Index (MBIDXR132)	16	RW	0000h
C28h	MB Configuration Control Status (MBCCSR133)	16	RW	0000h
C2Ah	MB Cycle Counter Filter (MBCCFR133)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
C2Ch	MB Frame ID (MBFIDR133)	16	RW	0000h
C2Eh	MB Index (MBIDXR133)	16	RW	0000h
C30h	MB Configuration Control Status (MBCCSR134)	16	RW	0000h
C32h	MB Cycle Counter Filter (MBCCFR134)	16	RW	0000h
C34h	MB Frame ID (MBFIDR134)	16	RW	0000h
C36h	MB Index (MBIDXR134)	16	RW	0000h
C38h	MB Configuration Control Status (MBCCSR135)	16	RW	0000h
C3Ah	MB Cycle Counter Filter (MBCCFR135)	16	RW	0000h
C3Ch	MB Frame ID (MBFIDR135)	16	RW	0000h
C3Eh	MB Index (MBIDXR135)	16	RW	0000h
C40h	MB Configuration Control Status (MBCCSR136)	16	RW	0000h
C42h	MB Cycle Counter Filter (MBCCFR136)	16	RW	0000h
C44h	MB Frame ID (MBFIDR136)	16	RW	0000h
C46h	MB Index (MBIDXR136)	16	RW	0000h
C48h	MB Configuration Control Status (MBCCSR137)	16	RW	0000h
C4Ah	MB Cycle Counter Filter (MBCCFR137)	16	RW	0000h
C4Ch	MB Frame ID (MBFIDR137)	16	RW	0000h
C4Eh	MB Index (MBIDXR137)	16	RW	0000h
C50h	MB Configuration Control Status (MBCCSR138)	16	RW	0000h
C52h	MB Cycle Counter Filter (MBCCFR138)	16	RW	0000h
C54h	MB Frame ID (MBFIDR138)	16	RW	0000h
C56h	MB Index (MBIDXR138)	16	RW	0000h
C58h	MB Configuration Control Status (MBCCSR139)	16	RW	0000h
C5Ah	MB Cycle Counter Filter (MBCCFR139)	16	RW	0000h
C5Ch	MB Frame ID (MBFIDR139)	16	RW	0000h
C5Eh	MB Index (MBIDXR139)	16	RW	0000h
C60h	MB Configuration Control Status (MBCCSR140)	16	RW	0000h
C62h	MB Cycle Counter Filter (MBCCFR140)	16	RW	0000h
C64h	MB Frame ID (MBFIDR140)	16	RW	0000h
C66h	MB Index (MBIDXR140)	16	RW	0000h
C68h	MB Configuration Control Status (MBCCSR141)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
C6Ah	MB Cycle Counter Filter (MBCCFR141)	16	RW	0000h
C6Ch	MB Frame ID (MBFIDR141)	16	RW	0000h
C6Eh	MB Index (MBIDXR141)	16	RW	0000h
C70h	MB Configuration Control Status (MBCCSR142)	16	RW	0000h
C72h	MB Cycle Counter Filter (MBCCFR142)	16	RW	0000h
C74h	MB Frame ID (MBFIDR142)	16	RW	0000h
C76h	MB Index (MBIDXR142)	16	RW	0000h
C78h	MB Configuration Control Status (MBCCSR143)	16	RW	0000h
C7Ah	MB Cycle Counter Filter (MBCCFR143)	16	RW	0000h
C7Ch	MB Frame ID (MBFIDR143)	16	RW	0000h
C7Eh	MB Index (MBIDXR143)	16	RW	0000h
C80h	MB Configuration Control Status (MBCCSR144)	16	RW	0000h
C82h	MB Cycle Counter Filter (MBCCFR144)	16	RW	0000h
C84h	MB Frame ID (MBFIDR144)	16	RW	0000h
C86h	MB Index (MBIDXR144)	16	RW	0000h
C88h	MB Configuration Control Status (MBCCSR145)	16	RW	0000h
C8Ah	MB Cycle Counter Filter (MBCCFR145)	16	RW	0000h
C8Ch	MB Frame ID (MBFIDR145)	16	RW	0000h
C8Eh	MB Index (MBIDXR145)	16	RW	0000h
C90h	MB Configuration Control Status (MBCCSR146)	16	RW	0000h
C92h	MB Cycle Counter Filter (MBCCFR146)	16	RW	0000h
C94h	MB Frame ID (MBFIDR146)	16	RW	0000h
C96h	MB Index (MBIDXR146)	16	RW	0000h
C98h	MB Configuration Control Status (MBCCSR147)	16	RW	0000h
C9Ah	MB Cycle Counter Filter (MBCCFR147)	16	RW	0000h
C9Ch	MB Frame ID (MBFIDR147)	16	RW	0000h
C9Eh	MB Index (MBIDXR147)	16	RW	0000h
CA0h	MB Configuration Control Status (MBCCSR148)	16	RW	0000h
CA2h	MB Cycle Counter Filter (MBCCFR148)	16	RW	0000h
CA4h	MB Frame ID (MBFIDR148)	16	RW	0000h
CA6h	MB Index (MBIDXR148)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
CA8h	MB Configuration Control Status (MBCCSR149)	16	RW	0000h
CAAh	MB Cycle Counter Filter (MBCCFR149)	16	RW	0000h
CACh	MB Frame ID (MBFIDR149)	16	RW	0000h
CAEh	MB Index (MBIDXR149)	16	RW	0000h
CB0h	MB Configuration Control Status (MBCCSR150)	16	RW	0000h
CB2h	MB Cycle Counter Filter (MBCCFR150)	16	RW	0000h
CB4h	MB Frame ID (MBFIDR150)	16	RW	0000h
CB6h	MB Index (MBIDXR150)	16	RW	0000h
CB8h	MB Configuration Control Status (MBCCSR151)	16	RW	0000h
CBAh	MB Cycle Counter Filter (MBCCFR151)	16	RW	0000h
CBCh	MB Frame ID (MBFIDR151)	16	RW	0000h
CBEh	MB Index (MBIDXR151)	16	RW	0000h
CC0h	MB Configuration Control Status (MBCCSR152)	16	RW	0000h
CC2h	MB Cycle Counter Filter (MBCCFR152)	16	RW	0000h
CC4h	MB Frame ID (MBFIDR152)	16	RW	0000h
CC6h	MB Index (MBIDXR152)	16	RW	0000h
CC8h	MB Configuration Control Status (MBCCSR153)	16	RW	0000h
CCAh	MB Cycle Counter Filter (MBCCFR153)	16	RW	0000h
CCCh	MB Frame ID (MBFIDR153)	16	RW	0000h
CCEh	MB Index (MBIDXR153)	16	RW	0000h
CD0h	MB Configuration Control Status (MBCCSR154)	16	RW	0000h
CD2h	MB Cycle Counter Filter (MBCCFR154)	16	RW	0000h
CD4h	MB Frame ID (MBFIDR154)	16	RW	0000h
CD6h	MB Index (MBIDXR154)	16	RW	0000h
CD8h	MB Configuration Control Status (MBCCSR155)	16	RW	0000h
CDAh	MB Cycle Counter Filter (MBCCFR155)	16	RW	0000h
CDCh	MB Frame ID (MBFIDR155)	16	RW	0000h
CDEh	MB Index (MBIDXR155)	16	RW	0000h
CE0h	MB Configuration Control Status (MBCCSR156)	16	RW	0000h
CE2h	MB Cycle Counter Filter (MBCCFR156)	16	RW	0000h
CE4h	MB Frame ID (MBFIDR156)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
CE6h	MB Index (MBIDXR156)	16	RW	0000h
CE8h	MB Configuration Control Status (MBCCSR157)	16	RW	0000h
CEAh	MB Cycle Counter Filter (MBCCFR157)	16	RW	0000h
CECh	MB Frame ID (MBFIDR157)	16	RW	0000h
CEEh	MB Index (MBIDXR157)	16	RW	0000h
CF0h	MB Configuration Control Status (MBCCSR158)	16	RW	0000h
CF2h	MB Cycle Counter Filter (MBCCFR158)	16	RW	0000h
CF4h	MB Frame ID (MBFIDR158)	16	RW	0000h
CF6h	MB Index (MBIDXR158)	16	RW	0000h
CF8h	MB Configuration Control Status (MBCCSR159)	16	RW	0000h
CFAh	MB Cycle Counter Filter (MBCCFR159)	16	RW	0000h
CFCh	MB Frame ID (MBFIDR159)	16	RW	0000h
CFEh	MB Index (MBIDXR159)	16	RW	0000h
D00h	MB Configuration Control Status (MBCCSR160)	16	RW	0000h
D02h	MB Cycle Counter Filter (MBCCFR160)	16	RW	0000h
D04h	MB Frame ID (MBFIDR160)	16	RW	0000h
D06h	MB Index (MBIDXR160)	16	RW	0000h
D08h	MB Configuration Control Status (MBCCSR161)	16	RW	0000h
D0Ah	MB Cycle Counter Filter (MBCCFR161)	16	RW	0000h
D0Ch	MB Frame ID (MBFIDR161)	16	RW	0000h
D0Eh	MB Index (MBIDXR161)	16	RW	0000h
D10h	MB Configuration Control Status (MBCCSR162)	16	RW	0000h
D12h	MB Cycle Counter Filter (MBCCFR162)	16	RW	0000h
D14h	MB Frame ID (MBFIDR162)	16	RW	0000h
D16h	MB Index (MBIDXR162)	16	RW	0000h
D18h	MB Configuration Control Status (MBCCSR163)	16	RW	0000h
D1Ah	MB Cycle Counter Filter (MBCCFR163)	16	RW	0000h
D1Ch	MB Frame ID (MBFIDR163)	16	RW	0000h
D1Eh	MB Index (MBIDXR163)	16	RW	0000h
D20h	MB Configuration Control Status (MBCCSR164)	16	RW	0000h
D22h	MB Cycle Counter Filter (MBCCFR164)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
D24h	MB Frame ID (MBFIDR164)	16	RW	0000h
D26h	MB Index (MBIDXR164)	16	RW	0000h
D28h	MB Configuration Control Status (MBCCSR165)	16	RW	0000h
D2Ah	MB Cycle Counter Filter (MBCCFR165)	16	RW	0000h
D2Ch	MB Frame ID (MBFIDR165)	16	RW	0000h
D2Eh	MB Index (MBIDXR165)	16	RW	0000h
D30h	MB Configuration Control Status (MBCCSR166)	16	RW	0000h
D32h	MB Cycle Counter Filter (MBCCFR166)	16	RW	0000h
D34h	MB Frame ID (MBFIDR166)	16	RW	0000h
D36h	MB Index (MBIDXR166)	16	RW	0000h
D38h	MB Configuration Control Status (MBCCSR167)	16	RW	0000h
D3Ah	MB Cycle Counter Filter (MBCCFR167)	16	RW	0000h
D3Ch	MB Frame ID (MBFIDR167)	16	RW	0000h
D3Eh	MB Index (MBIDXR167)	16	RW	0000h
D40h	MB Configuration Control Status (MBCCSR168)	16	RW	0000h
D42h	MB Cycle Counter Filter (MBCCFR168)	16	RW	0000h
D44h	MB Frame ID (MBFIDR168)	16	RW	0000h
D46h	MB Index (MBIDXR168)	16	RW	0000h
D48h	MB Configuration Control Status (MBCCSR169)	16	RW	0000h
D4Ah	MB Cycle Counter Filter (MBCCFR169)	16	RW	0000h
D4Ch	MB Frame ID (MBFIDR169)	16	RW	0000h
D4Eh	MB Index (MBIDXR169)	16	RW	0000h
D50h	MB Configuration Control Status (MBCCSR170)	16	RW	0000h
D52h	MB Cycle Counter Filter (MBCCFR170)	16	RW	0000h
D54h	MB Frame ID (MBFIDR170)	16	RW	0000h
D56h	MB Index (MBIDXR170)	16	RW	0000h
D58h	MB Configuration Control Status (MBCCSR171)	16	RW	0000h
D5Ah	MB Cycle Counter Filter (MBCCFR171)	16	RW	0000h
D5Ch	MB Frame ID (MBFIDR171)	16	RW	0000h
D5Eh	MB Index (MBIDXR171)	16	RW	0000h
D60h	MB Configuration Control Status (MBCCSR172)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
D62h	MB Cycle Counter Filter (MBCCFR172)	16	RW	0000h
D64h	MB Frame ID (MBFIDR172)	16	RW	0000h
D66h	MB Index (MBIDXR172)	16	RW	0000h
D68h	MB Configuration Control Status (MBCCSR173)	16	RW	0000h
D6Ah	MB Cycle Counter Filter (MBCCFR173)	16	RW	0000h
D6Ch	MB Frame ID (MBFIDR173)	16	RW	0000h
D6Eh	MB Index (MBIDXR173)	16	RW	0000h
D70h	MB Configuration Control Status (MBCCSR174)	16	RW	0000h
D72h	MB Cycle Counter Filter (MBCCFR174)	16	RW	0000h
D74h	MB Frame ID (MBFIDR174)	16	RW	0000h
D76h	MB Index (MBIDXR174)	16	RW	0000h
D78h	MB Configuration Control Status (MBCCSR175)	16	RW	0000h
D7Ah	MB Cycle Counter Filter (MBCCFR175)	16	RW	0000h
D7Ch	MB Frame ID (MBFIDR175)	16	RW	0000h
D7Eh	MB Index (MBIDXR175)	16	RW	0000h
D80h	MB Configuration Control Status (MBCCSR176)	16	RW	0000h
D82h	MB Cycle Counter Filter (MBCCFR176)	16	RW	0000h
D84h	MB Frame ID (MBFIDR176)	16	RW	0000h
D86h	MB Index (MBIDXR176)	16	RW	0000h
D88h	MB Configuration Control Status (MBCCSR177)	16	RW	0000h
D8Ah	MB Cycle Counter Filter (MBCCFR177)	16	RW	0000h
D8Ch	MB Frame ID (MBFIDR177)	16	RW	0000h
D8Eh	MB Index (MBIDXR177)	16	RW	0000h
D90h	MB Configuration Control Status (MBCCSR178)	16	RW	0000h
D92h	MB Cycle Counter Filter (MBCCFR178)	16	RW	0000h
D94h	MB Frame ID (MBFIDR178)	16	RW	0000h
D96h	MB Index (MBIDXR178)	16	RW	0000h
D98h	MB Configuration Control Status (MBCCSR179)	16	RW	0000h
D9Ah	MB Cycle Counter Filter (MBCCFR179)	16	RW	0000h
D9Ch	MB Frame ID (MBFIDR179)	16	RW	0000h
D9Eh	MB Index (MBIDXR179)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
DA0h	MB Configuration Control Status (MBCCSR180)	16	RW	0000h
DA2h	MB Cycle Counter Filter (MBCCFR180)	16	RW	0000h
DA4h	MB Frame ID (MBFIDR180)	16	RW	0000h
DA6h	MB Index (MBIDXR180)	16	RW	0000h
DA8h	MB Configuration Control Status (MBCCSR181)	16	RW	0000h
DAAh	MB Cycle Counter Filter (MBCCFR181)	16	RW	0000h
DACh	MB Frame ID (MBFIDR181)	16	RW	0000h
DAEh	MB Index (MBIDXR181)	16	RW	0000h
DB0h	MB Configuration Control Status (MBCCSR182)	16	RW	0000h
DB2h	MB Cycle Counter Filter (MBCCFR182)	16	RW	0000h
DB4h	MB Frame ID (MBFIDR182)	16	RW	0000h
DB6h	MB Index (MBIDXR182)	16	RW	0000h
DB8h	MB Configuration Control Status (MBCCSR183)	16	RW	0000h
DBAh	MB Cycle Counter Filter (MBCCFR183)	16	RW	0000h
DBCh	MB Frame ID (MBFIDR183)	16	RW	0000h
DBEh	MB Index (MBIDXR183)	16	RW	0000h
DC0h	MB Configuration Control Status (MBCCSR184)	16	RW	0000h
DC2h	MB Cycle Counter Filter (MBCCFR184)	16	RW	0000h
DC4h	MB Frame ID (MBFIDR184)	16	RW	0000h
DC6h	MB Index (MBIDXR184)	16	RW	0000h
DC8h	MB Configuration Control Status (MBCCSR185)	16	RW	0000h
DCAh	MB Cycle Counter Filter (MBCCFR185)	16	RW	0000h
DCCh	MB Frame ID (MBFIDR185)	16	RW	0000h
DCEh	MB Index (MBIDXR185)	16	RW	0000h
DD0h	MB Configuration Control Status (MBCCSR186)	16	RW	0000h
DD2h	MB Cycle Counter Filter (MBCCFR186)	16	RW	0000h
DD4h	MB Frame ID (MBFIDR186)	16	RW	0000h
DD6h	MB Index (MBIDXR186)	16	RW	0000h
DD8h	MB Configuration Control Status (MBCCSR187)	16	RW	0000h
DDAh	MB Cycle Counter Filter (MBCCFR187)	16	RW	0000h
DDCh	MB Frame ID (MBFIDR187)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
DDEh	MB Index (MBIDXR187)	16	RW	0000h
DE0h	MB Configuration Control Status (MBCCSR188)	16	RW	0000h
DE2h	MB Cycle Counter Filter (MBCCFR188)	16	RW	0000h
DE4h	MB Frame ID (MBFIDR188)	16	RW	0000h
DE6h	MB Index (MBIDXR188)	16	RW	0000h
DE8h	MB Configuration Control Status (MBCCSR189)	16	RW	0000h
DEAh	MB Cycle Counter Filter (MBCCFR189)	16	RW	0000h
DECh	MB Frame ID (MBFIDR189)	16	RW	0000h
DEEh	MB Index (MBIDXR189)	16	RW	0000h
DF0h	MB Configuration Control Status (MBCCSR190)	16	RW	0000h
DF2h	MB Cycle Counter Filter (MBCCFR190)	16	RW	0000h
DF4h	MB Frame ID (MBFIDR190)	16	RW	0000h
DF6h	MB Index (MBIDXR190)	16	RW	0000h
DF8h	MB Configuration Control Status (MBCCSR191)	16	RW	0000h
DFAh	MB Cycle Counter Filter (MBCCFR191)	16	RW	0000h
DFCh	MB Frame ID (MBFIDR191)	16	RW	0000h
DFEh	MB Index (MBIDXR191)	16	RW	0000h
E00h	MB Configuration Control Status (MBCCSR192)	16	RW	0000h
E02h	MB Cycle Counter Filter (MBCCFR192)	16	RW	0000h
E04h	MB Frame ID (MBFIDR192)	16	RW	0000h
E06h	MB Index (MBIDXR192)	16	RW	0000h
E08h	MB Configuration Control Status (MBCCSR193)	16	RW	0000h
E0Ah	MB Cycle Counter Filter (MBCCFR193)	16	RW	0000h
E0Ch	MB Frame ID (MBFIDR193)	16	RW	0000h
E0Eh	MB Index (MBIDXR193)	16	RW	0000h
E10h	MB Configuration Control Status (MBCCSR194)	16	RW	0000h
E12h	MB Cycle Counter Filter (MBCCFR194)	16	RW	0000h
E14h	MB Frame ID (MBFIDR194)	16	RW	0000h
E16h	MB Index (MBIDXR194)	16	RW	0000h
E18h	MB Configuration Control Status (MBCCSR195)	16	RW	0000h
E1Ah	MB Cycle Counter Filter (MBCCFR195)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
E1Ch	MB Frame ID (MBFIDR195)	16	RW	0000h
E1Eh	MB Index (MBIDXR195)	16	RW	0000h
E20h	MB Configuration Control Status (MBCCSR196)	16	RW	0000h
E22h	MB Cycle Counter Filter (MBCCFR196)	16	RW	0000h
E24h	MB Frame ID (MBFIDR196)	16	RW	0000h
E26h	MB Index (MBIDXR196)	16	RW	0000h
E28h	MB Configuration Control Status (MBCCSR197)	16	RW	0000h
E2Ah	MB Cycle Counter Filter (MBCCFR197)	16	RW	0000h
E2Ch	MB Frame ID (MBFIDR197)	16	RW	0000h
E2Eh	MB Index (MBIDXR197)	16	RW	0000h
E30h	MB Configuration Control Status (MBCCSR198)	16	RW	0000h
E32h	MB Cycle Counter Filter (MBCCFR198)	16	RW	0000h
E34h	MB Frame ID (MBFIDR198)	16	RW	0000h
E36h	MB Index (MBIDXR198)	16	RW	0000h
E38h	MB Configuration Control Status (MBCCSR199)	16	RW	0000h
E3Ah	MB Cycle Counter Filter (MBCCFR199)	16	RW	0000h
E3Ch	MB Frame ID (MBFIDR199)	16	RW	0000h
E3Eh	MB Index (MBIDXR199)	16	RW	0000h
E40h	MB Configuration Control Status (MBCCSR200)	16	RW	0000h
E42h	MB Cycle Counter Filter (MBCCFR200)	16	RW	0000h
E44h	MB Frame ID (MBFIDR200)	16	RW	0000h
E46h	MB Index (MBIDXR200)	16	RW	0000h
E48h	MB Configuration Control Status (MBCCSR201)	16	RW	0000h
E4Ah	MB Cycle Counter Filter (MBCCFR201)	16	RW	0000h
E4Ch	MB Frame ID (MBFIDR201)	16	RW	0000h
E4Eh	MB Index (MBIDXR201)	16	RW	0000h
E50h	MB Configuration Control Status (MBCCSR202)	16	RW	0000h
E52h	MB Cycle Counter Filter (MBCCFR202)	16	RW	0000h
E54h	MB Frame ID (MBFIDR202)	16	RW	0000h
E56h	MB Index (MBIDXR202)	16	RW	0000h
E58h	MB Configuration Control Status (MBCCSR203)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
E5Ah	MB Cycle Counter Filter (MBCCFR203)	16	RW	0000h
E5Ch	MB Frame ID (MBFIDR203)	16	RW	0000h
E5Eh	MB Index (MBIDXR203)	16	RW	0000h
E60h	MB Configuration Control Status (MBCCSR204)	16	RW	0000h
E62h	MB Cycle Counter Filter (MBCCFR204)	16	RW	0000h
E64h	MB Frame ID (MBFIDR204)	16	RW	0000h
E66h	MB Index (MBIDXR204)	16	RW	0000h
E68h	MB Configuration Control Status (MBCCSR205)	16	RW	0000h
E6Ah	MB Cycle Counter Filter (MBCCFR205)	16	RW	0000h
E6Ch	MB Frame ID (MBFIDR205)	16	RW	0000h
E6Eh	MB Index (MBIDXR205)	16	RW	0000h
E70h	MB Configuration Control Status (MBCCSR206)	16	RW	0000h
E72h	MB Cycle Counter Filter (MBCCFR206)	16	RW	0000h
E74h	MB Frame ID (MBFIDR206)	16	RW	0000h
E76h	MB Index (MBIDXR206)	16	RW	0000h
E78h	MB Configuration Control Status (MBCCSR207)	16	RW	0000h
E7Ah	MB Cycle Counter Filter (MBCCFR207)	16	RW	0000h
E7Ch	MB Frame ID (MBFIDR207)	16	RW	0000h
E7Eh	MB Index (MBIDXR207)	16	RW	0000h
E80h	MB Configuration Control Status (MBCCSR208)	16	RW	0000h
E82h	MB Cycle Counter Filter (MBCCFR208)	16	RW	0000h
E84h	MB Frame ID (MBFIDR208)	16	RW	0000h
E86h	MB Index (MBIDXR208)	16	RW	0000h
E88h	MB Configuration Control Status (MBCCSR209)	16	RW	0000h
E8Ah	MB Cycle Counter Filter (MBCCFR209)	16	RW	0000h
E8Ch	MB Frame ID (MBFIDR209)	16	RW	0000h
E8Eh	MB Index (MBIDXR209)	16	RW	0000h
E90h	MB Configuration Control Status (MBCCSR210)	16	RW	0000h
E92h	MB Cycle Counter Filter (MBCCFR210)	16	RW	0000h
E94h	MB Frame ID (MBFIDR210)	16	RW	0000h
E96h	MB Index (MBIDXR210)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
E98h	MB Configuration Control Status (MBCCSR211)	16	RW	0000h
E9Ah	MB Cycle Counter Filter (MBCCFR211)	16	RW	0000h
E9Ch	MB Frame ID (MBFIDR211)	16	RW	0000h
E9Eh	MB Index (MBIDXR211)	16	RW	0000h
EA0h	MB Configuration Control Status (MBCCSR212)	16	RW	0000h
EA2h	MB Cycle Counter Filter (MBCCFR212)	16	RW	0000h
EA4h	MB Frame ID (MBFIDR212)	16	RW	0000h
EA6h	MB Index (MBIDXR212)	16	RW	0000h
EA8h	MB Configuration Control Status (MBCCSR213)	16	RW	0000h
EAAh	MB Cycle Counter Filter (MBCCFR213)	16	RW	0000h
EACh	MB Frame ID (MBFIDR213)	16	RW	0000h
EAEh	MB Index (MBIDXR213)	16	RW	0000h
EB0h	MB Configuration Control Status (MBCCSR214)	16	RW	0000h
EB2h	MB Cycle Counter Filter (MBCCFR214)	16	RW	0000h
EB4h	MB Frame ID (MBFIDR214)	16	RW	0000h
EB6h	MB Index (MBIDXR214)	16	RW	0000h
EB8h	MB Configuration Control Status (MBCCSR215)	16	RW	0000h
EBAh	MB Cycle Counter Filter (MBCCFR215)	16	RW	0000h
EBCh	MB Frame ID (MBFIDR215)	16	RW	0000h
EBEh	MB Index (MBIDXR215)	16	RW	0000h
EC0h	MB Configuration Control Status (MBCCSR216)	16	RW	0000h
EC2h	MB Cycle Counter Filter (MBCCFR216)	16	RW	0000h
EC4h	MB Frame ID (MBFIDR216)	16	RW	0000h
EC6h	MB Index (MBIDXR216)	16	RW	0000h
EC8h	MB Configuration Control Status (MBCCSR217)	16	RW	0000h
ECAh	MB Cycle Counter Filter (MBCCFR217)	16	RW	0000h
ECCh	MB Frame ID (MBFIDR217)	16	RW	0000h
ECEh	MB Index (MBIDXR217)	16	RW	0000h
ED0h	MB Configuration Control Status (MBCCSR218)	16	RW	0000h
ED2h	MB Cycle Counter Filter (MBCCFR218)	16	RW	0000h
ED4h	MB Frame ID (MBFIDR218)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
ED6h	MB Index (MBIDXR218)	16	RW	0000h
ED8h	MB Configuration Control Status (MBCCSR219)	16	RW	0000h
EDAh	MB Cycle Counter Filter (MBCCFR219)	16	RW	0000h
EDCh	MB Frame ID (MBFIDR219)	16	RW	0000h
EDEh	MB Index (MBIDXR219)	16	RW	0000h
EE0h	MB Configuration Control Status (MBCCSR220)	16	RW	0000h
EE2h	MB Cycle Counter Filter (MBCCFR220)	16	RW	0000h
EE4h	MB Frame ID (MBFIDR220)	16	RW	0000h
EE6h	MB Index (MBIDXR220)	16	RW	0000h
EE8h	MB Configuration Control Status (MBCCSR221)	16	RW	0000h
EEAh	MB Cycle Counter Filter (MBCCFR221)	16	RW	0000h
EECh	MB Frame ID (MBFIDR221)	16	RW	0000h
EEEh	MB Index (MBIDXR221)	16	RW	0000h
EF0h	MB Configuration Control Status (MBCCSR222)	16	RW	0000h
EF2h	MB Cycle Counter Filter (MBCCFR222)	16	RW	0000h
EF4h	MB Frame ID (MBFIDR222)	16	RW	0000h
EF6h	MB Index (MBIDXR222)	16	RW	0000h
EF8h	MB Configuration Control Status (MBCCSR223)	16	RW	0000h
EFAh	MB Cycle Counter Filter (MBCCFR223)	16	RW	0000h
EFCh	MB Frame ID (MBFIDR223)	16	RW	0000h
EFEh	MB Index (MBIDXR223)	16	RW	0000h
F00h	MB Configuration Control Status (MBCCSR224)	16	RW	0000h
F02h	MB Cycle Counter Filter (MBCCFR224)	16	RW	0000h
F04h	MB Frame ID (MBFIDR224)	16	RW	0000h
F06h	MB Index (MBIDXR224)	16	RW	0000h
F08h	MB Configuration Control Status (MBCCSR225)	16	RW	0000h
F0Ah	MB Cycle Counter Filter (MBCCFR225)	16	RW	0000h
F0Ch	MB Frame ID (MBFIDR225)	16	RW	0000h
F0Eh	MB Index (MBIDXR225)	16	RW	0000h
F10h	MB Configuration Control Status (MBCCSR226)	16	RW	0000h
F12h	MB Cycle Counter Filter (MBCCFR226)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
F14h	MB Frame ID (MBFIDR226)	16	RW	0000h
F16h	MB Index (MBIDXR226)	16	RW	0000h
F18h	MB Configuration Control Status (MBCCSR227)	16	RW	0000h
F1Ah	MB Cycle Counter Filter (MBCCFR227)	16	RW	0000h
F1Ch	MB Frame ID (MBFIDR227)	16	RW	0000h
F1Eh	MB Index (MBIDXR227)	16	RW	0000h
F20h	MB Configuration Control Status (MBCCSR228)	16	RW	0000h
F22h	MB Cycle Counter Filter (MBCCFR228)	16	RW	0000h
F24h	MB Frame ID (MBFIDR228)	16	RW	0000h
F26h	MB Index (MBIDXR228)	16	RW	0000h
F28h	MB Configuration Control Status (MBCCSR229)	16	RW	0000h
F2Ah	MB Cycle Counter Filter (MBCCFR229)	16	RW	0000h
F2Ch	MB Frame ID (MBFIDR229)	16	RW	0000h
F2Eh	MB Index (MBIDXR229)	16	RW	0000h
F30h	MB Configuration Control Status (MBCCSR230)	16	RW	0000h
F32h	MB Cycle Counter Filter (MBCCFR230)	16	RW	0000h
F34h	MB Frame ID (MBFIDR230)	16	RW	0000h
F36h	MB Index (MBIDXR230)	16	RW	0000h
F38h	MB Configuration Control Status (MBCCSR231)	16	RW	0000h
F3Ah	MB Cycle Counter Filter (MBCCFR231)	16	RW	0000h
F3Ch	MB Frame ID (MBFIDR231)	16	RW	0000h
F3Eh	MB Index (MBIDXR231)	16	RW	0000h
F40h	MB Configuration Control Status (MBCCSR232)	16	RW	0000h
F42h	MB Cycle Counter Filter (MBCCFR232)	16	RW	0000h
F44h	MB Frame ID (MBFIDR232)	16	RW	0000h
F46h	MB Index (MBIDXR232)	16	RW	0000h
F48h	MB Configuration Control Status (MBCCSR233)	16	RW	0000h
F4Ah	MB Cycle Counter Filter (MBCCFR233)	16	RW	0000h
F4Ch	MB Frame ID (MBFIDR233)	16	RW	0000h
F4Eh	MB Index (MBIDXR233)	16	RW	0000h
F50h	MB Configuration Control Status (MBCCSR234)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
F52h	MB Cycle Counter Filter (MBCCFR234)	16	RW	0000h
F54h	MB Frame ID (MBFIDR234)	16	RW	0000h
F56h	MB Index (MBIDXR234)	16	RW	0000h
F58h	MB Configuration Control Status (MBCCSR235)	16	RW	0000h
F5Ah	MB Cycle Counter Filter (MBCCFR235)	16	RW	0000h
F5Ch	MB Frame ID (MBFIDR235)	16	RW	0000h
F5Eh	MB Index (MBIDXR235)	16	RW	0000h
F60h	MB Configuration Control Status (MBCCSR236)	16	RW	0000h
F62h	MB Cycle Counter Filter (MBCCFR236)	16	RW	0000h
F64h	MB Frame ID (MBFIDR236)	16	RW	0000h
F66h	MB Index (MBIDXR236)	16	RW	0000h
F68h	MB Configuration Control Status (MBCCSR237)	16	RW	0000h
F6Ah	MB Cycle Counter Filter (MBCCFR237)	16	RW	0000h
F6Ch	MB Frame ID (MBFIDR237)	16	RW	0000h
F6Eh	MB Index (MBIDXR237)	16	RW	0000h
F70h	MB Configuration Control Status (MBCCSR238)	16	RW	0000h
F72h	MB Cycle Counter Filter (MBCCFR238)	16	RW	0000h
F74h	MB Frame ID (MBFIDR238)	16	RW	0000h
F76h	MB Index (MBIDXR238)	16	RW	0000h
F78h	MB Configuration Control Status (MBCCSR239)	16	RW	0000h
F7Ah	MB Cycle Counter Filter (MBCCFR239)	16	RW	0000h
F7Ch	MB Frame ID (MBFIDR239)	16	RW	0000h
F7Eh	MB Index (MBIDXR239)	16	RW	0000h
F80h	MB Configuration Control Status (MBCCSR240)	16	RW	0000h
F82h	MB Cycle Counter Filter (MBCCFR240)	16	RW	0000h
F84h	MB Frame ID (MBFIDR240)	16	RW	0000h
F86h	MB Index (MBIDXR240)	16	RW	0000h
F88h	MB Configuration Control Status (MBCCSR241)	16	RW	0000h
F8Ah	MB Cycle Counter Filter (MBCCFR241)	16	RW	0000h
F8Ch	MB Frame ID (MBFIDR241)	16	RW	0000h
F8Eh	MB Index (MBIDXR241)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
F90h	MB Configuration Control Status (MBCCSR242)	16	RW	0000h
F92h	MB Cycle Counter Filter (MBCCFR242)	16	RW	0000h
F94h	MB Frame ID (MBFIDR242)	16	RW	0000h
F96h	MB Index (MBIDXR242)	16	RW	0000h
F98h	MB Configuration Control Status (MBCCSR243)	16	RW	0000h
F9Ah	MB Cycle Counter Filter (MBCCFR243)	16	RW	0000h
F9Ch	MB Frame ID (MBFIDR243)	16	RW	0000h
F9Eh	MB Index (MBIDXR243)	16	RW	0000h
FA0h	MB Configuration Control Status (MBCCSR244)	16	RW	0000h
FA2h	MB Cycle Counter Filter (MBCCFR244)	16	RW	0000h
FA4h	MB Frame ID (MBFIDR244)	16	RW	0000h
FA6h	MB Index (MBIDXR244)	16	RW	0000h
FA8h	MB Configuration Control Status (MBCCSR245)	16	RW	0000h
FAAh	MB Cycle Counter Filter (MBCCFR245)	16	RW	0000h
FACh	MB Frame ID (MBFIDR245)	16	RW	0000h
FAEh	MB Index (MBIDXR245)	16	RW	0000h
FB0h	MB Configuration Control Status (MBCCSR246)	16	RW	0000h
FB2h	MB Cycle Counter Filter (MBCCFR246)	16	RW	0000h
FB4h	MB Frame ID (MBFIDR246)	16	RW	0000h
FB6h	MB Index (MBIDXR246)	16	RW	0000h
FB8h	MB Configuration Control Status (MBCCSR247)	16	RW	0000h
FBAh	MB Cycle Counter Filter (MBCCFR247)	16	RW	0000h
FBCh	MB Frame ID (MBFIDR247)	16	RW	0000h
FBEh	MB Index (MBIDXR247)	16	RW	0000h
FC0h	MB Configuration Control Status (MBCCSR248)	16	RW	0000h
FC2h	MB Cycle Counter Filter (MBCCFR248)	16	RW	0000h
FC4h	MB Frame ID (MBFIDR248)	16	RW	0000h
FC6h	MB Index (MBIDXR248)	16	RW	0000h
FC8h	MB Configuration Control Status (MBCCSR249)	16	RW	0000h
FCAh	MB Cycle Counter Filter (MBCCFR249)	16	RW	0000h
FCCh	MB Frame ID (MBFIDR249)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
FCEh	MB Index (MBIDXR249)	16	RW	0000h
FD0h	MB Configuration Control Status (MBCCSR250)	16	RW	0000h
FD2h	MB Cycle Counter Filter (MBCCFR250)	16	RW	0000h
FD4h	MB Frame ID (MBFIDR250)	16	RW	0000h
FD6h	MB Index (MBIDXR250)	16	RW	0000h
FD8h	MB Configuration Control Status (MBCCSR251)	16	RW	0000h
FDAh	MB Cycle Counter Filter (MBCCFR251)	16	RW	0000h
FDCh	MB Frame ID (MBFIDR251)	16	RW	0000h
FDEh	MB Index (MBIDXR251)	16	RW	0000h
FE0h	MB Configuration Control Status (MBCCSR252)	16	RW	0000h
FE2h	MB Cycle Counter Filter (MBCCFR252)	16	RW	0000h
FE4h	MB Frame ID (MBFIDR252)	16	RW	0000h
FE6h	MB Index (MBIDXR252)	16	RW	0000h
FE8h	MB Configuration Control Status (MBCCSR253)	16	RW	0000h
FEAh	MB Cycle Counter Filter (MBCCFR253)	16	RW	0000h
FECh	MB Frame ID (MBFIDR253)	16	RW	0000h
FEEh	MB Index (MBIDXR253)	16	RW	0000h
FF0h	MB Configuration Control Status (MBCCSR254)	16	RW	0000h
FF2h	MB Cycle Counter Filter (MBCCFR254)	16	RW	0000h
FF4h	MB Frame ID (MBFIDR254)	16	RW	0000h
FF6h	MB Index (MBIDXR254)	16	RW	0000h
FF8h	MB Configuration Control Status (MBCCSR255)	16	RW	0000h
FFAh	MB Cycle Counter Filter (MBCCFR255)	16	RW	0000h
FFCh	MB Frame ID (MBFIDR255)	16	RW	0000h
FFEh	MB Index (MBIDXR255)	16	RW	0000h
1000h - 1206h	MB Data Field Offset (MBDOR0 - MBDOR259)	16	RW	0000h
1210h - 121Ah	LRAM ECC Error Test Register (LEETR0 - LEETR5)	16	RW	0000h

Preliminary Information
COMPANY CONFIDENTIAL
Beijing CTA Automotive Technology Co.,Ltd.
c8c4025-4821-48a8-b251-39...

56.5.2 Module Version (MVR)

Offset

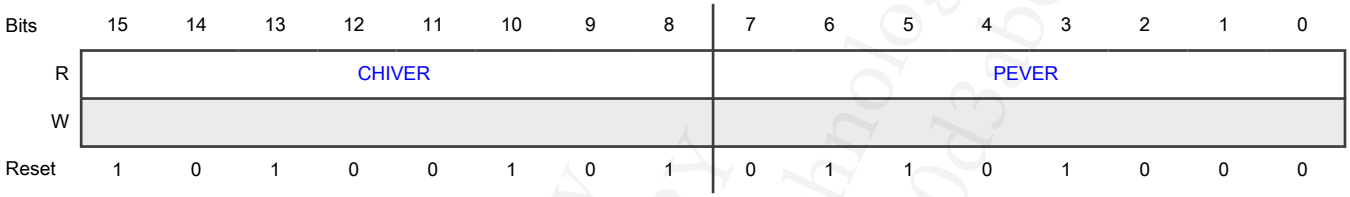
Register	Offset
MVR	0h

Function

Provides the CC version number.

The CC version number is derived from the CHI and PE version numbers.

Diagram



Fields

Field	Function
15-8 CHIVER	CHI Version Number Provides the CHI version number.
7-0 PEVER	PE Version Number Provides the PE version number.

56.5.3 Module Configuration (MCR)

Offset

Register	Offset
MCR	2h

Function

Defines the global CC configuration.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39043b624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MEN	SBFF	SCM	CHB	CHA	SFFE	ECCE	Reserved	FUM	FAM	0	Reserved	BITRATE			0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 MEN	Module Enable Indicates whether or not CC is in Disabled mode. You request CC to leave Disabled mode by writing 1 to this field. Before leaving Disabled Mode, you must configure the SCM, SBFF, CHB, CHA, TMODE, and BITRATE fields in this register. For details, see Modes of operation. If CC is enabled, it can only be disabled during mode: POC:default config. 0b - Write: only during POC:default config, CC disable Read: CC disabled 1b - Write: enable CC Read: CC enabled
14 SBFF	System Bus Failure Freeze Controls the CC behavior in case of a system bus failure. 0b - Continue normal operation 1b - Transition to freeze mode
13 SCM	Single Channel Device Mode Defines the CC channel device mode as described in Channel Device Modes. 0b - Dual channel 1b - Single channel
12 CHB	Channel B Enable Protocol related parameter: pChannels. The semantic of these control bits depends on the channel device mode controlled by the SCM bit and is given in the following table.

Table 460. FlexRay channel selection

SCM	CHB	CHA	Description
Dual Channel Device Modes			
0	0	0	ports FR_A_RX, FR_A_TX, and FR_A_TX_EN_b not driven by CC ports FR_B_RX, FR_B_TX, and FR_A_TX_EN_b not driven by CC
	0	1	ports FR_A_RX, FR_A_TX, and FR_A_TX_EN_b driven by CC - connected to FlexRay channel A ports FR_B_RX, FR_B_TX, and FR_A_TX_EN_b not driven by CC

Provided under NDA only
COMPANY PROPRIETARY
Beijing eSOL Automotive Technology Co., Ltd.
e5461e5-4881-48a2-b251-39c03a624c1

Table 460. FlexRay channel selection (continued)

Field	Function
	Table 460. FlexRay channel selection (continued)

SCM	CHB	CHA	Description
	1	0	reserved
	1	1	ports FR_A_RX, FR_A_TX, and FR_A_TX_EN_b driven by CC - connected to FlexRay channel A ports FR_B_RX, FR_B_TX, and FR_A_TX_EN_b driven by CC - connected to FlexRay channel B
Single Channel Device Mode			
1	0	0	ports FR_A_RX, FR_A_TX, and FR_A_TX_EN_b not driven by CC ports FR_B_RX, FR_B_TX, and FR_A_TX_EN_b not driven by CC
	0	1	ports FR_A_RX, FR_A_TX, and FR_A_TX_EN_b driven by CC - connected to FlexRay channel A ports FR_B_RX, FR_B_TX, and FR_A_TX_EN_b not driven by CC
	1	0	ports FR_A_RX, FR_A_TX, and FR_A_TX_EN_b driven by CC - connected to FlexRay channel B ports FR_B_RX, FR_B_TX, and FR_A_TX_EN_b not driven by CC
	1	1	reserved

11 CHA	Channel A Enable Protocol related parameter: pChannels The semantic of these control bits depends on the channel device mode controlled by the SCM bit and is given in the FlexRay channel selection table as seen above.
10 SFFE	Synchronization Frame Filter Enable Enables and disables the filtering for received synchronization frames. For details, see Sync Frame Filtering. 0b - Disabled 1b - Enabled
9 ECCE	ECC Functionality Enable Enables and disables the ECC memory error detection functionality (injection, detection, reporting, and response). For details, see Memory Content Fault Detection. 0b - Disabled 1b - Enabled
8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	Do not write any value different from the reset value.
7 FUM	FIFO Update Mode Determines whether FIFOA or FIFOB are updated when you write 1 to the FAFAIF or FAFBIF interrupt flags in Global Interrupt Flag And Enable (GIFER). See FIFO Update. 0b - Updated 1b - Not updated
6 FAM	FIFO Address Mode Controls the location of the system memory base address for the FIFOs. See FIFO Configuration. When FAM = 0, the address is located in System Memory Base Address High (SYMBADHR). When FAM = 1, the address is located in Receive FIFO System Memory Base Address High (RFSYMBADHR). 0b - SYMBADHR 1b - RFSYMBADHR
5 —	Reserved
4 —	Reserved
3-1 BITRATE	FlexRay Bus Bit Rate Defines the FlexRay bus bit rate. 000b - 10.0 Mbit/s 001b - 5.0 Mbit/s 010b - 2.5 Mbit/s 011b - 8.0 Mbit/s All other values are reserved.
0 —	Reserved

56.5.4 System Memory Base Address High (SYMBADHR)

Offset

Register	Offset
SYMBADHR	4h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c4625-4821-48a8-b251-390d3b624c1

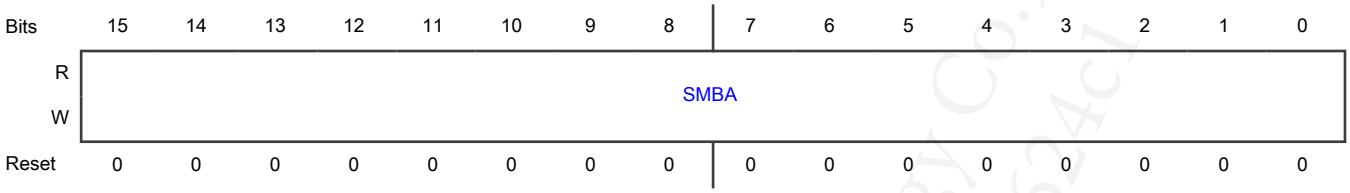
Function

Contains the most significant bits of the base address.

This register and System Memory Base Address Low (SYMBADLR) define the base address of the FlexRay memory area within the system memory. BMIF uses the base address to calculate the physical memory address for system memory accesses.

You must set the system memory base address before enabling the CC.

Diagram



Fields

Field	Function
15-0 SMBA	System Memory Base Address High Contains the value of the system memory base address for the individual MBs and sync frame table. This is the value of the system memory base address for the receive FIFO if MCR[FAM] = 1. It is defined as a byte address.

56.5.5 System Memory Base Address Low (SYMBADLR)

Offset

Register	Offset
SYMBADLR	6h

Function

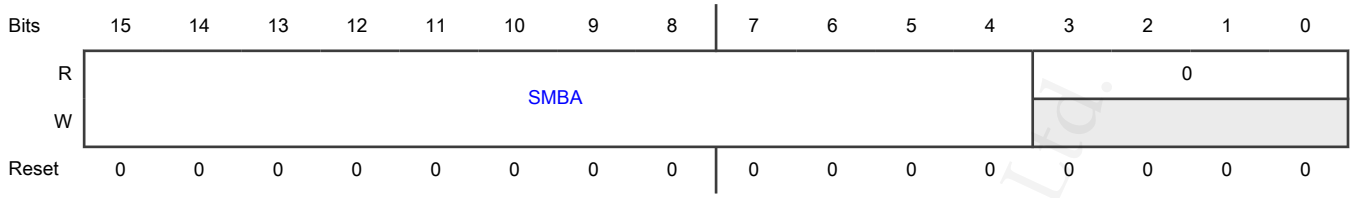
Contains the least significant bits of the base address.

This register and System Memory Base Address High (SYMBADHR) define the base address of the FlexRay memory area within the system memory. BMIF uses the base address to calculate the physical memory address for system memory accesses.

You must set the system memory base address before enabling the CC.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CE Automotive Technology Co., Ltd
c8c462e5-4e01-48a8-b251-90d3ab624c1

Diagram



Fields

Field	Function
15-4 SMBA	System Memory Base Address Low Contains the value of the system memory base address for the individual MBs and sync frame table. This is the value of the system memory base address for the receive FIFO if MCR[FAM] = 1. It is defined as a byte address.
3-0 —	Reserved

56.5.6 Strobe Signal Control (STBSCR)

Offset

Register	Offset
STBSCR	8h

Function

Assigns the individual protocol timing related strobe signals given in Table 461 to the external strobe ports.

Each strobe signal can be assigned to at most one strobe port. Each write access to registers overwrites the previously written ENB and STBPSEL values for the signal indicated by SEL. If more than one strobe signal is assigned to a strobe port, the current values of the strobe signals are combined with a binary OR and presented at the strobe port. If no strobe signal is assigned to a strobe port, the strobe port carries logic 0. For more information, see Strobe Signal Support.

In single channel device mode, channel B related strobe signals are undefined and must not be assigned to the strobe ports.

Table 461. Strobe signal mapping

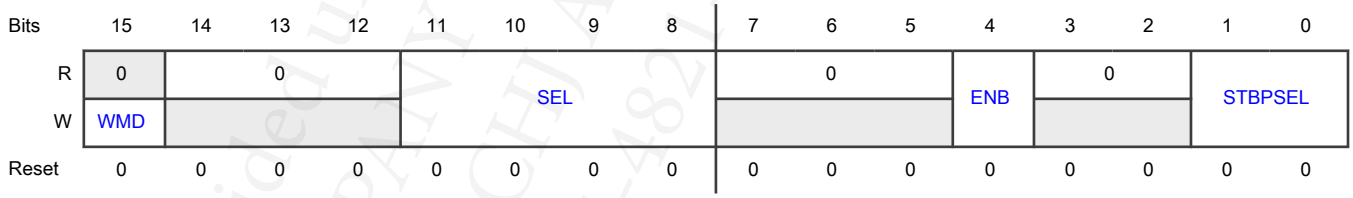
SEL		Description	Channel	Type	Offset (in PE clock cycles)	Reference
Dec	Hex					
0	0	Arm	-	Value	+1	MT start
1	1	MT	-	Value	+1	MT start
2	2	Cycle start	-	Pulse	0	MT start

Table continues on the next page...

Table 461. Strobe signal mapping (continued)

SEL		Description	Channel	Type	Offset (in PE clock cycles)	Reference
Dec	Hex					
3	3	Minislot start	-	Pulse	0	MT start
4	4	Slot start	A	Pulse	0	MT start
5	5		B			
6	6	Receive data after glitch filtering	A	Value	+4	FR_A_RX
7	7		B			FR_B_RX
8	8	Channel idle indicator	A	Level	+5	FR_A_RX
9	9		B			FR_B_RX
10	A	Syntax error detected	A	Pulse	+4	FR_A_RX
11	B		B			FR_B_RX
12	C	Content error detected	A	Level	+4	FR_A_RX
13	D		B			FR_B_RX
14	E	Receive FIFO almost-full interrupt signals	A	Value	n.a.	RX FIFO A almost full interrupt
15	F		B			RX FIFO B almost full interrupt

Diagram



Fields

Field	Function
15 WMD	Write Mode Defines the write mode of this register.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Write to all fields in this register on a write access 1b - Write only to the SEL field on write access
14-12 —	Reserved
11-8 SEL	Strobe Signal Select Selects one of the strobe signals given in Table 461 to be enabled or disabled and assigned to one of the four strobe ports given in that table. Write: Disabled Mode
7-5 —	Reserved
4 ENB	Strobe Signal Enable Enables and disables the strobe signal selected by STBSSEL. 0b - Disabled and not assigned to any strobe port 1b - Enabled and assigned to the strobe port selected by STBPSEL
3-2 —	Reserved
1-0 STBPSEL	Strobe Port Select Selects the strobe port that the strobe signal selected by SEL is assigned to. All strobe signals that are enabled and assigned to the same strobe port are combined with a binary OR operation. 00b - FR_DBG[0] 01b - FR_DBG[1] 10b - FR_DBG[2] 11b - FR_DBG[3]

56.5.7 MB Data Size (MBDSR)

Offset

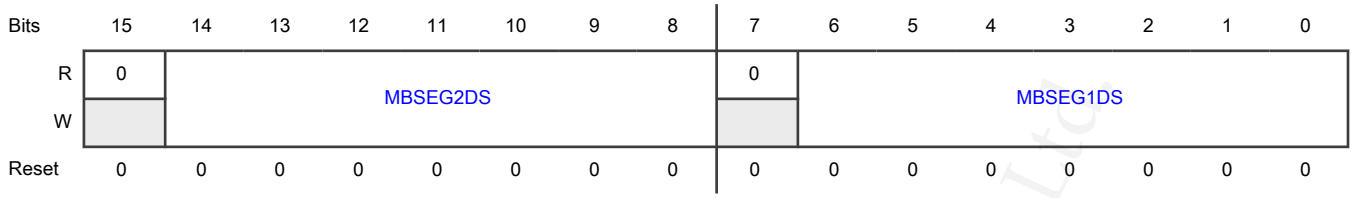
Register	Offset
MBDSR	Ch

Function

Defines the size of the MB data section for the two MB segments in a number of two-byte entities.

CC provides two independent segments for the individual MBs. All individual MBs within a segment must have the same size for the MB data section. This size can be different for the two MB segments.

Diagram



Fields

Field	Function
15 —	Reserved
14-8 MBSEG2DS	MB Segment 2 Data Size Defines the size of the MB data section in two-byte entities for MBs within the second MB segment.
7 —	Reserved
6-0 MBSEG1DS	MB Segment 1 Data Size Defines the size of the MB data section in two-byte entities for MBs within the first MB segment.

56.5.8 MB Segment Size and Utilization (MBSSUTR)

Offset

Register	Offset
MBSSUTR	Eh

Function

Defines:

- The last individual MB that belongs to the first MB segment
- The number of the last used individual MB

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d2ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	LAST_MB_SEG1								LAST_MB_UTIL							
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
15-8 LAST_MB_SEG1	Last MB In Segment 1 Defines the MB number of the last individual MB that is assigned to the first MB segment. The individual MBs in the first segment correspond to the following MB control registers with n ≤ LAST_MB_SEG1: • MB Configuration Control Status (MBCCSR0 - MBCCSR255) • MB Cycle Counter Filter (MBCCFR0 - MBCCFR255) • MB Frame ID (MBFIDR0 - MBFIDR255) • MB Index (MBIDXR0 - MBIDXR255) The first MB segment contains: • At least one individual MB • LAST_MB_SEG1+1 individual MBs The individual MBs in the second MB segment correspond to the following MB control registers with LAST_MB_SEG1 < n < 256: • MB Configuration Control Status (MBCCSR0 - MBCCSR255) • MB Cycle Counter Filter (MBCCFR0 - MBCCFR255) • MB Frame ID (MBFIDR0 - MBFIDR255) • MB Index (MBIDXR0 - MBIDXR255) If LAST_MB_SEG1 = 255, all individual MBs belong to the first MB segment and the second MB segment is empty.
7-0 LAST_MB_UTIL	Last MB Utilized Defines the MB number of the last utilized individual MB. The MB search engine examines all individual MBs with an MB number n ≤ LAST_MB_UTIL. If LAST_MB_UTIL = LAST_MB_SEG1, all individual MBs belong to the first MB segment and the second MB segment is empty.

Preliminary Information COMPANY CONFIDENTIAL Beijing CJJ Automotive Technology Co., Ltd. c8c462e5-821f-48a3-b251-390d3a1624d1

56.5.9 PE DRAM Access (PEDRAR)

Offset

Register	Offset
PEDRAR	10h

Function

Triggers write and read operations on the PE data memory (PE DRAM). These operations are used for memory error injection and memory error observation.

Each write access to this register initiates a read or write operation on PE DRAM. CC writes 0 to PEDRAR[DAD] after the write access, and writes 1 to that field if the PE DRAM access is finished.

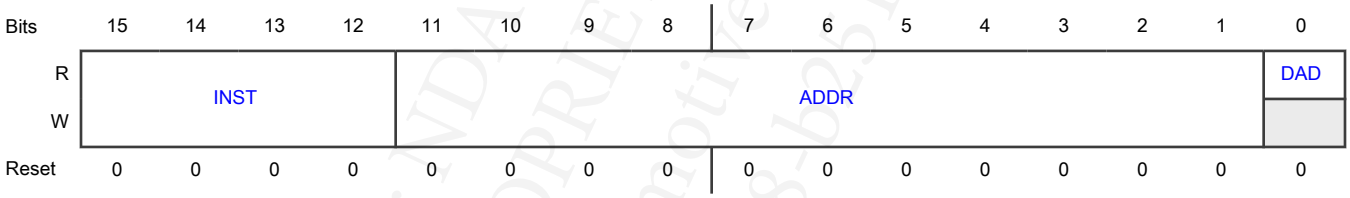
In case of a PE DRAM write access, the data provided in PEDRDR is:

- Written in PE DRAM
- Read back from PE DRAM
- Stored in PEDRDR

In case of an PE DRAM read access, the requested data is read from PE DRAM and stored in PEDRDR.

For a detailed description, see Memory Content Fault Detection.

Diagram



Fields

Field	Function
15-12 INST	PE DRAM Access Instruction Defines the operation to be executed on PE DRAM. other reserved 0011b - Write PEDRDR[DATA] to PE DRAM address ADDR (16 bit) 0101b - Read data from PE DRAM address ADDR (16 bit) into PEDRDR[DATA] All other values are reserved.
11-1 ADDR	PE DRAM Access Address Defines the address in PE DRAM to be written to or read from.
0 DAD	PE DRAM Access Done

Table continues on the next page...

Table continued from the previous page...

Field	Function
	CC writes 0 to this field when you have written to PERDR. CC writes 1 to this field when the PE DRAM access has finished. 0b - Access running 1b - Access done

56.5.10 PE DRAM Data (PEDRDR)

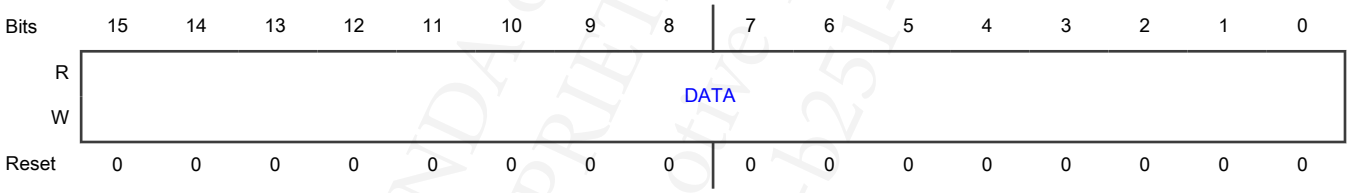
Offset

Register	Offset
PEDRDR	12h

Function

Provides the PE DRAM data that you want to write or read when you initiate a write access to PE DRAM Access (PEDRAR).

Diagram



Fields

Field	Function
15-0 DATA	Data To Or From PE DRAM Contains the PE DRAM that you want to write or read when you initiate a write access to PE DRAM Access (PEDRAR).

56.5.11 Protocol Operation Control Register (POCR)

Offset

Register	Offset
POCR	14h

Function

Issues commands for protocol control and external clock correction.

Protocol control commands are issued by writing to the POCCMD field. For more information on protocol control commands, see Protocol Control Command Execution.

External clock correction commands are issued by writing to the EOC_AP and ERC_AP fields. For more information on external clock correction, see External Clock Synchronization.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0			EOC_AP		ERC_AP		BSY_WMC	0			POCCMD			
W	WME															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 WME	Write Mode External Correction Controls the write mode of the EOC_AP and ERC_AP fields. 0b - Write to EOC_AP and ERC_AP fields on register write. 1b - No write to EOC_AP and ERC_AP fields on register write.
14-12 —	Reserved
11-10 EOC_AP	External Offset Correction Application Triggers the application of the external offset correction value defined in Protocol Configuration 29 (PCR29). 00b - Do not apply external offset correction value 01b - Reserved 10b - Subtract external offset correction value 11b - Add external offset correction value
9-8 ERC_AP	External Rate Correction Application Triggers the application of the external rate correction value defined in Protocol Configuration 21 (PCR21). 00b - Do not apply external rate correction value 01b - Reserved 10b - Subtract external rate correction value 11b - Add external rate correction value
7 BSY_WMC	Protocol Control Command Busy Or Command Write Serves two functions depending on whether you are reading or writing this field. When you read this field, it indicates the acceptance of the protocol control command that you issued via the POCCMD field. CC writes 1 to BSY_WMC when you issue the command, and writes 0 to BSY_WMC when PE accepts the command. When you issue a protocol control command while BSY_WMC = 1, CC:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Ignores this command • Writes 1 to CHIERFR[PCMI_EF] ets the protocol command ignored error flag PCMI_EF in the CHI Error Flag Register (CHIERFR) • Does not change the value of the POCCMD field When you write this field, it indicates the acceptance of the protocol control command that you issued via the POCCMD field. CC writes 1 to BSY_WMC when you issue the command, and writes 0 to BSY_WMC when PE accepts the command. When you issue a protocol control command while BSY_WMC = 1, CC: • Ignores this command • Writes 1 to CHIERFR[PCMI_EF] ets the protocol command ignored error flag PCMI_EF in the CHI Error Flag Register (CHIERFR) • Does not change the value of the POCCMD field 0b - Read operation: Command write idle; command accepted and ready to receive new protocol command. Write operation: Write to POCCMD field on register write. 1b - Read operation: Command write busy; command not yet accepted; not ready to receive new protocol command. Write operation: Do not write to POCCMD field on register write.
6-4 —	Reserved
3-0 POCCMD	Protocol Control Command Write to this field to issue a protocol control command to the PE. CC sends the protocol command to the PE immediately. While the transfer is running, BSY_WMC is 1. 0000b - ALLOW_COLDSTART. Immediately activate capability of node to cold start cluster. 0001b - ALL_SLOTS. Delayed transition to the all slots transmission mode. (Delayed means on completion of current communication cycle.) 0010b - CONFIG. Immediately transition to the POC:config state. 0011b - FREEZE. Immediately transition to the POC:halt state. 0100b - READY, CONFIG_COMPLETE. Immediately transition to the POC:ready state. 0101b - RUN. Immediately transition to the POC:startup start state. 0110b - DEFAULT_CONFIG. Immediately transition to the POC:default config state. 0111b - HALT. Delayed transition to the POC:halt state 1000b - WAKEUP. Immediately initiate the wakeup procedure. All other values are reserved.

56.5.12 Global Interrupt Flag And Enable (GIFER)

Offset

Register	Offset
GIFER	16h

Function

Provides the means to control some of the interrupt request lines and provides the corresponding interrupt flags.

The interrupt flags MIF, PRIF, CHIF, RBIF, and TBIF are the outcome of a binary OR of the related individual interrupt flags and interrupt enables. The generation scheme for these flags is depicted in Module Interrupt. For more details on interrupt generation, see Interrupt Support. These flags become 0 automatically when you write 0 to all of the corresponding interrupt flags or interrupt enables in the related interrupt flag and enable registers.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MIF	PRIF	CHIF	WUPIF	FAFBIF	FAFAIF	RBIF	TBIF	MIE	PRIE	CHIE	WUPIE	FAFBIE	FAFAIE	RBIE	TBIE
W				W1C	W1C	W1C										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 MIF	Module Interrupt Flag CC writes 1 to this field if at least one of the other interrupt flags in this register and the related interrupt enable field are 1. 0b - No interrupt flag and related interrupt enable bit are set. 1b - At least one of the other interrupt flags in this register and the related interrupt field are 1.
14 PRIF	Protocol Interrupt Flag CC writes 1 to this field if at least one of the individual flags in Protocol Interrupt Flag 0 (PIFR0) and Protocol Interrupt Flag 1 (PIFR1), and the related interrupt enable field, are 1. 0b - No individual protocol interrupt flag and related interrupt enable field are 1. 1b - At least one of the individual protocol interrupt flags and the related interrupt enable field are 1.
13 CHIF	CHI Interrupt Flag CC writes 1 to this field if at least one of the error flags in CHI Error Flag (CHIERFR) and GIFER[CHIE] are 1. 0b - All CHI error flags are 0 or the CHI error interrupt is disabled. 1b - At least one CHI error flag and the CHI error interrupt enable are 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 WUPIF	Wakeup Interrupt Flag Indicates whether CC has received a wakeup symbol on the FlexRay bus. You can determine which channel received the wakeup symbol by reading the related WUB and WUA fields in Protocol Status 3 (PSR3). 0b - Not received 1b - Received
11 FAFBIF	Receive FIFO Channel B Almost Full Interrupt Flag CC causes this field to become 1 when one of the following events occurs: • The current number of FIFO B entries is equal to or greater than the watermark defined by RFWMSR[WM] and CC writes a received message into FIFO A. • The current number of FIFO B entries is at least 1 and the periodic timer, as defined by Receive FIFO Periodic Timer (RFPTR), expires. 0b - No such event 1b - FIFO B almost-full event has occurred
10 FAFAIF	Receive FIFO Channel A Almost Full Interrupt Flag CC causes this field to become 1 when one of the following events occurs: • The current number of FIFO A entries is equal to or greater than the watermark defined by RFWMSR[WM] and CC writes a received message into FIFO A. • The current number of FIFO B entries is at least 1 and the periodic timer, as defined by Receive FIFO Periodic Timer (RFPTR), expires. 0b - No such event 1b - FIFO A almost-full event has occurred
9 RBIF	Receive MB Interrupt Flag CC writes 1 to this field if for at least one of the individual MBs has the following field values in MB Configuration Control Status (MBCCSR0 - MBCCSR255): • MTD = 0 • MBIF = 1 • MBIE = 1 You cannot write 0 to this field directly. Instead, CC writes 0 to this field after either of the following occurs: • You write 0 to all of the individual MBIF fields of the individual transmit MBs. • You write 0 to the related MBIE field. 0b - None of the individual transmit MBs has MBIF = MBIE = 1. 1b - At least one individual transmit MB has MBIF = MBIE = 1.
8	Transmit MB Interrupt Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
TBIF	CC writes 1 to this field if for at least one of the individual MBs all of the following fields in MB Configuration Control Status (MBCCSR0 - MBCCSR255) are 1: • MTD • MBIF • MBIE You cannot write 0 to this field directly. Instead, CC writes 0 to this field after either of the following occurs: • You write 0 to all of the individual MBIF fields of the individual transmit MBs. • You write 0 to the related MBIE field. 0b - None of the individual transmit MBs has MBIF = MBIE = 1. 1b - At least one individual transmit MB has MBIF = MBIE = 1. In Functional Test Mode, writing 1 toggles the interrupt flags.
7 MIE	Module Interrupt Enable Enables or disables the Module Interrupt line when MIF = 1. 0b - Disable 1b - Enable
6 PRIE	Protocol Interrupt Enable Enables or disables the Protocol Interrupt line when PRIF = 1. 0b - Disable 1b - Enable
5 CHIE	CHI Interrupt Enable Enables or disables the CHI Interrupt line when CHIF = 1. 0b - Disable 1b - Enable
4 WUPIE	Wakeup Interrupt Enable Enables or disables the Wakeup Interrupt line when WUPIF = 1. 0b - Disable 1b - Enable
3 FAFBIE	Receive FIFO Channel B Almost Full Interrupt Enable Enables or disables the RX FIFO B Almost Full Interrupt line when FAFBIF = 1. 0b - Disable 1b - Enable
2	Receive FIFO Channel A Almost Full Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
FAFAIE	Enables or disables the RX FIFO A Almost Full Interrupt line when FAFAIF = 1. 0b - Disable 1b - Enable
1 RBIE	Receive MB Interrupt Enable Enables or disables the Receive MB Interrupt line when RBIF = 1. 0b - Disable 1b - Enable
0 TBIE	Transmit MB Interrupt Enable Enables or disables the Transmit MB Interrupt line when TBIF = 1. 0b - Disable 1b - Enable

56.5.13 Protocol Interrupt Flag 0 (PIFR0)

Offset

Register	Offset
PIFR0	18h

Function

Holds one set of the protocol-related individual interrupt flags.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FATL_IF	INTL_IF	ILCF_IF	CSA_IF	MRC_IF	MOC_IF	CCL_IF	MXS_IF	MTX_IF	LTXB_IF	LTXA_IF	TBVB_IF	TBVA_IF	TI2_IF	TI1_IF	CYS_IF
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 FATL_IF	Fatal Protocol Error Interrupt Flag Becomes 1 when PE has detected a fatal protocol error. In this case, PE enters the POC:halt state immediately. The fatal protocol errors are:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• pLatestTx violation, as described in the MAC process of the FlexRay protocol • Transmission across slot boundary violation, as described in the FSP process of the FlexRay protocol 0b - No such event 1b - Fatal protocol error detected
14 INTL_IF	Internal Protocol Error Interrupt Flag Becomes 1 when PE engine has detected an internal protocol error. In this case, PE enters the POC:halt state immediately. An internal protocol error occurs when PE has not finished a calculation and a new calculation is requested. This can be caused by a hardware error. 0b - No such event 1b - Internal protocol error detected
13 ILCF_IF	Illegal Protocol Configuration Interrupt Flag Becomes 1 when PE has detected an illegal protocol configuration parameter setting. In this case, PE enters the POC:halt state immediately. PE checks the listen_timeout field in Protocol Configuration 14 (PCR14) and Protocol Configuration 15 (PCR15) when the application sent the CONFIG_COMPLETE command via Protocol Operation Control Register (POCR). If listen_timeout = 0, the protocol configuration setting is considered illegal. 0b - No such event 1b - Illegal protocol configuration detected
12 CSA_IF	Cold Start Abort Interrupt Flag Becomes 1 when the configured number of allowed cold start attempts is reached and none of these attempts was successful. The number of allowed cold start attempts is configured by PCR3[coldstart_attempts]. 0b - No such event 1b - Cold start aborted and no more coldstart attempts allowed
11 MRC_IF	Missing Rate Correction Interrupt Flag Becomes 1 when an insufficient number of measurements is available for rate correction at the end of the communication cycle. 0b - No such event 1b - Insufficient number of measurements for rate correction detected
10 MOC_IF	Missing Offset Correction Interrupt Flag Becomes 1 when an insufficient number of measurements is available for offset correction. This is related to the MISSING_TERM event in the CSP process for offset correction in the FlexRay protocol. 0b - No such event 1b - Insufficient number of measurements for offset correction detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 CCL_IF	Clock Correction Limit Reached Interrupt Flag Becomes 1 when the internal calculated offset or rate calculation values have reached or exceeded configured thresholds as specified by PCR9[offset_correction_out] and PCR14[rate_correction_out]. 0b - No such event 1b - Offset or rate correction limit reached
8 MXS_IF	Max Sync Frames Detected Interrupt Flag Becomes 1 when the number of synchronization frames detected in the current communication cycle exceeds the value of PCR30[sync_node_max]. **NOTE** Only synchronization frames that have passed the synchronization frame acceptance and rejection filters are taken into account. 0b - No such event 1b - More than node_sync_max sync frames detected
7 MTX_IF	Media Access Test Symbol Received Interrupt Flag Becomes 1 when MTS was received on channel A or channel B. 0b - No such event 1b - MTS symbol received
6 LTXB_IF	pLatestTx Violation on Channel B Interrupt Flag Becomes 1 when the frame transmission on channel B in the dynamic segment exceeds the dynamic segment boundary. This is related to the pLatestTx violation, as described in the MAC process of the FlexRay protocol. 0b - No such event 1b - pLatestTx violation occurred on channel B
5 LTXA_IF	pLatestTx Violation on Channel A Interrupt Flag Becomes 1 when the frame transmission on channel A in the dynamic segment exceeds the dynamic segment boundary. This is related to the pLatestTx violation as described in the MAC process of the FlexRay protocol. 0b - No such event 1b - pLatestTx violation occurred on channel A
4 TBVB_IF	Transmission Across Boundary On Channel B Interrupt Flag Becomes 1 when the frame transmission on channel B crosses the slot boundary. This is related to the transmission across slot boundary violation as described in the FSP process of the FlexRay protocol. 0b - No such event 1b - Transmission across boundary violation occurred on channel B

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 TBVA_IF	Transmission Across Boundary On Channel A Interrupt Flag Becomes 1 when the frame transmission on channel A crosses the slot boundary. This is related to the transmission across slot boundary violation as described in the FSP process of the FlexRay protocol. 0b - No such event 1b - Transmission across boundary violation occurred on channel A
2 TI2_IF	Timer 2 Expired Interrupt Flag Becomes 1 whenever timer 2 expires. 0b - No such event 1b - Timer 2 has reached its time limit
1 TI1_IF	Timer 1 Expired Interrupt Flag Becomes 1 whenever timer 1 expires. 0b - No such event 1b - Timer 1 has reached its time limit
0 CYS_IF	Cycle Start Interrupt Flag Becomes 1 when a communication cycle starts. 0b - No such event 1b - Communication cycle started

56.5.14 Protocol Interrupt Flag 1 (PIFR1)

Offset

Register	Offset
PIFR1	1Ah

Function

Holds one set of the protocol-related individual interrupt flags.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EMC_IF	IPC_IF	PECF_IF	PSC_IF	SSI3_IF	SSI2_IF	SSI1_IF	SSI0_IF	0		EVT_IF	ODT_IF	0			
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C			W1C	W1C				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 EMC_IF	Error Mode Changed Interrupt Flag Becomes 1 when CC changes the value of PSR0[ERRMODE]. 0b - No such event 1b - ERRMODE field changed
14 IPC_IF	Illegal Protocol Control Command Interrupt Flag Becomes 1 when PE tries to execute a protocol control command that POCR[POCCMD] issued and that PE detected that this protocol control command is not allowed in the current protocol state. In this case the command is not executed. For more details, see Protocol Control Command Execution. 0b - No such event 1b - Illegal protocol control command detected
13 PECF_IF	PE Communication Failure Interrupt Flag Becomes 1 if CC detects a communication failure between PE and CHI. 0b - No such event 1b - Failure detected
12 PSC_IF	Protocol State Changed Interrupt Flag Becomes 1 when the protocol state in PSR0[PROTSTATE] has changed. 0b - No such event 1b - Protocol state changed
11-8 SSIn_IF	Slot Status Counter Incremented Interrupt Flag n Becomes 1 when the SLOTSTATUSCNT field in the corresponding Slot Status Counter (SSCR0 - SSCR3) increments. 0b - No such event 1b - The corresponding slot status counter has incremented
7-6 —	Reserved
5	Even Cycle Table Written Interrupt Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
EVT_IF	Becomes 1 if CC wrote the sync frame measurement or ID tables into the FlexRay memory area for the even cycle. 0b - No such event 1b - Sync frame measurement table written
4 ODT_IF	Odd Cycle Table Written Interrupt Flag Becomes 1 if CC wrote the sync frame measurement or ID tables into the FlexRay memory area for the odd cycle. 0b - No such event 1b - Sync frame measurement table written
3-0 —	Reserved

56.5.15 Protocol Interrupt Enable 0 (PIER0)

Offset

Register	Offset
PIER0	1Ch

Function

Defines whether the individual interrupt flags defined in Protocol Interrupt Flag 0 (PIFR0) can generate a protocol interrupt request.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	FATL_IE	INTL_IE	ILCF_IE	CSA_IE	MRC_IE	MOC_IE	CCL_IE	MXS_IE	MTX_IE	LTXB_IE	LTXA_IE	TBVB_IE	TBVA_IE	TI2_IE	TI1_IE	CYS_IE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 FATL_IE	Fatal Protocol Error Interrupt Enable Enables and disables FATL_IF interrupt request generation. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 INTL_IE	Internal Protocol Error Interrupt Enable Enables and disables INTL_IF interrupt request generation. 0b - Disabled 1b - Enabled
13 ILCF_IE	Illegal Protocol Configuration Interrupt Enable Enables and disables ILCF_IF interrupt request generation. 0b - Disabled 1b - Enabled
12 CSA_IE	Cold Start Abort Interrupt Enable Enables and disables CSA_IF interrupt request generation. 0b - Disabled 1b - Enabled
11 MRC_IE	Missing Rate Correction Interrupt Enable Enables and disables MRC_IF interrupt request generation. 0b - Disabled 1b - Enabled
10 MOC_IE	Missing Offset Correction Interrupt Enable Enables and disables MOC_IF interrupt request generation. 0b - Disabled 1b - Enabled
9 CCL_IE	Clock Correction Limit Reached Interrupt Enable Enables and disables CCL_IF interrupt request generation. 0b - Disabled 1b - Enabled
8 MXS_IE	Max Sync Frames Detected Interrupt Enable Enables and disables MXS_IF interrupt request generation. 0b - Disabled 1b - Enabled
7 MTX_IE	Media Access Test Symbol Received Interrupt Enable Enables and disables MTX_IF interrupt request generation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
6 LTXB_IE	pLatestTx Violation on Channel B Interrupt Enable Enables and disables LTXB_IF interrupt request generation. 0b - Disabled 1b - Enabled
5 LTXA_IE	pLatestTx Violation On Channel A Interrupt Enable Enables and disables LTXA_IF interrupt request generation. 0b - Disabled 1b - Enabled
4 TBVB_IE	Transmission Across Boundary On Channel B Interrupt Enable Enables and disables TBVB_IF interrupt request generation. 0b - Disabled 1b - Enabled
3 TBVA_IE	Transmission Across Boundary On Channel A Interrupt Enable Enables and disables TBVA_IF interrupt request generation. 0b - Disabled 1b - Enabled
2 TI2_IE	Timer 2 Expired Interrupt Enable Enables and disables TI1_IF interrupt request generation. 0b - Disabled 1b - Enabled
1 TI1_IE	Timer 1 Expired Interrupt Enable Enables and disables TI1_IF interrupt request generation. 0b - Disabled 1b - Enabled
0 CYS_IE	Cycle Start Interrupt Enable Enables and disables CYC_IF interrupt request generation. 0b - Disabled 1b - Enabled

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co., Ltd.
c8c402e5-4821-4a8-b251-390d3ab624c1

56.5.16 Protocol Interrupt Enable 1 (PIER1)

Offset

Register	Offset
PIER1	1Eh

Function

Defines whether the individual interrupt flags defined in Protocol Interrupt Flag 1 (PIFR1) can generate a protocol interrupt request.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EMC_IE	IPC_IE	PECF_IE	PSC_IE	SSI3_IE	SSI2_IE	SSI1_IE	SSI0_IE	0		EVT_IE	ODT_IE	0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 EMC_IE	Error Mode Changed Interrupt Enable Enables and disables EMC_IF interrupt request generation. 0b - Disabled 1b - Enabled
14 IPC_IE	Illegal Protocol Control Command Interrupt Enable Enables and disables IPC_IF interrupt request generation. 0b - Disabled 1b - Enabled
13 PECF_IE	PE Communication Failure Interrupt Enable Enables and disables PECF_IF interrupt request generation. 0b - Disabled 1b - Enabled
12 PSC_IE	Protocol State Changed Interrupt Enable Enables and disables PSC_IF interrupt request generation. 0b - Disabled 1b - Enabled
11-8	Slot Status Counter Incremented Interrupt Enable Enables and disables SSI[3:0]_IF interrupt request generation.

Table continues on the next page...

Table continued from the previous page...

Field	Function
SSIn_IE	0b - Disabled 1b - Enabled
7-6 —	Reserved
5 EVT_IE	Even Cycle Table Written Interrupt Enable Enables and disables EVT_IF interrupt request generation. 0b - Disabled 1b - Enabled
4 ODT_IE	Odd Cycle Table Written Interrupt Enable Enables and disables ODT_IF interrupt request generation. 0b - Disabled 1b - Enabled
3-0 —	Reserved

56.5.17 CHI Error Flag (CHIERFR)

Offset

Register	Offset
CHIERFR	20h

Function

Holds the CHI-related error flags. The interrupt generation for each of these error flags is controlled by the CHI interrupt enable bit CHIE in the Global Interrupt Flag and Enable Register (GIFER).

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4861-48a8-b251-390d3ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FRLB_EF	FRLA_EF	PCMI_EF	FOVB_EF	FOVA_EF	MBS_EF	MBU_EF	LCK_EF	0	SBCF_EF	FID_EF	DPL_EF	SPL_EF	NML_EF	NMF_EF	ILSA_EF
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C		W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 FRLB_EF	Frame Lost Channel B Error Flag Becomes 1 if a complete frame is received on channel B but could not be stored in the selected individual MB because the application has currently locked this MB. In this case, the frame and the related slot status information are lost. 0b - No such event 1b - Frame lost on channel B detected
14 FRLA_EF	Frame Lost Channel A Error Flag Becomes 1 if a complete frame is received on channel A but could not be stored in the selected individual MB because the application has currently locked this MB. In this case, the frame and the related slot status information are lost. 0b - No such error 1b - Frame lost on channel A detected
13 PCMI_EF	Protocol Command Ignored Error Flag Becomes 1 if the application has issued a POC command by writing to POCR[POCCMD] while the BSY flag is equal to 1. In this case CC ignores and discards the command. 0b - No such error 1b - POC command is ignored
12 FOVB_EF	Receive FIFO Overrun Channel B Error Flag Becomes 1 when an overrun of the FIFO for channel B occurrs. The error occurs if a semantically valid frame is received on channel B and matches the all criteria to be appended to the FIFO for channel B but the FIFO is full. In this case, the received frame and its related slot status information is lost. 0b - No such error 1b - FIFO overrun on channel B is detected
11 FOVA_EF	Receive FIFO Overrun Channel A Error Flag Becomes 1 when an overrun of the FIFO for channel A occurs. The error occurs if a semantically valid frame is received on channel A and matches all the criteria to be appended to the FIFO for channel A but the FIFO is full. In this case, the received frame and its related slot status information is lost. 0b - No such error 1b - FIFO overrun on channel B is detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 MBS_EF	MB Search Error Flag Becomes 1 if at least one of the following events occurs: • The MB search engine is still running while the next search must be started because of the FlexRay protocol timing • A MB index greater than 259 is detected in the MBIDX field of MB Index (MBIDXR0 - MBIDXR255) of a found MB or in RSBIR[RSBIDX]. See MB search error for details. 0b - No such event 1b - Search engine is active while search start appears or illegal MB index is detected
9 MBU_EF	MB Usage Error Flag Becomes 1 if the application writes to a MB control field that is beyond the number of used MBs programmed in MB Segment Size and Utilization (MBSSUTR). If the application writes to MB Configuration Control Status (MBCCSR0 - MBCCSR255) with n > LAST_MB_UTIL, CC ignores the write attempt and causes MBU_EF to become 1. 0b - No such event 1b - Unused MB enabled
8 LCK_EF	Lock Error Flag Becomes 1 if the application tries to lock a MB that CC already locked because of internal operations. In that case, CC does not grant the lock to the application. The application must issue the lock request again. 0b - No such error 1b - Lock error detected
7 —	Reserved
6 SBCF_EF	System Bus Communication Failure Error Flag Becomes 1 if a system bus access is not finished within the required amount of time (see System Bus Access Timeout). 0b - No such event 1b - System bus access not finished in time
5 FID_EF	Frame ID Error Flag Becomes 1 if the frame ID stored in the MB header area differs from the frame ID stored in the MB control register. 0b - No such error occurred 1b - Frame ID error occurred
4	Dynamic Payload Length Error Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
DPL_EF	Becomes 1 if the payload length written into the MB header field of a transmit MB assigned to the dynamic segment is greater than the maximum payload length for the dynamic segment as it is configured in PCR24[max_payload_length_dynamic]. 0b - No such error occurred 1b - Dynamic payload length error occurred
3 SPL_EF	Static Payload Length Error Flag Becomes 1 if the payload length written into the MB header field of a transmit MB assigned to the static segment is different from the payload length for the static segment as it is configured in PCR19[payload_length_static]. 0b - No such error occurred 1b - Static payload length error occurred
2 NML_EF	Network Management Length Error Flag Becomes 1 if the payload length written into the header structure of a receive MB assigned to the static segment is less than the configured NMV length as configured in Network Management Vector Length Register (NMVLR). In this case, the received NMV part is used to update NMV. 0b - No such error occurred 1b - Network management length error occurred
1 NMF_EF	Network Management Frame Error Flag Becomes 1 if a received message in the static segment with a Preamble Indicator flag PP asserted has its Null Frame indicator flag NF asserted as well. In this case, NMV n (NMVR0 - NMVR5) are not updated. 0b - No such error occurred 1b - Network management frame error occurred
0 ILSA_EF	Illegal System Bus Address Error Flag Becomes 1 if the external system bus subsystem has detected an access to an illegal system bus address from CC (see System Bus Illegal Address Access). 0b - No such event 1b - Illegal system bus address accessed

56.5.18 MB Interrupt Vector (MBIVEC)

Offset

Register	Offset
MBIVEC	22h

Function

Indicates the lowest numbered receive MB and the lowest numbered transmit MB that have their MBIF = MBIE = 1. This means that MBs with lower MB numbers have higher priority.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TBIVEC								RBIVEC							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8 TBIVEC	Transmit Buffer Interrupt Vector Provides the MB number of the lowest numbered enabled transmit MB that has its MBIF = MBIE = 1. If no receive MB exists with MBIF = MBIE = 1, the value of this field is 0.
7-0 RBIVEC	Receive Buffer Interrupt Vector Provides the MB number of the lowest numbered receive MB that has its MBIF = MBIE = 1. If no receive MB exists with MBIF = MBIE = 1, the value of this field is 0.

56.5.19 Channel A Status Error Counter Register (CASERCR)

Offset

Register	Offset
CASERCR	24h

Function

Provides the channel status error counter for channel A.

PE generates a slot status vector for:

- Each static slot
- Each dynamic slot
- The symbol window
- NIT

The slot status vector contains the four protocol-related error indicator bits:

- vSS!SyntaxError
- vSS!ContentError
- vSS!BViolation
- vSS!TxConflict

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-3903ab624c1

CC increments the status error counter by 1 if, for a slot or segment, at least one error indicator bit is 1. The counter wraps around after it reaches the maximum value. For more information on slot status monitoring, see Slot Status Monitoring.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHAERSCNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 CHAERSCNT	Channel A Status Error Counter Provides the current channel status error count. The counter value is updated within the first MT of the following slot or segment.

56.5.20 Channel B Status Error Counter (CBSERCR)

Offset

Register	Offset
CBSERCR	26h

Function

Provides the channel status error counter for channel B.

PE generates a slot status vector for:

- Each static slot
- Each dynamic slot
- The symbol window
- NIT

The slot status vector contains the four protocol-related error indicator bits:

- vSS!SyntaxError
- vSS!ContentError
- vSS!BViolation
- vSS!TxConflict

CC increments the status error counter by 1 if, for a slot or segment, at least one error indicator bit is 1. The counter wraps around after it reaches the maximum value. For more information on slot status monitoring, see Slot Status Monitoring.

Licensed under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c62e5-4821-48a8-b251-3903ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHBERSCNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 CHBERSCNT	Channel B Status Error Counter Provides the current channel status error count. The counter value is updated within the first MT of the following slot or segment.

56.5.21 Protocol Status 0 (PSR0)

Offset

Register	Offset
PSR0	28h

Function

Provides information about the current protocol status.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRMODE		SLOTMODE		0	PROTSTATE			STARTUPSTATE				0	WAKEUPSTATUS		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 ERRMODE	Error Mode This field indicates the protocol error mode. Protocol related variable: vPOC!ErrorMode 00b - ACTIVE 01b - PASSIVE

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c42e5-4821-48a8-b251-390d3a0624c1

Table continued from the previous page...

Field	Function
	10b - COMM_HALT 11b - Reserved
13-12 SLOTMODE	Slot Mode Indicates the protocol slot mode. Protocol related variable: vPOC!SlotMode 00b - SINGLE 01b - ALL_PENDING 10b - ALL 11b - Reserved
11 —	Reserved
10-8 PROTSTATE	Protocol State Indicates the state of the protocol. Protocol related variable: vPOC!State 000b - POC:default config 001b - POC:config 010b - POC:wakeup 011b - POC:ready 100b - POC:normal passive 101b - POC:normal active 110b - POC:halt 111b - POC:startup
7-4 STARTUPSTATE	Startup State Indicates the current sub-state of the startup procedure. Protocol related variable: vPOC!StartupState 0010b - POC:coldstart collision resolution 0011b - POC:coldstart listen 0100b - POC:integration consistency check 0101b - POC:integration listen 0111b - POC:initialize schedule 1010b - POC:coldstart consistency check 1101b - POC:integration coldstart check

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1110b - POC:coldstart gap 1111b - POC:coldstart join All other values are reserved.
3 —	Reserved
2-0 WAKEUPSTATUS	Wakeup Status Provides the outcome of the execution of the wakeup mechanism. Protocol-related variable: vPOC!WakeupStatus 000b - UNDEFINED 001b - RECEIVED_HEADER 010b - RECEIVED_WUP 011b - COLLISION_HEADER 100b - COLLISION_WUP 101b - COLLISION_UNKNOWN 110b - TRANSMITTED 111b - Reserved

56.5.22 Protocol Status 1 (PSR1)

Offset

Register	Offset
PSR1	2Ah

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CSAA	CSP	0	REMCSAT					CPN	HHR	FRZ	APTAC				
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15	Cold Start Attempt Aborted Flag

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESWIN Automotive Technology Co., Ltd.
8c4625-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
CSAA	Becomes 1 when CC aborts a cold start attempt. Protocol related event: set coldstart abort indicator in CHI 0b - No such event 1b - Cold start attempt aborted
14 CSP	Leading Cold Start Path Becomes 1 when CC reaches the POC:normal active state via the leading cold start path. This indicates that this node has started the network. 0b - No such event 1b - POC:normal active reached from POC:startup state via the leading cold start path
13 —	Reserved
12-8 REMCSAT	Remaining Coldstart Attempts Provides the number of remaining cold start attempts that CC will execute. Protocol related variable: vRemainingColdstartAttempts
7 CPN	Leading Cold Start Path Noise Becomes 1 if CC reaches the POC:normal active state via the leading cold start path under noise conditions. This indicates some activity occurred on the FlexRay bus while CC was starting up the cluster. Protocol related variable: vPOC!ColdstartNoise 0b - No such event 1b - POC:normal active state was reached from POC:startup state via noisy leading cold start path
6 HHR	Host Halt Request Pending Becomes 1 when CC receives the HALT command from the application via Protocol Operation Control Register (POCR). CC returns this field to 0 after a hard reset condition or when the protocol is in the POC:default config state. Protocol related variable: vPOC!CHIHaltRequest 0b - No such event 1b - HALT command received
5 FRZ	Freeze Occurred Becomes 1 when CC reaches the POC:halt state because of the host FREEZE command or an internal error condition requiring immediate halt. CC returns this field to 0 after a hard reset condition or when the protocol is in the POC:default config state. Protocol related variable: vPOC!Freeze

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No such event 1b - Immediate halt because of FREEZE or internal error condition
4-0 APTAC	Allow Passive To Active Counter Provides the number of consecutive even/odd communication cycle pairs that have passed with valid rate and offset correction terms, but the protocol is still in the POC:normal passive state because of an application configured delay to enter the POC:normal active state. This delay is defined by PCR12[allow_passive_to_active]. Protocol related variable: vPOC!vAllowPassivetoActive

56.5.23 Protocol Status Register 2 (PSR2)

Offset

Register	Offset
PSR2	2Ch

Function

Provides a snapshot of status information about:

- NIT
- The symbol window
- Clock synchronization

CC updates the NIT-related status fields—NBVB, NSEB, NBVA, and NSEA—after the end of the NIT and before the end of the first slot of the next communication cycle.

CC updates the fields related to the symbol window—STCB, SBVB, SSEB, MTB, STCA, SBVA, SSEB, and MTA—after the end of the symbol window and before the end of the current communication cycle. If no symbol window is configured, the status fields related to the symbol window remain in their reset state.

CC updates the clock-synchronization-related CLKCORRFAILCNT after the end of the static segment and before the end of the current communication cycle.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-8a8-b251-390d3ab624

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NBVB	NSEB	STCB	SBVB	SSEB	MTB	NBVA	NSEA	STCA	SBVA	SSEA	MTA	CKCORFCNT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 NBVB	NIT Boundary Violation On Channel B Becomes 1 when some media activity occurs on the FlexRay bus channel B at the end of NIT. Protocol related variable: vSS!BViolation for NIT on channel B 0b - No such event 1b - Media activity detected at boundaries
14 NSEB	NIT Syntax Error On Channel B Becomes 1 when a syntax error is detected during NIT on channel B. Protocol related variable: vSS!SyntaxError for NIT on channel B 0b - No such event 1b - Syntax error detected
13 STCB	Symbol Window Transmit Conflict On Channel B Becomes 1 if a transmission conflict occurs during the symbol window on channel B. Protocol related variable: vSS!TxConflict for symbol window on channel B 0b - No such event 1b - Transmission conflict detected
12 SBVB	Symbol Window Boundary Violation On Channel B Becomes 1 if some media activity occurs on the FlexRay bus channel B at the start or at the end of the symbol window. Protocol related variable: vSS!BViolation for symbol window on channel B 0b - No such event 1b - Media activity detected at boundaries
11 SSEB	Symbol Window Syntax Error On Channel B Becomes 1 when a syntax error is detected during the symbol window on channel B. Protocol related variable: vSS!SyntaxError for symbol window on channel B 0b - No such event 1b - Syntax error detected

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 MTB	MTS Received On Channel B Becomes 1 if MTS is received in the symbol window on channel B. Protocol related variable: vSS!ValidMTS for Symbol Window on channel B 0b - No such event 1b - MTS received
9 NBVA	NIT Boundary Violation On Channel A Becomes 1 when some media activity occurs on the FlexRay bus channel A at the end of NIT. Protocol related variable: vSS!BViolation for NIT on channel A. 0b - No such event 1b - Media activity detected at boundaries
8 NSEA	NIT Syntax Error On Channel A Becomes 1 when a syntax error is detected during NIT on channel A. Protocol related variable: vSS!SyntaxError for NIT on channel A 0b - No such event 1b - Syntax error detected
7 STCA	Symbol Window Transmit Conflict On Channel A Becomes 1 if transmission conflicts occur during the symbol window on channel A. Protocol related variable: vSS!TxConflict for symbol window on channel A 0b - No such event 1b - Transmission conflict detected
6 SBVA	Symbol Window Boundary Violation On Channel A Becomes 1 if some media activity occurs on the FlexRay bus channel A at the start or at the end of the symbol window. Protocol related variable: vSS!BViolation for symbol window on channel A 0b - No such event 1b - Media activity detected at boundaries
5 SSEA	Symbol Window Syntax Error On Channel A Becomes 1 when a syntax error is detected during the symbol window on channel A. Protocol related variable: vSS!SyntaxError for symbol window on channel A 0b - No such event 1b - Syntax error detected
4	MTS Received On Channel A

Table continues on the next page...

Table continued from the previous page...

Field	Function
MTA	Becomes 1 if MTS is received in the symbol window on channel A. Protocol related variable: vSS!ValidMTS for symbol window on channel A 0b - No such event 1b - MTS received
3-0 CKCORFCNT	Clock Correction Failed Counter Provides the number of consecutive even/odd communication cycle pairs that have passed without clock synchronization having performed an offset or a rate correction because of lack of synchronization frames. This field does not increment when it reaches the value in either the max_without_clock_correction_fatal or max_without_clock_correction_passive fields in Protocol Configuration 8 (PCR8). CC resets this field: • After a hard reset condition • When the protocol enters the POC:normal active state • When both the rate and offset correction terms have been calculated successfully Protocol related variable: vClockCorrectionFailed

56.5.24 Protocol Status 3 (PSR3)

Offset

Register	Offset
PSR3	2Eh

Function

Provides:

- Aggregated channel status information as an accrued status of channel activity for all communication slots, regardless of whether they are assigned for transmission or subscribed for reception
- Accrued information for:
  - The symbol window
  - NIT
  - The wakeup status

Provided under NDA of 1.1
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4824-48a8-b251-390d3a8624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		WUB	ABVB	AACB	ACEB	ASEB	AVFB	0		WUA	ABVA	AACA	ACEA	ASEA	AVFA
W			W1C	W1C	W1C	W1C	W1C	W1C			W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 —	Reserved
13 WUB	Wakeup Symbol Received On Channel B Becomes 1 when a wakeup symbol was received on channel B. 0b - Not received 1b - Received
12 ABVB	Aggregated Boundary Violation On Channel B Becomes 1 when a boundary violation has been detected on channel B. Boundary violations are detected in the communication slots, the symbol window, and NIT. 0b - No violation detected 1b - Violation detected
11 AACB	Aggregated Additional Communication On Channel B Becomes 1 when at least one valid frame is received on channel B in a slot that also contains an additional communication with syntax errors, content errors, or boundary violations. 0b - No additional communication detected 1b - Additional communication detected
10 ACEB	Aggregated Content Error On Channel B Becomes 1 when a content error is detected on channel B. Content errors are detected in the communication slots, the symbol window, and NIT. 0b - No error detected 1b - Errors detected
9 ASEB	Aggregated Syntax Error On Channel B Becomes 1 when a syntax error is detected on channel B. Syntax errors are detected in the communication slots, the symbol window and NIT. 0b - No error detected 1b - Errors detected
8	Aggregated Valid Frame On Channel B

Table continues on the next page...

Table continued from the previous page...

Field	Function
AVFB	Becomes 1 when a syntactically correct valid frame is received in any static or dynamic slot through channel B. 0b - No syntactically valid frames received 1b - At least one syntactically valid frame received
7-6 —	Reserved
5 WUA	Wakeup Symbol Received On Channel A Becomes 1 when a wakeup symbol is received on channel A. 0b - Not received 1b - Received
4 ABVA	Aggregated Boundary Violation On Channel A Becomes 1 when a boundary violation is detected on channel A. Boundary violations are detected in the communication slots, the symbol window, and NIT. 0b - No violation detected 1b - Violation detected
3 AACA	Aggregated Additional Communication On Channel A Becomes 1 when a valid frame is received in a slot on channel A that also contains an additional communication with syntax errors, content errors, or boundary violations. 0b - No additional communication detected 1b - Additional communication detected
2 ACEA	Aggregated Content Error On Channel A Becomes 1 when a content error is detected on channel A. Content errors are detected in the communication slots, the symbol window, and NIT. 0b - No error detected 1b - Errors detected
1 ASEA	Aggregated Syntax Error On Channel A Becomes 1 when a syntax error is detected on channel A. Syntax errors are detected in the communication slots, the symbol window, and NIT. 0b - No error detected 1b - Errors detected
0 AVFA	Aggregated Valid Frame On Channel A

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Becomes 1 when a syntactically correct valid frame is received in any static or dynamic slot through channel A. 0b - No syntactically valid frames received 1b - At least one syntactically valid frame received

56.5.25 MT Counter (MTCTR)

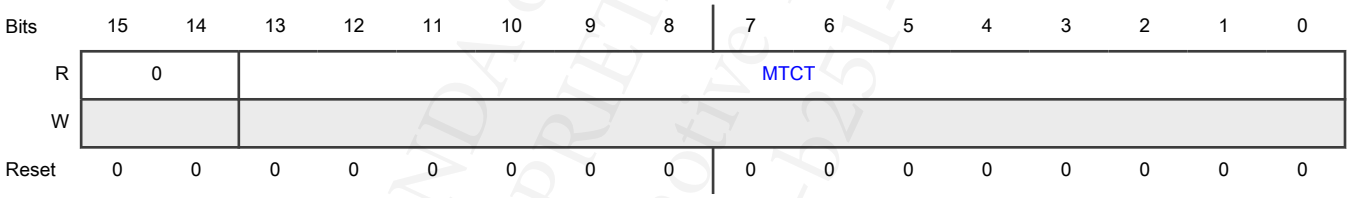
Offset

Register	Offset
MTCTR	30h

Function

Provides the MT count of the current communication cycle.

Diagram



Fields

Field	Function
15-14 —	Reserved
13-0 MTCT	MT Counter Provides the MT count of the current communication cycle. Protocol related variable: vMacrotick

56.5.26 Cycle Counter (CYCTR)

Offset

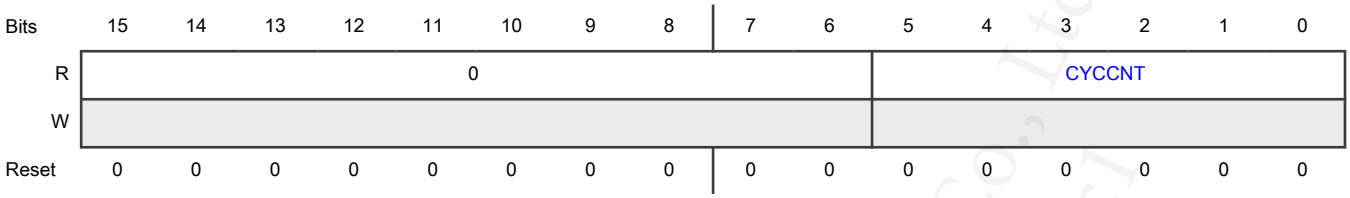
Register	Offset
CYCTR	32h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Function

Provides the number of the current communication cycle.

Diagram



Fields

Field	Function
15-6 —	Reserved
5-0 CYCCNT	Cycle Counter Provides the number of the current communication cycle. If the counter reaches the maximum value of 63, the counter wraps and starts from zero again. Protocol related variable: vCycleCounter.

56.5.27 Slot Counter Channel A (SLTCTAR)

Offset

Register	Offset
SLTCTAR	34h

Function

Provides the number of the current slot in the current communication cycle for channel A.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-590d3ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					SLOTCNTA										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-11 —	Reserved
10-0 SLOTCNTA	Slot Counter Value For Channel A Provides the number of the current slot in the current communication cycle. Protocol related variable: vSlotCounter for channel A

56.5.28 Slot Counter Channel B (SLTCTBR)

Offset

Register	Offset
SLTCTBR	36h

Function

Provides the number of the current slot in the current communication cycle for channel B.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					SLOTCNTB										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-11 —	Reserved
10-0	Slot Counter Value For Channel B

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
05c462e5-4821-4848-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
SLOTCNTB	Provides the number of the current slot in the current communication cycle. Protocol related variable: vSlotCounter for channel B

56.5.29 Rate Correction Value (RTCORVR)

Offset

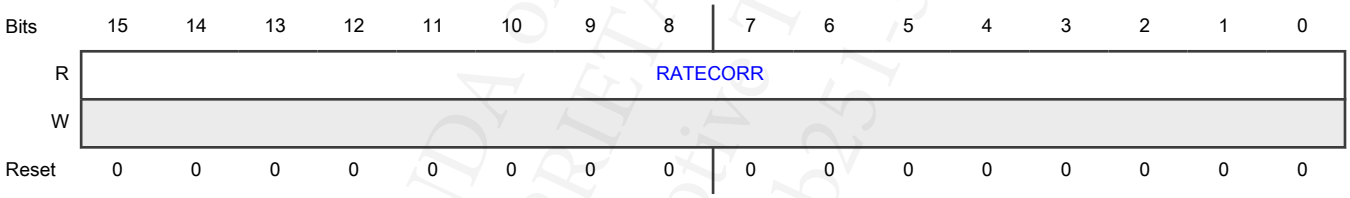
Register	Offset
RTCORVR	38h

Function

Provides the sign-extended rate correction value in µTs as the clock synchronization algorithm calculates it.

CC updates this register during NIT of each odd-numbered communication cycle.

Diagram



Fields

Field	Function
15-0 RATECORR	Rate Correction Value Provides the sign-extended rate correction value in µTs as the clock synchronization algorithm calculates it. The value is represented in twos-complement format. This value does not include the value limitation and the application of the external rate correction. If the magnitude of the internally calculated rate correction value exceeds the limit given by PCR14[rate_correction_out], CC changes PIFR0[CCL_IF] to 1. Protocol related variable: vRateCorrection (before value limitation and external rate correction). **NOTE** If CC cannot calculate a new rate correction term because of a lack of synchronization frames, the RATECORR value is not updated.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-1b2_48a8-b251-390c3ab624c1

56.5.30 Offset Correction Value (OFCORVR)

Offset

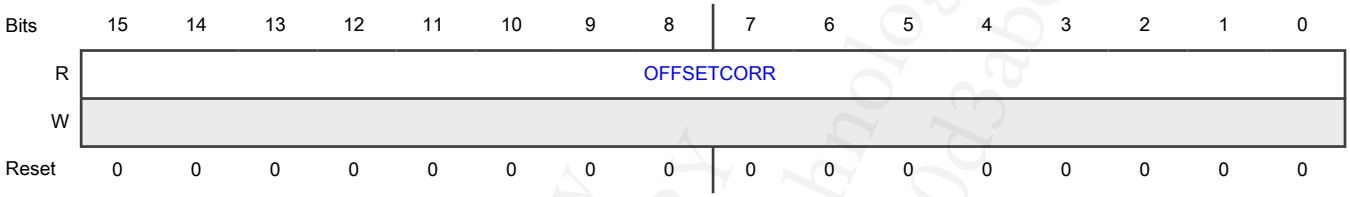
Register	Offset
OFCORVR	3Ah

Function

Provides the sign-extended offset correction value in µTs as the clock synchronization algorithm calculates it.

CC updates this register during NIT.

Diagram



Fields

Field	Function
15-0 OFFSETCORR	Offset Correction Value Provides the sign-extended offset correction value in µTs as the the clock synchronization algorithm calculates it. The value is represented in twos-complement format. This value does not include the value limitation and the application of the external offset correction. If the magnitude of the internally-calculated rate correction value exceeds the limit set by PCR29[extern_offset_correction], CC changes PIFR0[CCL_IF] to 1. Protocol related variable: vOffsetCorrection (before value limitation and external offset correction) **NOTE** If CC cannot calculate a new offset correction term because of a lack of synchronization frames, the OFFSETCORR value is not updated.

56.5.31 Combined Interrupt Flag (CIFR)

Offset

Register	Offset
CIFR	3Ch

Function

Provides five combined interrupt flags and a copy of three individual interrupt flags.

The combined interrupt flags are the result of a binary OR of the values of other interrupt flags regardless of the state of the interrupt enable fields. The generation scheme for the combined interrupt flags is described in Module Interrupt.

The individual interrupt flags WUPIF, FAFBIF, and FAFAIF are copies of corresponding flags in Global Interrupt Flag And Enable (GIFER) and are provided here to simplify the application interrupt flag check. To clear the individual interrupt flags, you must use Global Interrupt Flag And Enable (GIFER).

NOTE

The meanings of the combined status fields MIF, PRIF, CHIF, RBIF, and TBIF are different from those mentioned in Global Interrupt Flag And Enable (GIFER).

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								MIF	PRIF	CHIF	WUPIF	FAFBIF	FAFAIF	RBIF	TBIF
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8 —	Reserved
7 MIF	Module Interrupt Flag Becomes 1 if at least one interrupt source has its interrupt flag equal to 1. 0b - No interrupt source has its interrupt flag equal to 1. 1b - At least one interrupt source has its interrupt flag equal to 1.
6 PRIF	Protocol Interrupt Flag Becomes 1 if at least one of the individual protocol interrupt flags in Protocol Interrupt Flag 0 (PIFR0) or Protocol Interrupt Flag 1 (PIFR1) is 1. 0b - All individual protocol interrupt flags are 0. 1b - At least one of the individual protocol interrupt flags is 1.
5 CHIF	CHI Interrupt Flag Becomes 1 if at least one of the individual CHI error flags in CHI Error Flag (CHIERFR) is 1. 0b - All CHI error flags are 0. 1b - At least one CHI error flag is 1.
4 WUPIF	Wakeup Interrupt Flag Provides the same value as GIFER[WUPIF].
3 FAFBIF	Receive FIFO Channel B Almost Full Interrupt Flag Provides the same value as GIFER[FAFBIF].
2	Receive FIFO Channel A Almost Full Interrupt Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
FAFAIF	Provides the same value as GIFER[FAFAIF].
1 RBIF	Receive MB Interrupt Flag Becomes 1 if for at least one of the individual receive MBs (MBCCSR*n*[MTD] = 0), the corresponding MBCCSR*n*[MBIF] = 1. 0b - None of the individual receive MBs have have MBIF = 1. 1b - At least one individual receive MB has MBIF = 1.
0 TBIF	Transmit MB Interrupt Flag Becomes 1 if for at least one of the individual transmit MBs (MBCCSR*n*[MTD] = 1), the corresponding MBCCSR*n*[MBIF] = 1. 0b - None of the individual transmit MBs have MBIF = 1. 1b - At least one individual transmit MB has MBIF = 1.

56.5.32 System Memory Access Timeout (SYMATOR)

Offset

Register	Offset
SYMATOR	3Eh

Function

Controls when a system bus access timeout is detected.

For a detailed description, see Configure System Memory Access Time-Out Register (SYMATOR) and System Bus Access Timeout.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								TIMEOUT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Fields

Field	Function
15-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 TIMEOUT	System Memory Access Timeout Defines when a system bus access timeout is detected.

56.5.33 Sync Frame Counter (SFCNTR)

Offset

Register	Offset
SFCNTR	40h

Function

Provides the number of synchronization frames that are used for clock synchronization in the last even and in the last odd numbered communication cycle.

This register is updated after the start of NIT and before 10 MT after offset correction start.

If you have locked the even synchronization table at the end of the static segment of an even communication cycle, CC does not update the SFEVB and SFEVA fields.

Similarly, if you have locked the odd synchronization table at the end of the static segment of an odd communication cycle, CC does not update the SFODB and SFODA fields.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SFEVB				SFEVA				SFODB				SFODA			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-12 SFEVB	Sync Frames Channel B, Even Cycle Provides the size of the internal list of frame IDs of received synchronization frames used for clock synchronization. Protocol related variable: size of (vsSyncIdListB for even cycle)
11-8 SFEVA	Sync Frames Channel A, Even Cycle Provides the size of the internal list of frame IDs of received synchronization frames used for clock synchronization. Protocol related variable: size of (vsSyncIdListA for even cycle)

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-4 SFODB	Sync Frames Channel B, Odd cycle Provides the size of the internal list of frame IDs of received synchronization frames used for clock synchronization. Protocol related variable: size of (vsSyncIdListB for odd cycle)
3-0 SFODA	Sync Frames Channel A, Odd Cycle Provides the size of the internal list of frame IDs of received synchronization frames used for clock synchronization. Protocol related variable: size of (vsSyncIdListA for odd cycle)

56.5.34 Sync Frame Table Offset (SFTOR)

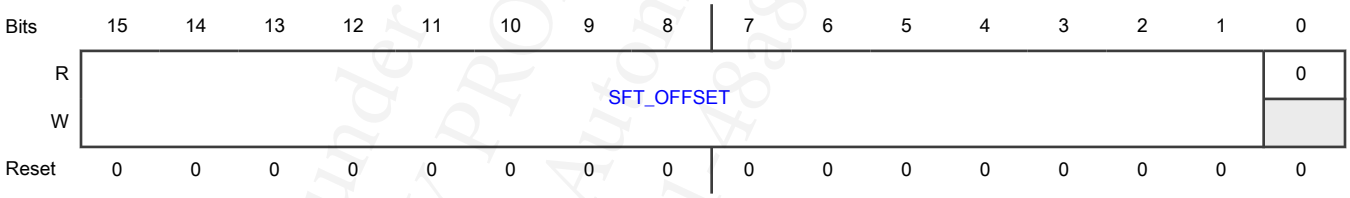
Offset

Register	Offset
SFTOR	42h

Function

Defines the FlexRay memory area related offset for sync frame tables. For more details, see Sync Frame ID and Sync Frame Deviation Tables.

Diagram



Fields

Field	Function
15-1 SFT_OFFSET	Sync Frame Table Offset Configures the offset of the sync frame tables in the FlexRay memory area. This offset must be 16-bit aligned. Thus, SFT_OFFSET[0] is always 0.
0 —	Reserved

Printed under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e25-4821-4848-b251-390d3ab624c1

56.5.35 Sync Frame Table Configuration Control Status (SFTCCSR)

Offset

Register	Offset
SFTCCSR	44h

Function

Provides configuration, control, and status information related to the generation and access of the clock sync ID tables and clock sync measurement tables. For a detailed description, see Sync Frame ID and Sync Frame Deviation Tables.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	CYCNUM						ELKS	OLKS	EVAL	OVAL	0	0	SDVEN	SIDEN
W	ELKT	OLKT												OPT		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 ELKT	Even Cycle Tables Lock/Unlock Trigger Locks and unlocks the even cycle tables. 0b - No effect 1b - Trigger lock and unlock
14 OLKT	Odd Cycle Tables Lock And Unlock Trigger Locks and unlocks the odd cycle tables. 0b - No effect 1b - Trigger lock and unlock
13-8 CYCNUM	Cycle Number Provides the number of the cycle in which the currently locked table was recorded. If none or both tables are locked, this value is related to the even cycle table.
7 ELKS	Even Cycle Tables Lock Status Indicates whether the application has locked the even cycle tables. 0b - Not locked 1b - Locked
6 OLKS	Odd Cycle Tables Lock Status Indicates whether the application has locked the odd cycle tables.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Not locked 1b - Locked
5 EVAL	Even Cycle Tables Valid Indicates whether the sync frame ID and sync frame deviation tables for the even cycle are valid. CC writes 0 to this field when CC starts updating the tables, and keeps its value at 0 while the update occurs. CC then writes 1 to this field when CC has finished the table update and the tables are consistent. 0b - Invalid 1b - Valid
4 OVAL	Odd Cycle Tables Valid Indicates whether the sync frame ID and sync frame deviation tables for the odd cycle are valid. CC writes 0 to this field when CC starts updating the tables, and keeps its value at 0 while the update occurs. CC then writes 1 to this field when CC has finished the table update and the tables are consistent. 0b - Invalid 1b - Valid
3 —	Reserved
2 OPT	One Pair Trigger Controls how CC writes pairs of sync frame tables into the FlexRay memory area. If OPT = 1 while SDVEN or SIDEN is 1, CC writes only one pair of the enabled sync frame tables corresponding to the next even-odd-cycle pair into the FlexRay memory area. In this case, CC immediately changes SDVEN or SIDEN to 0. If OPT = 0 while SDVEN or SIDEN is 1, CC continuously writes the enabled sync frame tables into the FlexRay memory area. 0b - Write continuously 1b - Write only one pair
1 SDVEN	Sync Frame Deviation Table Enable Controls the generation of the sync frame deviation tables. You must write 1 to this field to request CC to write the sync frame deviation tables into the FlexRay memory area. **NOTE** If SDVEN = 1, then you must also write 1 to SIDEN. 0b - Do not write sync frame deviation tables 1b - Write sync frame deviation tables
0	Sync Frame ID Table Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
SIDEN	Controls the generation of the sync frame ID tables. You must write 1 to this field to request CC to write the sync frame ID tables into the FlexRay memory area. 0b - Do not write sync frame ID tables 1b - Write sync frame ID tables

56.5.36 Sync Frame ID Rejection Filter (SFIDRFR)

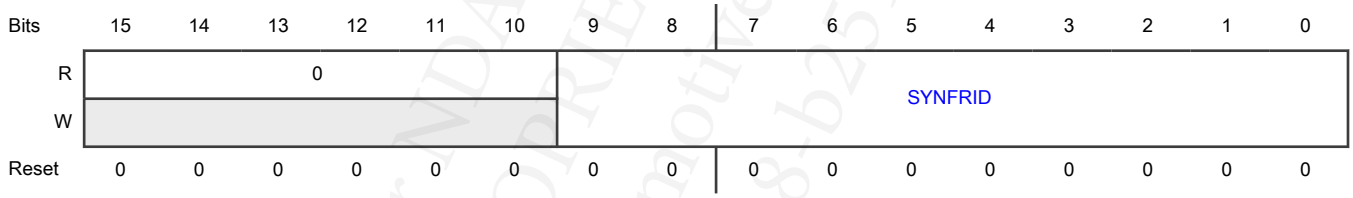
Offset

Register	Offset
SFIDRFR	46h

Function

Defines the sync frame rejection filter ID. You must update this register outside of the static segment. If you update this register in the static segment, it can appear that CC accepts the sync frame in the current cycle.

Diagram



Fields

Field	Function
15-10 —	Reserved
9-0 SYNFRID	Sync Frame Rejection ID Defines the frame ID of a frame that must not be used for clock synchronization. For details, see Sync Frame Rejection Filtering.

56.5.37 Sync Frame ID Acceptance Filter Value (SFIDAFVR)

Offset

Register	Offset
SFIDAFVR	48h

Function

Defines the sync frame acceptance filter value. For details on filtering, see Sync Frame Filtering.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								FVAL							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-10 —	Reserved
9-0 FVAL	Filter Value Defines the value for the sync frame acceptance filtering.

56.5.38 Sync Frame ID Acceptance Filter Mask (SFIDAFMR)

Offset

Register	Offset
SFIDAFMR	4Ah

Function

Defines the sync frame acceptance filter mask. For details on filtering, see Sync Frame Acceptance Filtering.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								FMSK							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-10 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
9-0 FMSK	Filter Mask Defines the mask for the sync frame acceptance filtering.

56.5.39 NMV n (NMVR0 - NMVR5)

Offset

Register	Offset
NMVR0	4Ch
NMVR1	4Eh
NMVR2	50h
NMVR3	52h
NMVR4	54h
NMVR5	56h

Function

Each of these registers holds one NMV part. Network Management Vector Length Register (NMVLR) configures the NMV length. If Network Management Vector Length Register (NMVLR) is programmed with a value that is less than 12 bytes, the remaining bytes of the NMVR*n* registers, which are not used for the network management vector accumulating, remain 0.

The NMVR provides accrued information over all received NMVs in the last communication cycle. All NMVs received in one cycle are ORed into the NMVR. The NMVR is updated at the end of the communication cycle.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NMVP															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 NMVP	NMV Part Contains part of the NMV. The following table shows the mapping between the NMVR*n* and the receive MB payload bytes in NMV[0:11].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a6-b1-390d3ab624c1

Field	Function
	Table 462. Mapping of NMVRn to the received payload bytes NMVn

Register, field, and bit position	NMVn received payload
NMVR0[NMVP[15:8]]	NMV0
NMVR0[NMVP[7:0]]	NMV1
NMVR1[NMVP[15:8]]	NMV2
NMVR1[NMVP[7:0]]	NMV3
...	
NMVR5[NMVP[15:8]]	NMV10
NMVR5[NMVP[7:0]]	NMV11

56.5.40 Network Management Vector Length Register (NMVLR)

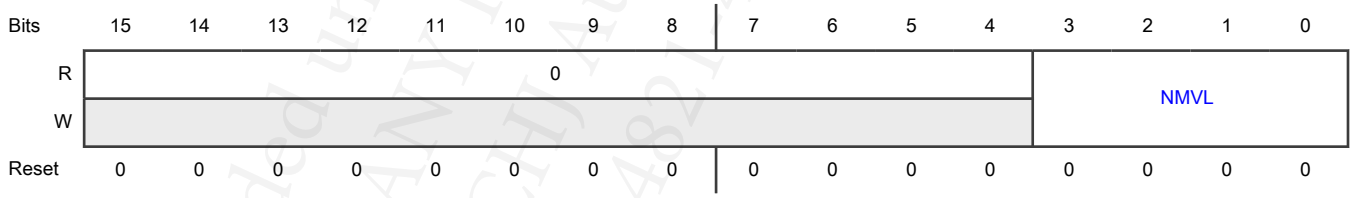
Offset

Register	Offset
NMVLR	58h

Function

This register defines the length of the network management vector in bytes.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0	Network Management Vector Length

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4888-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
NMVL	Protocol related variable: gNetworkManagementVectorLength. This field defines the length of the Network Management Vector in bytes. Legal values are between 0 and 12.

56.5.41 Timer Configuration And Control (TICCR)

Offset

Register	Offset
TICCR	5Ah

Function

Configures and controls the two timers T1 and T2. For timer details, see Timer Support.

T1 is an absolute timer. T2 can be configured as an absolute or relative timer.

Both timers are deactivated immediately when the protocol enters a state different from POC:normal active or POC:normal passive.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		T2_CFG	T2_REP	0	0	0	T2ST	0			T1_REP	0	0	0	T1ST
W						T2SP	T2TR							T1SP	T1TR	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 —	Reserved
13 T2_CFG	Timer T2 Configuration Configures the timebase mode (absolute or relative) of timer T2. 0b - Absolute 1b - Relative
12 T2_REP	Timer T2 Repetitive Mode Configures the repetition mode of Timer T2. 0b - Non-repetitive 1b - Repetitive

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 —	Reserved
10 T2SP	Timer T2 Stop Stops timer T2. 0b - No effect 1b - Stop
9 T2TR	Timer T2 Trigger Starts timer T2. 0b - No effect 1b - Start
8 T2ST	Timer T2 State Provides the current state of timer T2. 0b - Idle 1b - Running
7-5 —	Reserved
4 T1_REP	Timer T1 Repetitive Mode Configures the repetition mode of timer T1. 0b - Non-repetitive 1b - Repetitive
3 —	Reserved
2 T1SP	Timer T1 Stop Stops timer T1. 0b - No effect 1b - Stop
1 T1TR	Timer T1 Trigger Starts timer T1. 0b - No effect 1b - Start

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
0 T1ST	Timer T1 State Provides the current state of timer T1. 0b - Idle 1b - Running

56.5.42 Timer 1 Cycle Set Register (TI1CYSR)

Offset

Register	Offset
TI1CYSR	5Ch

Function

Defines the cycle filter value and the cycle filter mask for timer T1. For a detailed description of timer T1, see Absolute Timer T1.

NOTE

If you modify the value in this register while the timer is running, the change becomes effective immediately and timer T1 expires according to the changed value.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		T1_CYC_VAL						0		T1_CYC_MSK					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 —	Reserved
13-8 T1_CYC_VAL	Timer T1 Cycle Filter Value Defines the cycle filter value for timer T1.
7-6 —	Reserved
5-0 T1_CYC_MSK	Timer T1 Cycle Filter Mask Defines the cycle filter mask for timer T1.

56.5.43 Timer 1 MT Offset (TI1MTOR)

Offset

Register	Offset
TI1MTOR	5Eh

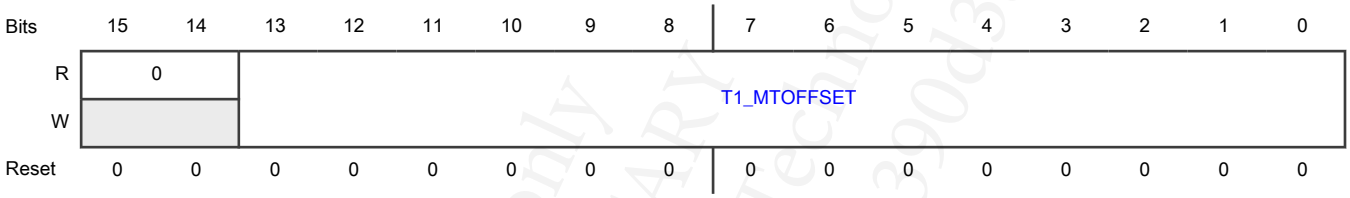
Function

Holds the MT offset value for timer T1. For a detailed description of timer T1, see Absolute Timer T1.

NOTE

If you modify the value in this register while the timer is running, the change becomes effective immediately and timer T1 expires according to the changed value.

Diagram



Fields

Field	Function
15-14 —	Reserved
13-0 T1_MTOFFSET	Timer 1 MT Offset Defines the MT offset value for timer 1.

56.5.44 Timer 2 Configuration 0 (Absolute Timer Configuration) (TI2CR0_ABS)

Offset

Register	Offset
TI2CR0_ABS	60h

Function

The content of this register depends on the value of TICCR[T2_CFG]. For a detailed description of timer T2, see Absolute Timer T2.

NOTE

If you configure timer T2 as an absolute timer and modify the values in this register while the timer is running, the change becomes effective immediately and timer T2 expires according to the changed values.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		T2CYCVAL						Reserved		T2CYCMSK					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 —	Reserved Do not change the value of these bits. Always use a read-modify-write operation.
13-8 T2CYCVAL	Timer T2 Cycle Filter Mask
7-6 —	Reserved Do not change the value of these bits. Always use a read-modify-write operation.
5-0 T2CYCMSK	Timer T2 Cycle Filter Mask

56.5.45 Timer 2 Configuration 0 (Relative Timer Configuration) (TI2CR0_REL)

Offset

Register	Offset
TI2CR0_REL	60h

Function

The content of this register depends on the value of TICCR[T2_CFG]. For a detailed description of timer T2, see Relative Timer T2.

NOTE

If you configure timer T2 as a relative timer and change the values in this register while the timer is running, the change becomes effective when the timer expires according to the old values.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CAL Automotive Technology Co., Ltd.
c8c462e5-a02c-48a8-b251-390d3ab624d1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	T2MTCNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 T2MTCNT	Timer T2 MT High Word

56.5.46 Timer 2 Configuration 1 (Absolute Timer Configuration) (TI2CR1_ABS)

Offset

Register	Offset
TI2CR1_ABS	62h

Function

The content of this register depends on the value of TICCR[T2_CFG]. For a detailed description of timer T2, see Absolute Timer T2.

NOTE

If you configure timer T2 as an absolute timer and modify the values in this register while the timer is running, the change becomes effective immediately and timer T2 expires according to the changed values.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		T2MOFF													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 —	Reserved Do not change the value of these bits. Always use a read-modify-write operation.
13-0 T2MOFF	Timer T2 MT Offset

56.5.47 Timer 2 Configuration 1 (Relative Timer Configuration) (TI2CR1_REL)

Offset

Register	Offset
TI2CR1_REL	62h

Function

The content of this register depends on the value of TICCR[T2_CFG]. For a detailed description of timer T2, see Relative Timer T2.

NOTE

If you configure timer T2 as a relative timer and change the values in this register while the timer is running, the change becomes effective when the timer has expired according to the old values.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	T2MTCNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 T2MTCNT	Timer T2 Macrotick Low Word

56.5.48 Slot Status Selection (SSSR)

Offset

Register	Offset
SSSR	64h

Function

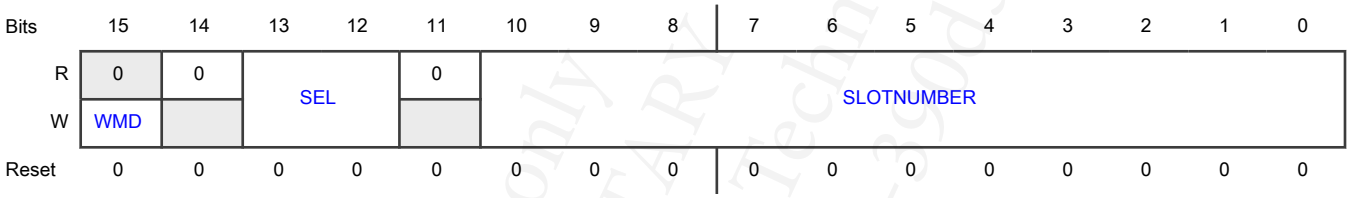
Used to access the four internal non-memory mapped slot status selection (SSSR0–SSSR3) registers. Each internal register selects a slot, symbol window, or NIT, whose status vector is saved in the corresponding slot status (SSR0–SSR7) registers according to Slot Status Selection (SSSR). For a detailed description of slot status monitoring, see Slot Status Monitoring.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e65-4821-48a8-b251-390d3ab6f461

Table 463. Mapping between SSSR*n* and SSR*n*

Internal slot status selection register	Write the slot status of the slot selected by SSSRn for each			
	Even communication cycle		Odd communication cycle	
	For channel B to	For channel A to	For channel B to	For channel A to
SSSR0	SSR0[15:8]	SSR0[7:0]	SSR1[15:8]	SSR1[7:0]
SSSR1	SSR2[15:8]	SSR2[7:0]	SSR3[15:8]	SSR3[7:0]
SSSR2	SSR4[15:8]	SSR4[7:0]	SSR5[15:8]	SSR5[7:0]
SSSR3	SSR6[15:8]	SSR6[7:0]	SSR7[15:8]	SSR7[7:0]

Diagram



Fields

Field	Function
15 WMD	Write Mode Defines whether a write access to this register modifies all fields or only the SEL field. 0b - All fields 1b - Only the SEL field
14 —	Reserved
13-12 SEL	Selector Selects one of the four internal slot status selection registers for access. 00b - SSSR0 01b - SSSR1 10b - SSSR2 11b - SSSR3
11 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
10-0 SLOTNUMBER	Slot Number Specifies the number of the slot whose status is saved in the corresponding slot status registers. **NOTE** If SLOTNUMBER = 0, the related slot status register provides the status of the symbol window after the NIT start, and provides the NIT status after the cycle start.

56.5.49 Slot Status Counter Condition (SSCCR)

Offset

Register	Offset
SSCCR	66h

Function

Allows you to access and program the four internal non-memory mapped Slot Status Counter Condition (SSCCR0–SSCCR3) registers. Each of these registers define the mode and the conditions for incrementing the counter in the corresponding Slot Status Counter (SSCR0-SSCR3) registers. Table 464 shows the mapping. For a detailed description of slot status counters, see Slot Status Counter Registers.

Table 464. Mapping between internal SSCCR*n* and SSCR*n*

Condition register	Condition defined for register
SSCCR0	SSCR0
SSCCR1	SSCR1
SSCCR2	SSCR2
SSCCR3	SSCR3

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-9251-90d3ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	SEL		0	CNTCFG		MCY	VFR	SYF	NUF	SUF	STATUSMASK			
W	WMD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 WMD	Write Mode Defines whether a write access to this register modifies all fields or only the SEL field. 0b - All fields 1b - Only the SEL field
14 —	Reserved
13-12 SEL	Selector Selects one of the four internal slot counter condition registers for access. 00b - SSCCR0 01b - SSCCR1 10b - SSCCR2 11b - SSCCR3
11 —	Reserved
10-9 CNTCFG	Counter Configuration Controls the channel-related incrementing of the slot status counter. 00b - Increment by 1 if condition is fulfilled on channel A. 01b - Increment by 1 if condition is fulfilled on channel B. 10b - Increment by 1 if condition is fulfilled on at least one channel. 11b - Increment by 2 if condition is fulfilled on both channels; increment by 1 if condition is fulfilled on only one channel.
8 MCY	Multi Cycle Selection Defines whether the slot status counter accumulates over multiple communication cycles or provides information only for the previous communication cycle. 0b - Only for the previous communication cycle 1b - Over multiple communication cycles

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 VFR	Valid Frame Restriction This bit is used to restrict the counter only to valid received frames. 0b - Not restricted 1b - Restricted
6 SYF	Sync Frame Restriction Restricts the counter to received frames that have the sync frame indicator bit set to 1. 0b - Not restricted 1b - Restricted
5 NUF	Null Frame Restriction Restrict the counter to received frames that have the null frame indicator bit set to 0. 0b - Not restricted 1b - Restricted
4 SUF	Startup Frame Restriction Restricts the counter to received frames that have the startup frame indicator bit set to 1. 0b - Not restricted 1b - Restricted
3-0 STATUSMASK	Slot Status Mask Enable the counter with respect to the four slot status error indicator bits.

Bit position in STATUSMASK	Function
0	Enables the counting for slots with the transmission conflict indicator bit set to 1.
1	Enables the counting for slots with the boundary violation indicator bit set to 1.
2	Enables the counting for slots with the content error indicator bit set to 1.
3	Enables the counting for slots with the syntax error indicator bit set to 1.

56.5.50 Slot Status (SSR0 - SSR7)

Offset

For n = 0 to 7:

Register	Offset
SSRn	68h + (n × 2h)

Function

Holds the status vector of the slot specified in the corresponding internal slot status selection register, which can be programmed using Slot Status Selection (SSSR). Each register is updated after the end of the corresponding slot as shown in Slot Status Monitoring. The register fields directly relate to the protocol variables and described in more detail in Slot Status Monitoring.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 VFB	Valid Frame On Channel B Protocol-related variable: vSS!ValidFrame channel B 0b - vSS!ValidFrame = 0 1b - vSS!ValidFrame = 1
14 SYB	Sync Frame Indicator Channel B Protocol-related variable: vRF!Header!SyFIndicator channel B 0b - vRF!Header!SyFIndicator = 0 1b - vRF!Header!SyFIndicator = 1
13 NFB	Null Frame Indicator Channel B Protocol-related variable: vRF!Header!NFIndicator channel B 0b - vRF!Header!NFIndicator = 0 1b - vRF!Header!NFIndicator = 1
12 SUB	Startup Frame Indicator Channel B Protocol-related variable: vRF!Header!SuFIndicator channel B 0b - vRF!Header!SuFIndicator = 0 1b - vRF!Header!SuFIndicator = 1
11 SEB	Syntax Error On Channel B Protocol-related variable: vSS!SyntaxError channel B 0b - vSS!SyntaxError = 0 1b - vSS!SyntaxError = 1
10 CEB	Content Error On Channel B Protocol-related variable: vSS!ContentError channel B

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - vSS!ContentError = 0 1b - vSS!ContentError = 1
9 BVB	Boundary Violation On Channel B Protocol-related variable: vSS!BViolation channel B 0b - vSS!BViolation = 0 1b - vSS!BViolation = 1
8 TCB	Transmission Conflict On Channel B Protocol-related variable: vSS!TxConflict channel B 0b - vSS!TxConflict = 0 1b - vSS!TxConflict = 1
7 VFA	Valid Frame On Channel A Protocol-related variable: vSS!ValidFrame channel A 0b - vSS!ValidFrame = 0 1b - vSS!ValidFrame = 1
6 SYA	Sync Frame Indicator Channel A Protocol-related variable: vRF!Header!SyFIndicator channel A 0b - vRF!Header!SyFIndicator = 0 1b - vRF!Header!SyFIndicator = 1
5 NFA	Null Frame Indicator Channel A Protocol-related variable: vRF!Header!NFIndicator channel A 0b - vRF!Header!NFIndicator = 0 1b - vRF!Header!NFIndicator = 1
4 SUA	Startup Frame Indicator Channel A Protocol-related variable: vRF!Header!SuFIndicator channel A 0b - vRF!Header!SuFIndicator = 0 1b - vRF!Header!SuFIndicator = 1
3 SEA	Syntax Error On Channel A Protocol-related variable: vSS!SyntaxError channel A 0b - vSS!SyntaxError = 0 1b - vSS!SyntaxError = 1
2	Content Error On Channel A

Table continues on the next page...

Table continued from the previous page...

Field	Function
CEA	Protocol-related variable: vSS!ContentError channel A 0b - vSS!ContentError = 0 1b - vSS!ContentError = 1
1 BVA	Boundary Violation On Channel A Protocol-related variable: vSS!BViolation channel A 0b - vSS!BViolation = 0 1b - vSS!BViolation = 1
0 TCA	Transmission Conflict On Channel A Protocol-related variable: vSS!TxConflict channel A 0b - vSS!TxConflict = 0 1b - vSS!TxConflict = 1

56.5.51 Slot Status Counter (SSCR0 - SSCR3)

Offset

Register	Offset
SSCR0	78h
SSCR1	7Ah
SSCR2	7Ch
SSCR3	7Eh

Function

Provides the slot status counter value for the previous communication cycle(s) and is updated at the cycle start. The provided value depends on the control fields in the related internal slot status counter condition registers SSCCR*n*, which you can program by using Slot Status Counter Condition (SSCCR). For more details, see Slot Status Counter Registers.

NOTE

If the counter has reached its maximum value FFFFh and is in Multicycle mode—that is, SSCCR*n*[MCY] = 1—the counter is not reset to 0h.You can reset the counter by writing 0 to SSCCR*n*[MCY] and waiting for the next cycle start, when CC clears the counter. Subsequently, the counter can be set into Multicycle mode again.

Provided under NDA only
CONFIDENTIAL PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4626e-48a1-48a8-b251-390d3ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SLOTSTATUSCNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 SLOTSTATUSCNT	Slot Status Counter Provides the current value of the slot status counter.

56.5.52 MTS A Configuration (MTSACFR)

Offset

Register	Offset
MTSACFR	80h

Function

Controls the MTS transmission on channel A. For more details, see MTS Generation.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MTE	0	CYCCNTMSK						0	CYCCNTVAL						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 MTE	MTS Transmission Enable Enables and disables the MTS transmission in the selected set of cycles. 0b - Disable 1b - Enable
14 —	Reserved

Table continues on the next page...

Printed under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4848-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
13-8 CYCCNTMSK	Cycle Counter Mask Provides the filter mask for the MTS cycle count filter.
7-6 —	Reserved
5-0 CYCCNTVAL	Cycle Counter Value Provides the filter value for the MTS cycle count filter.

56.5.53 MTS B Configuration (MTSBCFR)

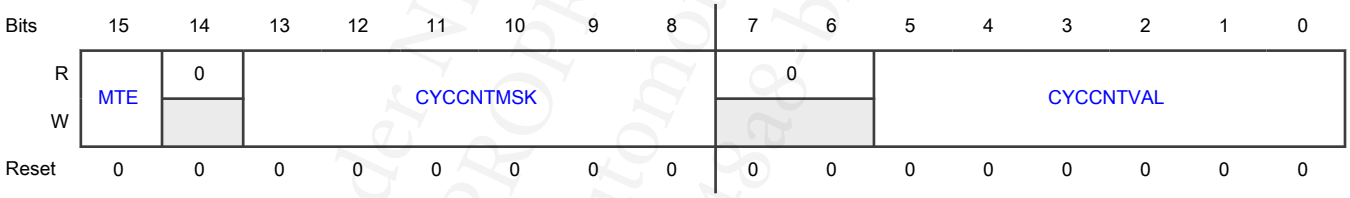
Offset

Register	Offset
MTSBCFR	82h

Function

Controls the MTS transmission on channel B. For more details, see MTS Generation.

Diagram



Fields

Field	Function
15 MTE	MTS Transmission Enable Enables and disable the MTS transmission in the selected set of cycles. 0b - Disable 1b - Enable
14 —	Reserved
13-8	Cycle Counter Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
CYCCNTMSK	Provides the filter mask for the MTS cycle count filter.
7-6 —	Reserved
5-0 CYCCNTVAL	Cycle Counter Value Provides the filter value for the MTS cycle count filter.

56.5.54 Receive Shadow Buffer Index (RSBIR)

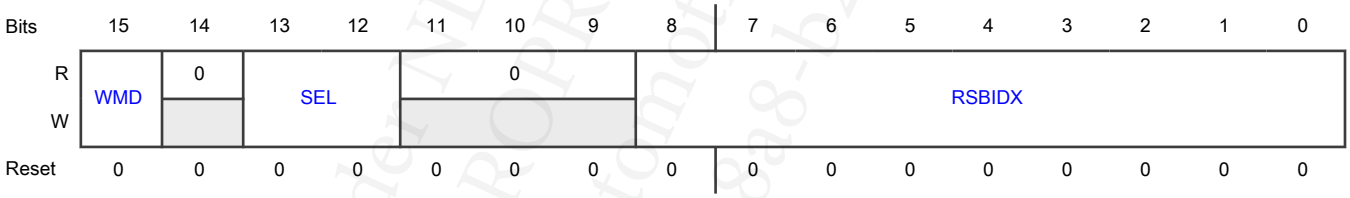
Offset

Register	Offset
RSBIR	84h

Function

Provides and retrieves the indices of the MB header fields currently associated with the receive shadow buffers. For more details on the receive shadow buffer concept, see Receive shadow buffer concept.

Diagram



Fields

Field	Function
15 WMD	Write Mode Controls the write mode for this register. 0b - Update SEL and RSBIDX field on register write 1b - Update only SEL field on register write
14 —	Reserved
13-12 SEL	Selector Selects the internal receive shadow buffer index register for access.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - RSBIR_A1—receive shadow buffer index register for channel A, segment 1 01b - RSBIR_A2—receive shadow buffer index register for channel A, segment 2 10b - RSBIR_B1—receive shadow buffer index register for channel B, segment 1 11b - RSBIR_B2—receive shadow buffer index register for channel B, segment 2
11-9 —	Reserved
8-0 RSBIDX	RSBIDXA1/RSBIDXA2/RSBIDXB1/RSBIDXB2- Receive Shadow Buffer Index Contains the current index of the MB header field of the receive shadow MB selected by the SEL field. CC uses this index to determine the physical location of the shadow buffer header field in the FlexRay memory area. CC updates this field during receive operation. The application provides an initial MB header index value in the configuration phase. CC—Updates the MB header index after successful reception. Application—Provides initial MB header index. Legal values are 0 ≤ i ≤ 259. Illegal values are detected during the MB search.

56.5.55 Receive FIFO Watermark And Selection (RFWMSR)

Offset

Register	Offset
RFWMSR	86h

Function

Used to:

- Select a receiver FIFO for subsequent programming access through the receiver FIFO configuration registers summarized in Receive FIFO Watermark And Selection (RFWMSR).
- Define the watermark for the selected FIFO.

Table 465. SEL controlled receiver FIFO registers

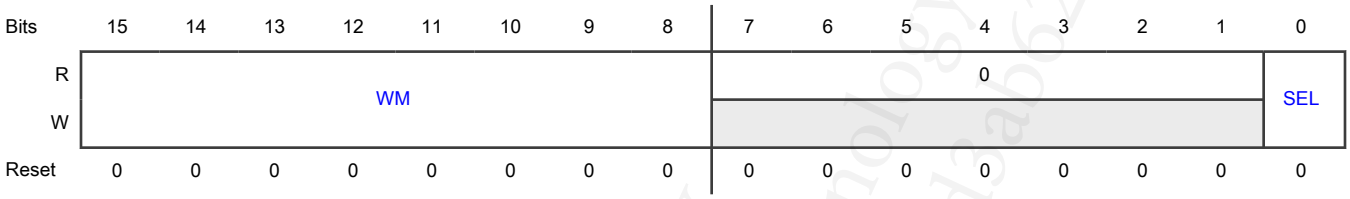
Register
Receive FIFO Start Index Register (RFSIR)
Receive FIFO Depth and Size Register (RFDSR)
Receive FIFO Message ID Acceptance Filter Value Register (RFMIDAFVR)
Receive FIFO Message ID Acceptance Filter Mask Register (RFMIDAFMR)

Table continues on the next page...

Table 465. SEL controlled receiver FIFO registers (continued)

Register
Receive FIFO Frame ID Rejection Filter Value Register (RFFIDRFVR)
Receive FIFO Frame ID Rejection Filter Mask Register (RFFIDRFMR)
Receive FIFO Range Filter Configuration Register (RFRFCFR)
Receive FIFO Range Filter Control Register (RFRFCTR)

Diagram



Fields

Field	Function
15-8 WM	WMA/WMB - Watermark Defines the watermark value for the selected FIFO. This value controls the generation of the almost full interrupt flags.
7-1 —	Reserved
0 SEL	Select Selects the receiver FIFO for subsequent programming. 0b - Channel A 1b - Channel B

56.5.56 Receive FIFO Start Index (RFSIR)

Offset

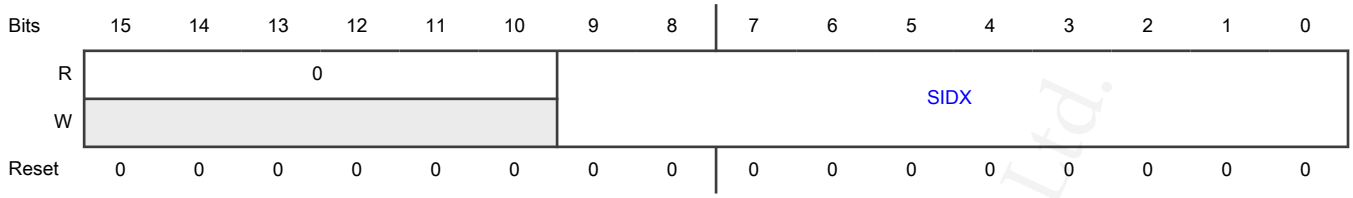
Register	Offset
RFSIR	88h

Function

Defines the MB header index of the first MB of the selected FIFO.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CTU Automotive Technology Co., Ltd.
cc462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
15-10 —	Reserved
9-0 SIDX	SIDXA Or SIDXB Start Index Defines the number of the MB header field of the first MB of the selected FIFO. CC uses the value of this field to determine the physical location of the receiver FIFO's first MB header field.

56.5.57 Receive FIFO Depth And Size (RFDSR)

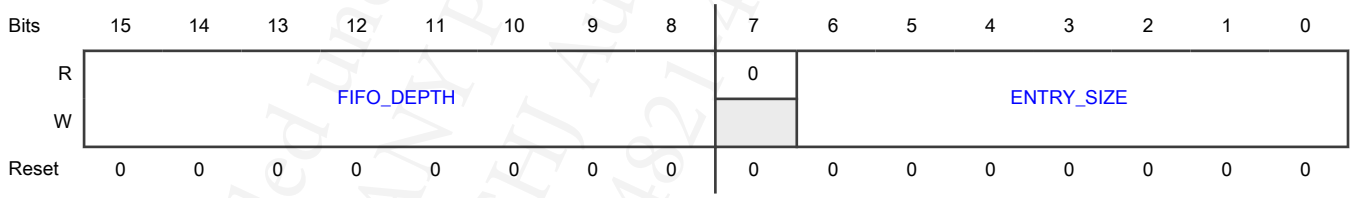
Offset

Register	Offset
RFDSR	8Ah

Function

Defines the structure—the number of entries and the size of each entry—of the selected FIFO. .

Diagram



Fields

Field	Function
15-8 FIFO_DEPTH	FIFO_DEPTHA And FIFO_DEPTHB FIFO Depth Defines the depth of the selected FIFO—that is, the number of entries.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE If you write 0 to FIFO_DEPTH, you must also write 0 to RFFIDRFMR[FIDRFMSK], to ensure that no frames are received into the FIFO.
7 —	Reserved
6-0 ENTRY_SIZE	ENTRY_SIZEA Or ENTRY_SIZEB Entry Size Defines the size of the frame data sections for the selected FIFO in 2-byte entities.

56.5.58 Receive FIFO A Read Index (RFARIR)

Offset

Register	Offset
RFARIR	8Ch

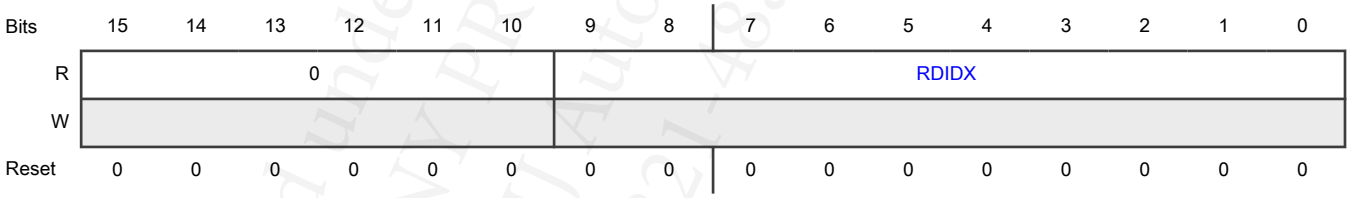
Function

Provides the MB header index of the next available FIFO A entry that the application can read.

NOTE

If the FIFO is empty, the RDIDX field points to a physical MB with invalid content.

Diagram



Fields

Field	Function
15-10 —	Reserved
9-0 RDIDX	Read Index Provides the MB header index of the next available FIFO MB that the application can read.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-9251-390d3ab624c1

56.5.59 Receive FIFO B Read Index (RFBRIR)

Offset

Register	Offset
RFBRIR	8Eh

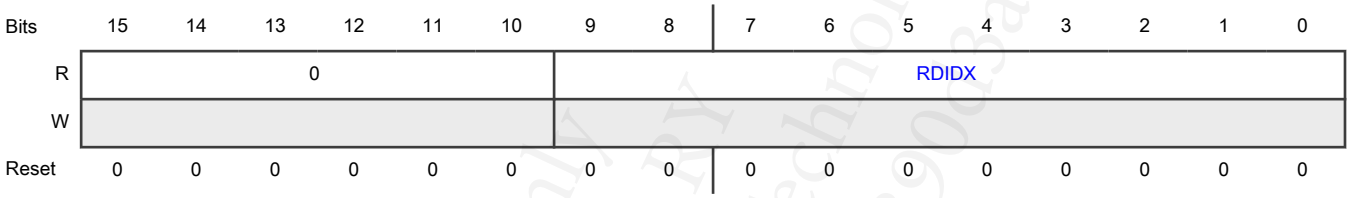
Function

Provides the MB header index of the next available FIFO B entry that the application can read.

NOTE

If the FIFO is empty, the RDIDX field points to a physical MB with invalid content.

Diagram



Fields

Field	Function
15-10 —	Reserved
9-0 RDIDX	Read Index Provides the MB header index of the next available FIFO MB that the application can read.

56.5.60 Receive FIFO Message ID Acceptance Filter Value (RFMIDAFVR)

Offset

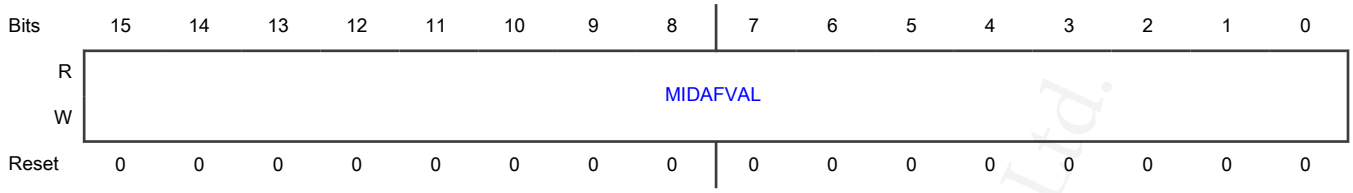
Register	Offset
RFMIDAFVR	90h

Function

Defines the filter value for the message ID acceptance filter of the selected FIFO. For details on message ID filtering, see FIFO Filtering.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e5-4821-4a8-b251-390a3ab624c1

Diagram



Fields

Field	Function
15-0 MIDAFVAL	MIDAFVALA Or MIDAFVALB Message ID Acceptance Filter Value Contains the value for the message ID acceptance filter.

56.5.61 Receive FIFO Message ID Acceptance Filter Mask (RFMIDAFMR)

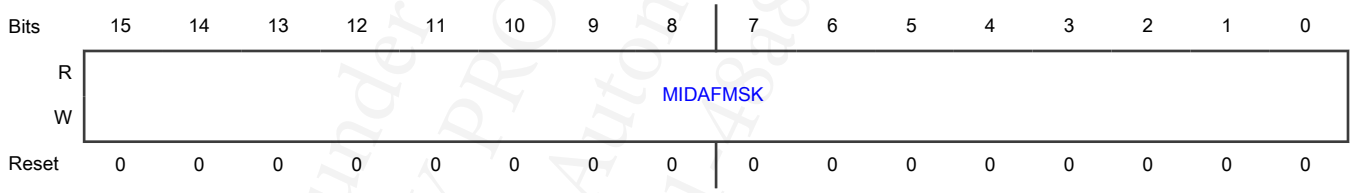
Offset

Register	Offset
RFMIDAFMR	92h

Function

Defines the filter mask for the message ID acceptance filter of the selected FIFO. For details on message ID filtering, see FIFO Filtering.

Diagram



Fields

Field	Function
15-0 MIDAFMSK	MIDAFMSKA Or MIDAFMSKB Message ID Acceptance Filter Mask Contains the mask for the message ID acceptance filter.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462c-4821-8a8-b251-390d3ab624c1

56.5.62 Receive FIFO Frame ID Rejection Filter Value (RFFIDRFVR)

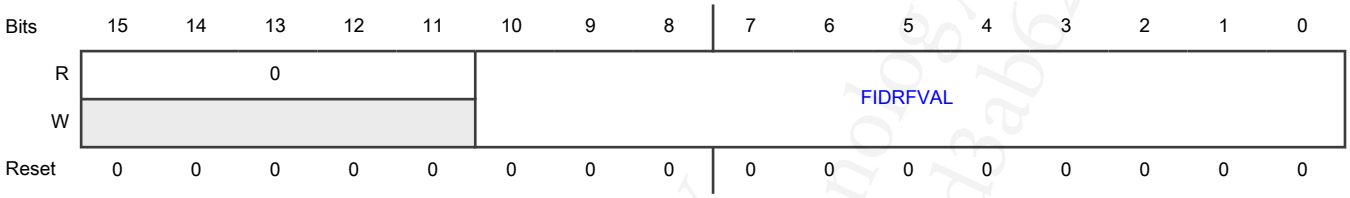
Offset

Register	Offset
RFFIDRFVR	94h

Function

Defines the filter value for the frame ID rejection filter of the selected FIFO. For details on frame ID filtering, see FIFO Filtering.

Diagram



Fields

Field	Function
15-11 —	Reserved
10-0 FIDRFVAL	FIDRFVALA Or FIDRFVALB Frame ID Rejection Filter Value Contains the value for the frame ID rejection filter.

56.5.63 Receive FIFO Frame ID Rejection Filter Mask (RFFIDRFMR)

Offset

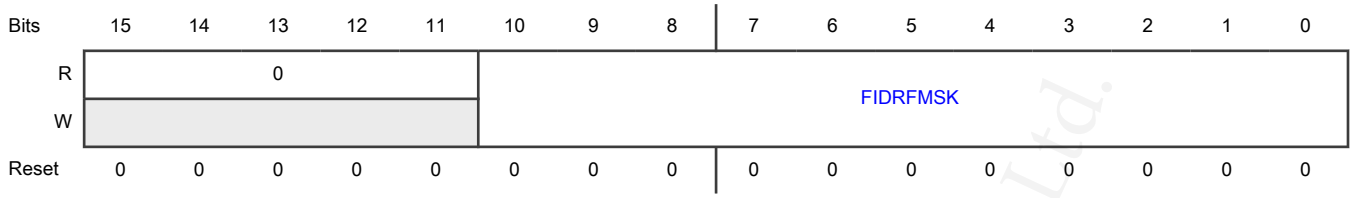
Register	Offset
RFFIDRFMR	96h

Function

Defines the filter mask for the frame ID rejection filter of the selected FIFO. For details on frame ID filtering, see FIFO Filtering.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4888-1251-390d3ab6240d

Diagram



Fields

Field	Function
15-11 —	Reserved
10-0 FIDRFMSK	Frame ID Rejection Filter Mask Contains the mask for the frame ID rejection filter.

56.5.64 Receive FIFO Range Filter Configuration (RFRFCFR)

Offset

Register	Offset
RFRFCFR	98h

Function

Provides access to the four internal frame ID range filter boundary registers of the selected FIFO. For details on frame ID range filter, see FIFO Filtering.

Diagram



Fields

Field	Function
15 WMD	Write Mode Defines the write mode of this register. 0b - Write to all fields in this register on a write access 1b - Write only to the SEL and IBD fields on write access

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c84625-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
14 IBD	Interval Boundary Selects the lower or upper interval boundary to be programmed with the SID value. 0b - Lower 1b - Upper
13-12 SEL	Filter Selector Selects the frame ID range filter to be accessed. 00b - Filter 0 01b - Filter 1 10b - Filter 2 11b - Filter 3
11 —	Reserved
10-0 SID	Slot ID Defines the IBD-selected frame ID boundary value for the SEL-selected range filter.

56.5.65 Receive FIFO Range Filter Control (RFRFCTR)

Offset

Register	Offset
RFRFCTR	9Ah

Function

Used to:

- Enable and disable each frame ID range filter
- Define whether each frame ID range filter is running as acceptance or rejection filter

Provided under NDA only
COMPANY PROPRIETARY
Beijing C&J Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				F3MD	F2MD	F1MD	F0MD	0				F3EN	F2EN	F1EN	F0EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-12 —	Reserved
11 F3MD	Range Filter 3 Mode Defines the filter mode (acceptance or rejection) of frame ID range filter 3. 0b - Acceptance 1b - Rejection
10 F2MD	Range Filter 2 Mode Defines the filter mode (acceptance or rejection) of frame ID range filter 2. 0b - Acceptance 1b - Rejection
9 F1MD	Range Filter 1 Mode Defines the filter mode (acceptance or rejection) of frame ID range filter 1. 0b - Acceptance 1b - Rejection
8 F0MD	Range Filter 0 Mode Defines the filter mode (acceptance or rejection) of frame ID range filter 0. 0b - Acceptance 1b - Rejection
7-4 —	Reserved
3 F3EN	Range Filter 3 Enable Enables and disables frame ID range filter 3. 0b - Disabled 1b - Enabled
2	Range Filter 2 Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
F2EN	Enables and disables frame ID range filter 2. 0b - Disabled 1b - Enabled
1 F1EN	Range Filter 1 Enable Enables and disables frame ID range filter 1. 0b - Disabled 1b - Enabled
0 F0EN	Range Filter 0 Enable Enables and disables frame ID range filter 0. 0b - Disabled 1b - Enabled

56.5.66 Last Dynamic Transmit Slot Channel A (LDTXSLAR)

Offset

Register	Offset
LDTXSLAR	9Ch

Function

Provides the number of the last transmission slot in the dynamic segment for channel A.

This register is updated after the end of the dynamic segment and before the start of the next communication cycle.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					LDYNTXSLOTA										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-11 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-45c8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
10-0 LDYNTXSLOTA	Last Dynamic Transmission Slot Channel A Provides the number of the last transmission slot in the dynamic segment for channel A. If no frame is transmitted during the dynamic segment on channel A, the value of this field becomes 0. Protocol-related variable: zLastDynTxSlot channel A

56.5.67 Last Dynamic Transmit Slot Channel B (LDTXSLBR)

Offset

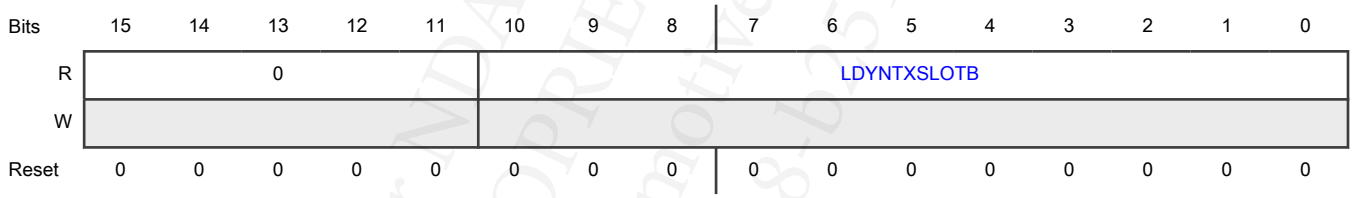
Register	Offset
LDTXSLBR	9Eh

Function

Provides the number of the last transmission slot in the dynamic segment for channel B.

This register is updated after the end of the dynamic segment and before the start of the next communication cycle.

Diagram



Fields

Field	Function
15-11 —	Reserved
10-0 LDYNTXSLOTB	Last Dynamic Transmission Slot Channel B Provides the number of the last transmission slot in the dynamic segment for channel B. If no frame is transmitted during the dynamic segment on channel B, the value of this field becomes 0. Protocol-related variable: zLastDynTxSlot channel B

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46f2-4821-48a8-b651-590d3ab624c1

56.5.68 Protocol Configuration 0 (PCR0)

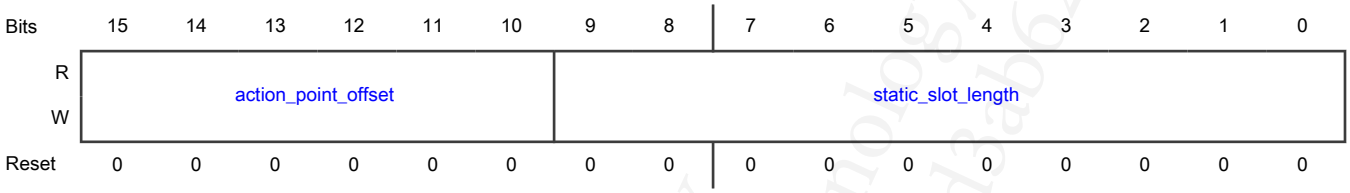
Offset

Register	Offset
PCR0	A0h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-10 action_point_offset	action_point_offset gdActionPointOffset - 1
9-0 static_slot_length	static_slot_length gdStaticSlot

56.5.69 Protocol Configuration 1 (PCR1)

Offset

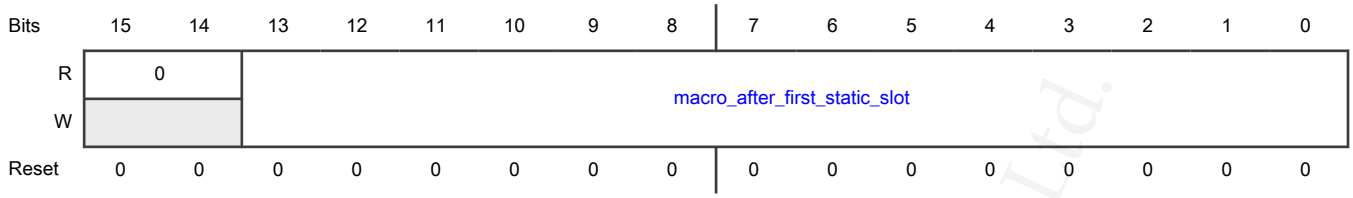
Register	Offset
PCR1	A2h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-b251-390d1ab6240l

Diagram



Fields

Field	Function
15-14 —	Reserved
13-0 macro_after_first_static_slot	macro_after_first_static_slot gMacroPerCycle - gdStaticSlot

56.5.70 Protocol Configuration 2 (PCR2)

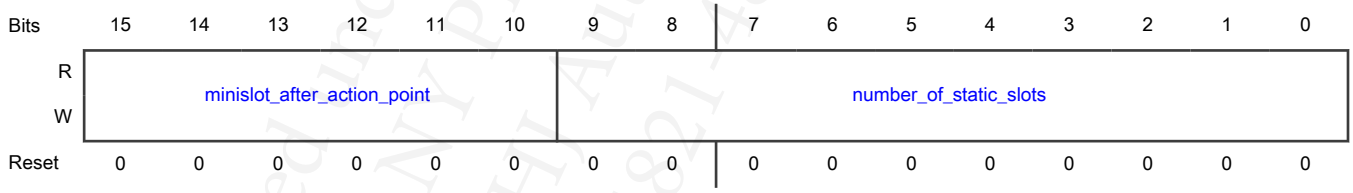
Offset

Register	Offset
PCR2	A4h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-10 minislot_after_action_point	minislot_after_action_point gdMinislot - gdMinislotActionPointOffset - 1
9-0	gNumberOfStaticSlots

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c62e5-4821-48a2-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
number_of_static_slots	

56.5.71 Protocol Configuration 3 (PCR3)

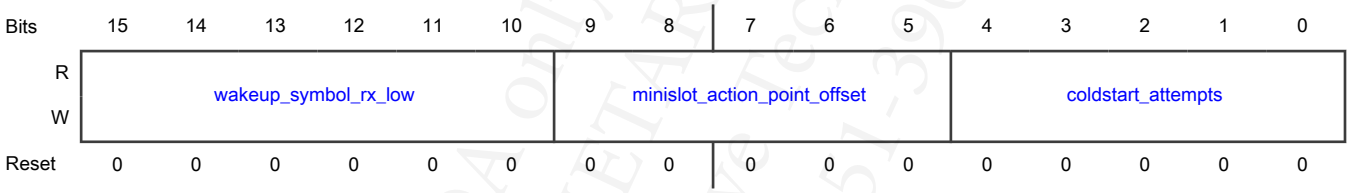
Offset

Register	Offset
PCR3	A6h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-10 wakeup_symbol_rx_low	wakeup_symbol_rx_low gdWakeupSymbolRxLow
9-5 minislot_action_point_offset	minislot_action_point_offset gdMinislotActionPointOffset - 1
4-0 coldstart_attempts	coldstart_attempts gColdstartAttempts

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

56.5.72 Protocol Configuration 4 (PCR4)

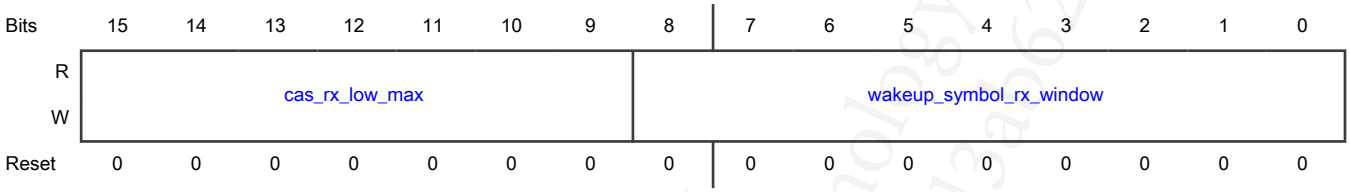
Offset

Register	Offset
PCR4	A8h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-9 cas_rx_low_max	cas_rx_low_max gdCASRxLowMax - 1
8-0 wakeup_symbol_rx_window	wakeup_symbol_rx_window gdWakeupSymbolRxWindow

56.5.73 Protocol Configuration 5 (PCR5)

Offset

Register	Offset
PCR5	AAh

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-b251-390d3a6624d1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	tss_transmitter				wakeup_symbol_tx_low						wakeup_symbol_rx_idle					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-12 tss_transmitter	tss_transmitter gdTSSTransmitter
11-6 wakeup_symbol_tx_low	wakeup_symbol_tx_low gdWakeupSymbolTxLow
5-0 wakeup_symbol_rx_idle	wakeup_symbol_rx_idle gdWakeupSymbolRxIdle

56.5.74 Protocol Configuration 6 (PCR6)

Offset

Register	Offset
PCR6	ACh

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	symbol_window_after_action_point							macro_initial_offset_a							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
14-7 symbol_window_after_action_point	symbol_window_after_action_point gdSymbolWindow - gdActionPointOffset - 1
6-0 macro_initial_offset_a	macro_initial_offset_a pMacroInitialOffset[A]

56.5.75 Protocol Configuration 7 (PCR7)

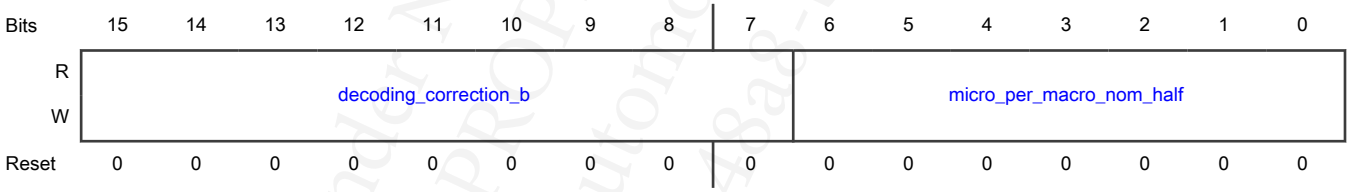
Offset

Register	Offset
PCR7	AEh

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-7 decoding_correction_b	decoding_correction_b pDecodingCorrection + pDelayCompensation[B] + 2
6-0 micro_per_macro_nom_half	micro_per_macro_nom_half round(pMicroPerMacroNom / 2)

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-b251-390d3ab624c1

56.5.76 Protocol Configuration 8 (PCR8)

Offset

Register	Offset
PCR8	B0h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	max_without_clock_correction_fa...				max_without_clock_correction_pa..				wakeup_symbol_tx_idle							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-12 max_without_clock_correction_fatal	max_without_clock_correction_fatal gMaxWithoutClockCorrectionFatal
11-8 max_without_clock_correction_passive	max_without_clock_correction_passive gMaxWithoutClockCorrectionPassive
7-0 wakeup_symbol_tx_idle	wakeup_symbol_tx_idle gdWakeupSymbolTxIdle

56.5.77 Protocol Configuration 9 (PCR9)

Offset

Register	Offset
PCR9	B2h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Provided under NDA only
COMPANY PROPRIETARY
Beijing C-IT Automotive Technology Co., Ltd.
8c462e5-4821-48a8-b251-390d3a6624d1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	minislo...	symbol_...	offset_correction_out													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 minislot_exists	minislot_exists gNumberOfMinislots ≠ 0
14 symbol_window_exists	symbol_window_exists gdSymbolWindow ≠ 0
13-0 offset_correction_out	offset_correction_out pOffsetCorrectionOut

56.5.78 Protocol Configuration 10 (PCR10)

Offset

Register	Offset
PCR10	B4h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	single_...	wakeup_...	macro_per_cycle													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15	single_slot_enabled

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
single_slot_enabled	pSingleSlotEnabled
14 wakeup_channel	wakeup_channel pWakeupChannel
13-0 macro_per_cycle	macro_per_cycle gMacroPerCycle

56.5.79 Protocol Configuration 11 (PCR11)

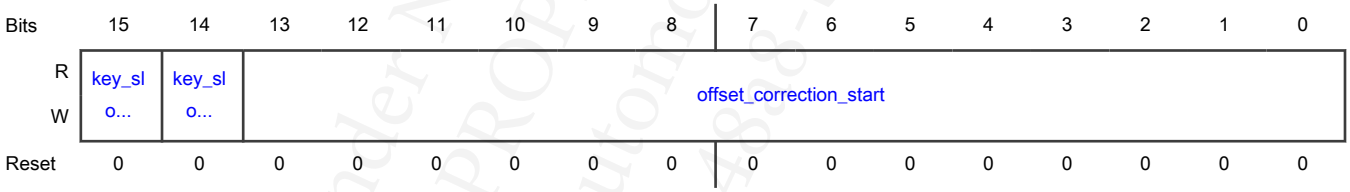
Offset

Register	Offset
PCR11	B6h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15 key_slot_used_for_startup	key_slot_used_for_startup pKeySlotUsedForStartup
14 key_slot_used_for_sync	key_slot_used_for_sync pKeySlotUsedForSync
13-0 offset_correction_start	offset_correction_start gOffsetCorrectionStart

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4888-b251-390d3ab624c1

56.5.80 Protocol Configuration 12 (PCR12)

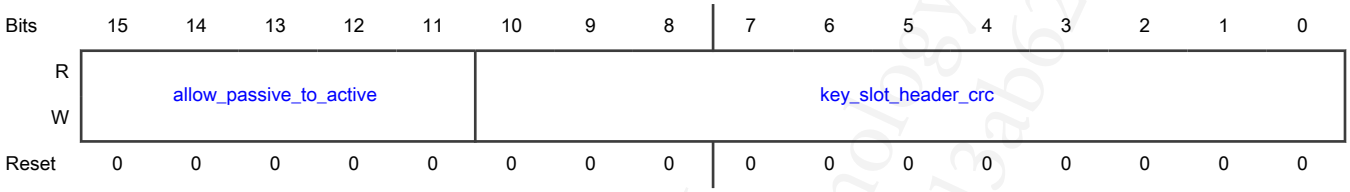
Offset

Register	Offset
PCR12	B8h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-11 allow_passive_to_active	allow_passive_to_active pAllowPassiveToActive
10-0 key_slot_header_crc	key_slot_header_crc header CRC for key slot

56.5.81 Protocol Configuration 13 (PCR13)

Offset

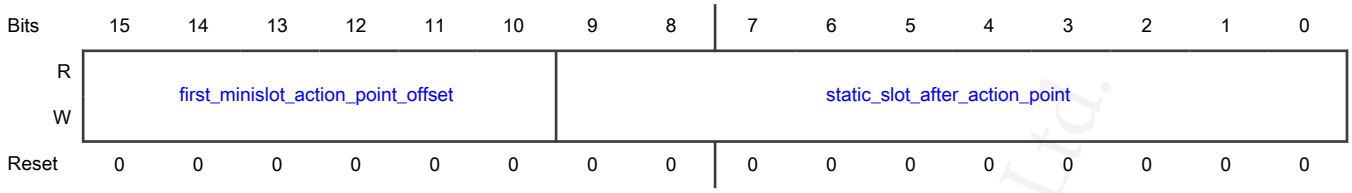
Register	Offset
PCR13	BAh

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab6240d

Diagram



Fields

Field	Function
15-10 first_minislot_action_point_offset	first_minislot_action_point_offset max(gdActionPointOffset, gdMinislotActionPointOffset) - 1
9-0 static_slot_after_action_point	static_slot_after_action_point gdStaticSlot - gdActionPointOffset - 1

56.5.82 Protocol Configuration 14 (PCR14)

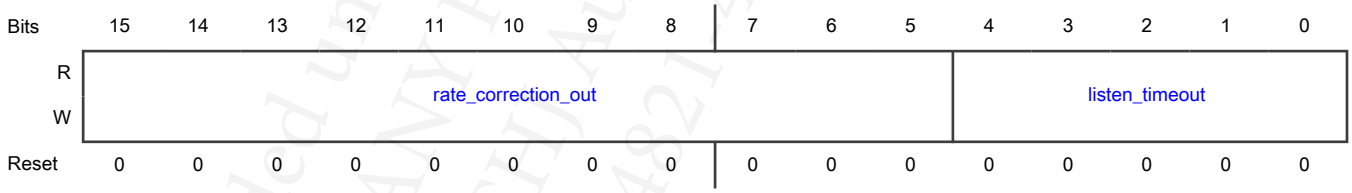
Offset

Register	Offset
PCR14	BCh

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-5 rate_correction_out	rate_correction_out pRateCorrectionOut
4-0 listen_timeout	listen_timeout pdListenTimeout - 1

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

56.5.83 Protocol Configuration 15 (PCR15)

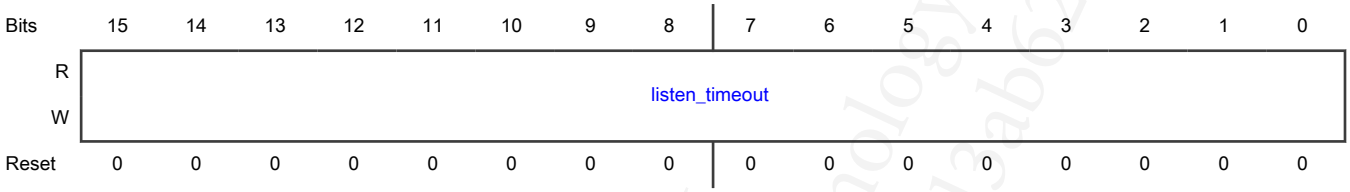
Offset

Register	Offset
PCR15	BEh

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-0 listen_timeout	listen_timeout pdListenTimeout - 1

56.5.84 Protocol Configuration 16 (PCR16)

Offset

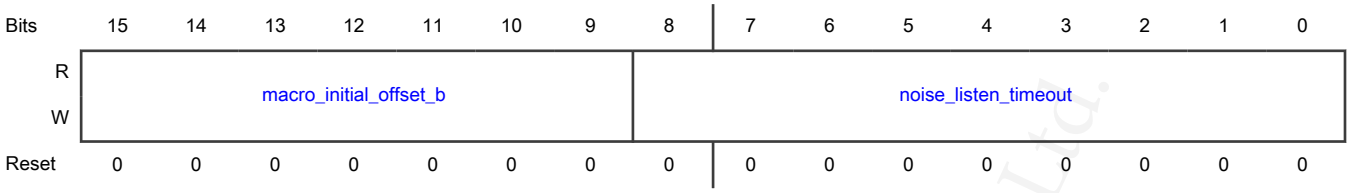
Register	Offset
PCR16	C0h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624d1

Diagram



Fields

Field	Function
15-9 macro_initial_offset_b	macro_initial_offset_b pMacroInitialOffset[B]
8-0 noise_listen_timeout	noise_listen_timeout (gListenNoise × pdListenTimeout) - 1

56.5.85 Protocol Configuration 17 (PCR17)

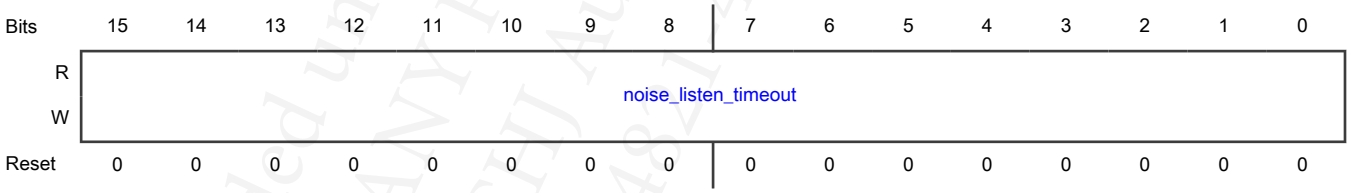
Offset

Register	Offset
PCR17	C2h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-0 noise_listen_timeout	noise_listen_timeout (gListenNoise × pdListenTimeout) - 1

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

56.5.86 Protocol Configuration 18 (PCR18)

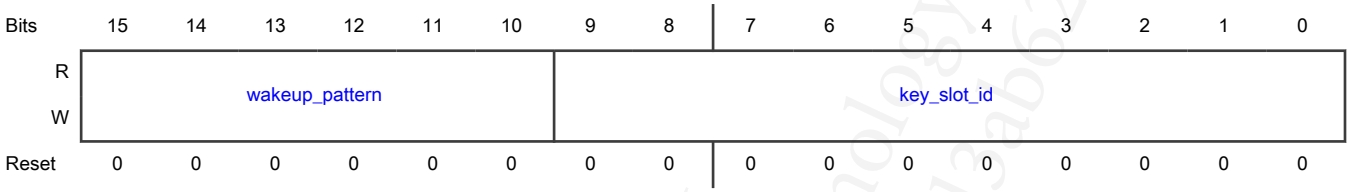
Offset

Register	Offset
PCR18	C4h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-10 wakeup_pattern	wakeup_pattern pWakeupPattern
9-0 key_slot_id	key_slot_id pKeySlotId

56.5.87 Protocol Configuration 19 (PCR19)

Offset

Register	Offset
PCR19	C6h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624d1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	decoding_correction_a								payload_length_static							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-7 decoding_correction_a	decoding_correction_a pDecodingCorrection + pDelayCompensation[A] + 2
6-0 payload_length_static	payload_length_static gPayloadLengthStatic

56.5.88 Protocol Configuration 20 (PCR20)

Offset

Register	Offset
PCR20	C8h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	micro_initial_offset_b								micro_initial_offset_a							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8 micro_initial_offset_b	micro_initial_offset_b pMicroInitialOffset[A]
7-0	micro_initial_offset_a

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
micro_initial_offset_a	pMicroInitialOffset[A]

56.5.89 Protocol Configuration 21 (PCR21)

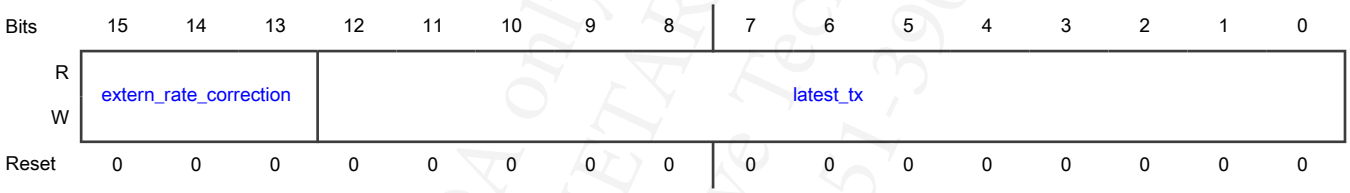
Offset

Register	Offset
PCR21	CAh

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-13 extern_rate_correction	extern_rate_correction pExternRateCorrection
12-0 latest_tx	latest_tx gNumberOfMinislots - pLatestTx

56.5.90 Protocol Configuration 22 (PCR22)

Offset

Register	Offset
PCR22	CCh

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	comp_accepted_startup_range_a											micro_per_cycle			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 —	Reserved bit, will not be changed. Do not write any value different from the reset value.
14-4 comp_accepted_startup_range_a	comp_accepted_startup_range_a pdAcceptedStartupRange - pDelayCompensation[A]
3-0 micro_per_cycle	micro_per_cycle pMicroPerCycle

56.5.91 Protocol Configuration 23 (PCR23)

Offset

Register	Offset
PCR23	CEh

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	micro_per_cycle															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 micro_per_cycle	micro_per_cycle pMicroPerCycle

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624d1

56.5.92 Protocol Configuration 24 (PCR24)

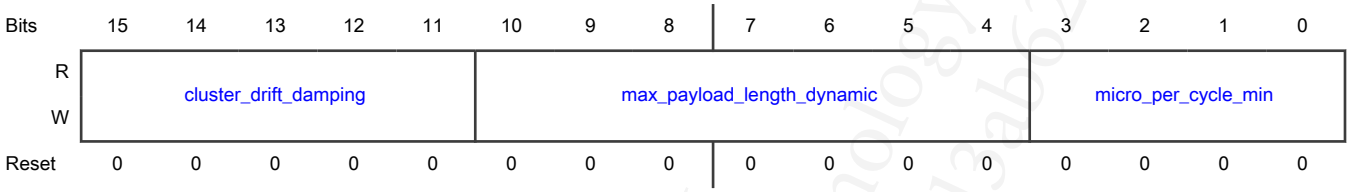
Offset

Register	Offset
PCR24	D0h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-11 cluster_drift_damping	cluster_drift_damping pClusterDriftDamping
10-4 max_payload_length_dynamic	max_payload_length_dynamic pPayloadLengthDynMax
3-0 micro_per_cycle_min	micro_per_cycle_min pMicroPerCycle - pdMaxDrift

56.5.93 Protocol Configuration 25 (PCR25)

Offset

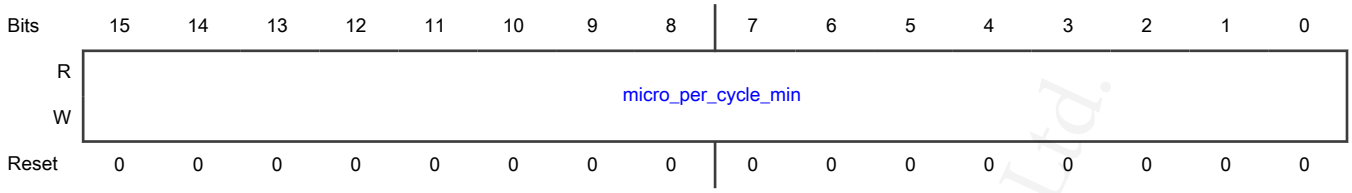
Register	Offset
PCR25	D2h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-390d3ab624d1

Diagram



Fields

Field	Function
15-0 micro_per_cycle_min	micro_per_cycle_min pMicroPerCycle - pdMaxDrift

56.5.94 Protocol Configuration 26 (PCR26)

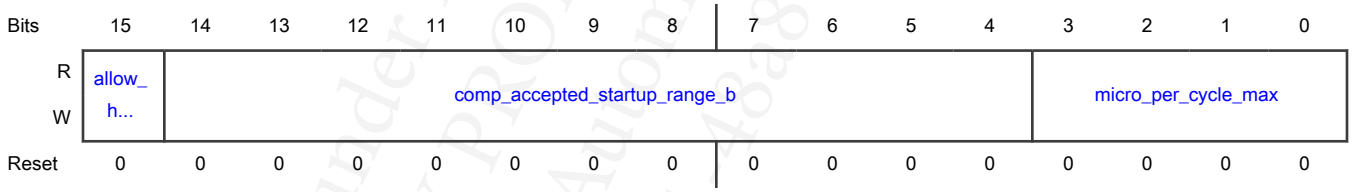
Offset

Register	Offset
PCR26	D4h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15 allow_halt_due_to_clock	allow_halt_due_to_clock pAllowHaltDueToClock
14-4 comp_accepted_startup_range_b	comp_accepted_startup_range_b pdAcceptedStartupRange - pDelayCompensation[B]

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 micro_per_cycle_max	micro_per_cycle_max pMicroPerCycle + pdMaxDrift

56.5.95 Protocol Configuration 27 (PCR27)

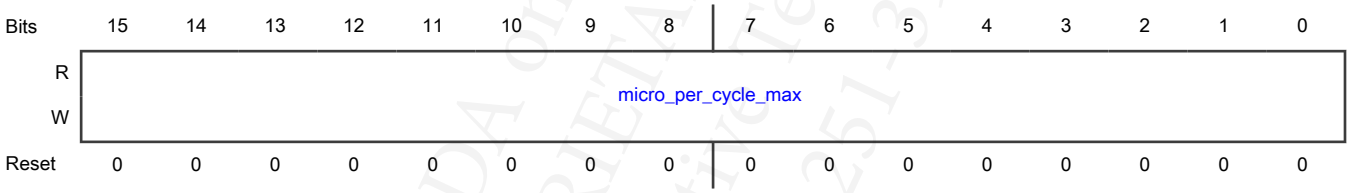
Offset

Register	Offset
PCR27	D6h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-0 micro_per_cycle_max	micro_per_cycle_max pMicroPerCycle + pdMaxDrift

56.5.96 Protocol Configuration 28 (PCR28)

Offset

Register	Offset
PCR28	D8h

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-39013ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	dynamic_slot_id								macro_after_offset_correction							
W	...															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 dynamic_slot_idle_phase	dynamic_slot_idle_phase gdDynamicSlotIdlePhase.
13-0 macro_after_offset_correction	macro_after_offset_correction gMacroPerCycle - gOffsetCorrectionStart

56.5.97 Protocol Configuration 29 (PCR29)

Offset

Register	Offset
PCR29	DAh

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	extern_offset_correction			minislots_max												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 extern_offset_correction	extern_offset_correction pExternOffsetCorrection
12-0 minislots_max	minislots_max gNumberOfMinislots - 1

56.5.98 Protocol Configuration 30 (PCR30)

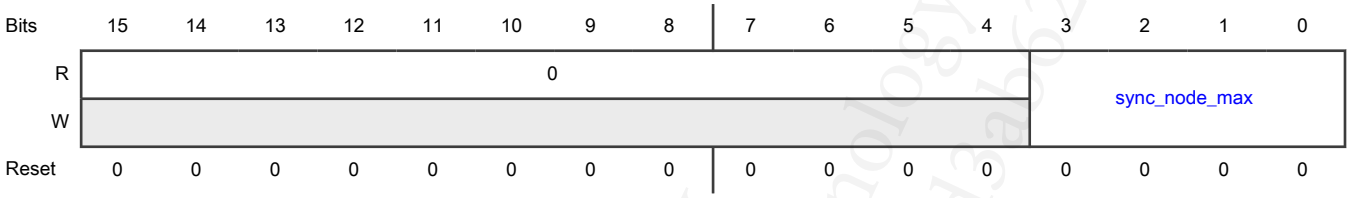
Offset

Register	Offset
PCR30	DCh

Function

Provides the necessary configuration information to PE. The individual values are described in Table 458.

Diagram



Fields

Field	Function
15-4 —	Reserved
3-0 sync_node_max	sync_node_max gSyncNodeMax

56.5.99 Stopwatch Count High (STPWHR)

Offset

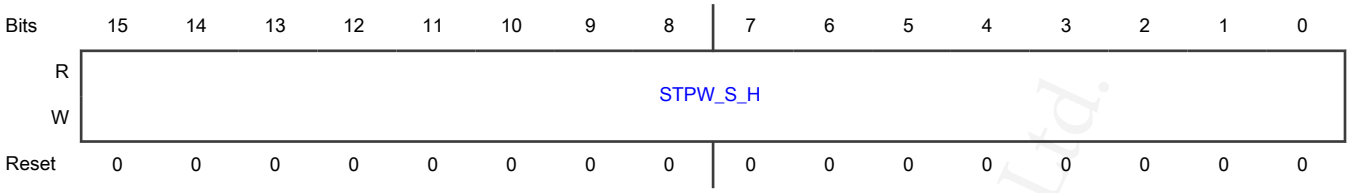
Register	Offset
STPWHR	DEh

Function

Works with Stopwatch Count Low (STPWLR) to form a stopwatch counter that runs on the system clock. This counter starts on cycle start even and stops on an external event “OS_tick”. When all 32 bits of the stopwatch counter are read, the counter automatically clears. For details, see StopWatch function.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462f-4821-48a8-b251-390da6b624d1

Diagram



Fields

Field	Function
15-0 STPW_S_H	Stopwatch Count High Contains the high 16 bits of the stopwatch counter. After a read operation, the field's value clears to 0.

56.5.100 Stopwatch Count Low (STPWLR)

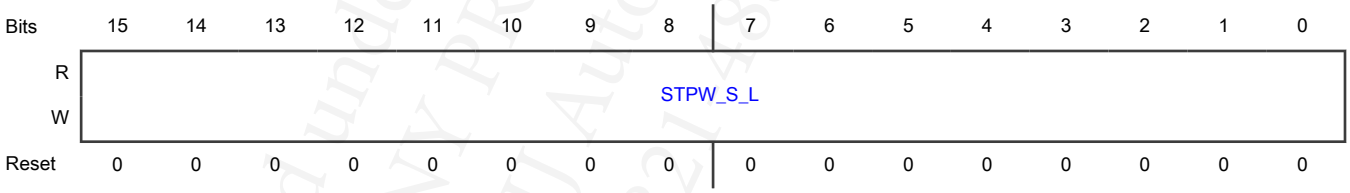
Offset

Register	Offset
STPWLR	E0h

Function

Works with Stopwatch Count High (STPWHR) to form a stopwatch counter that runs on the system clock. This counter starts on cycle start even and stops on an external event “OS_tick”. When all 32 bits of the stopwatch counter are read, the counter automatically clears. For details, see StopWatch function.

Diagram



Fields

Field	Function
15-0 STPW_S_L	Stopwatch Count Low Contains the low 16 bits of the stopwatch counter. After a read operation, the field's value clears to 0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c2e25_48214a8-951-390d3ab624c1

56.5.101 Protocol Event Output Enable And Stopwatch Control (PEOER)

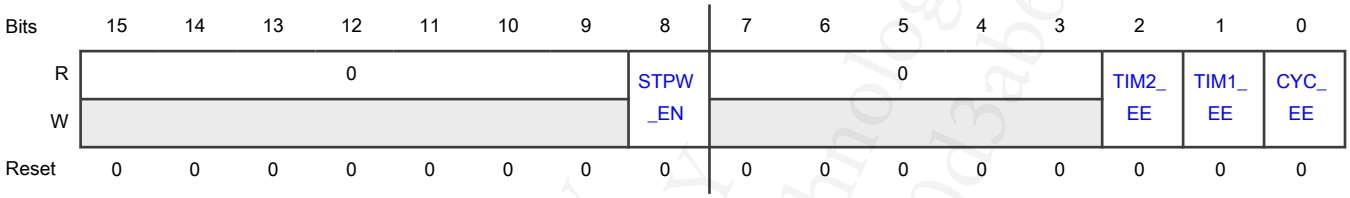
Offset

Register	Offset
PEOER	E2h

Function

Defines whether or not the event output ports are enabled based on the events such as Cycle start and Timer1 and Timer2 expiration (see Timer Support for details).

Diagram



Fields

Field	Function
15-9 —	Reserved
8 STPW_EN	Stopwatch Count Enable Enables and disables the stopwatch counter. 0b - Disabled 1b - Enabled
7-3 —	Reserved
2 TIM2_EE	Timer 2 Expired Event Output Enable Enables and disables the event on the "fr_evt_tim2" port. 0b - Disabled 1b - Enabled
1 TIM1_EE	Timer 1 Expired Event Output Enable Enables and disables the event on the "fr_evt_tim1" port. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 CYC_EE	Cycle Start Event Output Enable Enables and disables the event on the "fr_evt_cyc" port. 0b - Disabled 1b - Enabled

56.5.102 Receive FIFO Start Data Offset (RFSDOR)

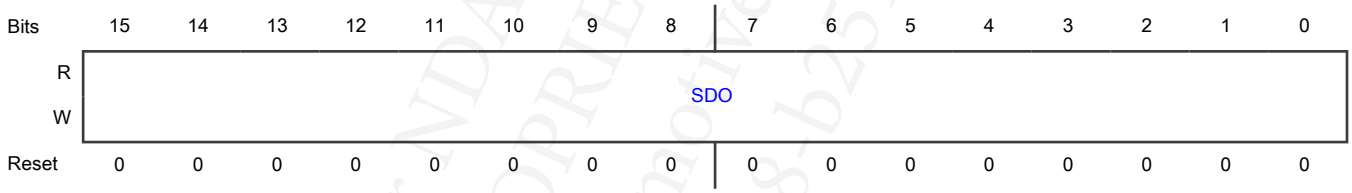
Offset

Register	Offset
RFSDOR	E6h

Function

Because all data fields of the FIFO are of equal length and are located at subsequent system memory addresses, the content of this register corresponds to the start address of payload area of the selected FIFO.

Diagram



Fields

Field	Function
15-0 SDO	SDOA And SDOB Start Data Field Offset Defines the data field offset of the header field of the first MB of the selected FIFO. CC uses the SDO value to determine the physical location of the receiver FIFO's first MB header field. For configuration constraints, see Configure Data Field Offsets.

56.5.103 Receive FIFO System Memory Base Address High (RFSYMBADHR)

Offset

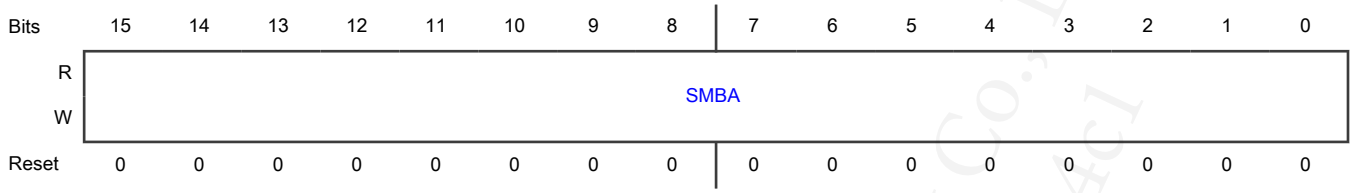
Register	Offset
RFSYMBADHR	E8h

Provided under NDA only
CONFIDENTIAL PROPRIETARY
Beijing C.T Automotive Technology Co., Ltd.
c8c46285-4214-48a8-b251-3900a3ab624c1

Function

Works with Receive FIFO System Memory Base Address Low (RFSYMBADLR) to define the system memory base address for the receive FIFO if MCR[FAM] = 1. BMIF uses the system memory base address to calculate the physical memory address for system memory accesses for the FIFOs.

Diagram



Fields

Field	Function
15-0 SMBA	System Memory Base Address Contains the value of the system memory base address, defined as a byte address.

56.5.104 Receive FIFO System Memory Base Address Low (RFSYMBADLR)

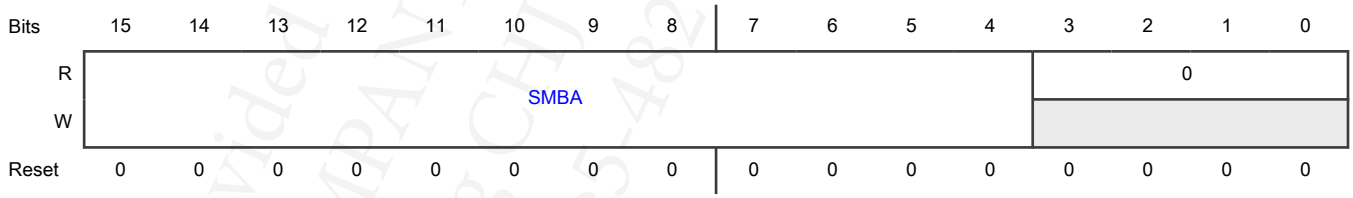
Offset

Register	Offset
RFSYMBADLR	EAh

Function

Works with Receive FIFO System Memory Base Address High (RFSYMBADHR) to define the system memory base address for the receive FIFO if MCR[FAM] = 1. BMIF uses the system memory base address to calculate the physical memory address for system memory accesses for the FIFOs.

Diagram



Fields

Field	Function
15-4 SMBA	System Memory Base Address Contains the value of the system memory base address, defined as a byte address.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 —	Reserved

56.5.105 Receive FIFO Periodic Timer (RFPTR)

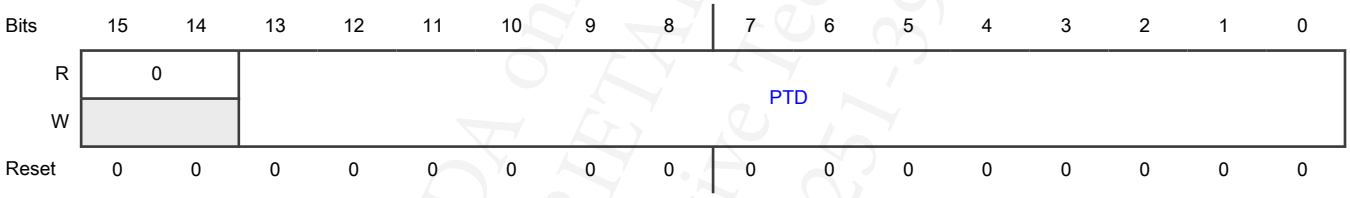
Offset

Register	Offset
RFPTR	ECh

Function

Holds the duration for the periodic FIFO timer. The periodic timer applies to both FIFOs (see FIFO Periodic Timer).

Diagram



Fields

Field	Function
15-14 —	Reserved
13-0 PTD	Periodic Timer Duration Defines the periodic timer duration, in terms of µTs, as follows: 0h - Timer remains expired 1–3FFEh - Timer expires after PTD µTs and restarts at each cycle start 3FFFh - Timer never expires

56.5.106 Receive FIFO Fill Level and Pop Count (RFFLPCR)

Offset

Register	Offset
RFFLPCR	EEh

Function

Provides the current fill level of the two receiver FIFOs and allows you to pop a number of entries from the FIFOs.

If the pop count value (PCA or PCB) is greater than the current FIFO fill level (FLB or FLA), then the FIFO is empty after the update. You do not receive any notification that the required number of entries was not removed.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	FLB_or_PCB								FLA_or_PCA							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-8 FLB_or_PCB	FLB Or PCB Displays the fill level value FLB or configures the pop count value PCB. **NOTE** This field behaves differently for read and write operations. When reading 00000000-11111111b - Fill Level FIFO B (FLB): Provides the current number of entries in FIFO B When writing 00000000-11111111b - Pop Count FIFO B (PCB): Defines the number of entries to be removed from FIFO B
7-0 FLA_or_PCA	FLA Or PCA Displays the fill level value FLA or configures the pop count value PCA. **NOTE** This field behaves differently for read and write operations. When reading 00000000-11111111b - Fill Level FIFO A (FLA): Provides the current number of entries in FIFO A When writing 00000000-11111111b - Pop Count FIFO A (PCA): Defines the number of entries to be removed from FIFO A

56.5.107 ECC Error Interrupt Flag And Enable (EEIFER)

Offset

Register	Offset
EEIFER	F0h

Function

Controls the ECC-related interrupt request lines and provides the corresponding interrupt flags.

You can return the interrupt flags to 0 by writing 1 to them. This resets the corresponding report registers. For a detailed description, see Memory Error Reporting.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LRNE_OF	LRCE_OF	DRNE_OF	DRCE_OF	LRNE_IF	LRCE_IF	DRNE_IF	DRCE_IF	0				LRNE_IE	LRCE_IE	DRNE_IE	DRCE_IE
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 LRNE_OF	LRAM Non-Corrected Error Overflow Flag Becomes 1 when at least one of the following events occurs: • Memory errors are detected but not corrected on CHI LRAM and LRNE_IF is already 1. • Memory errors are detected but not corrected on at least two banks of CHI LRAM. 0b - No such event 1b - Non-corrected error overflow detected on CHI LRAM
14 LRCE_OF	LRAM Corrected Error Overflow Flag Becomes 1 when at least one of the following events occurs: • Memory errors are detected and corrected on CHI LRAM and LRCE_IF is already 1. • Memory errors are detected and corrected on at least two banks of CHI LRAM. **NOTE** Error correction not implemented on CHI LRAM; flag will never be asserted. 0b - No such event 1b - Corrected error overflow detected on CHI LRAM
13 DRNE_OF	DRAM Non-Corrected Error Overflow Flag Becomes 1 when at least one of the following events occurs: • Memory errors are detected but not corrected on PE DRAM and DRNE_IF is already 1. • Memory errors are detected but not corrected on at least two banks of the PE DRAM. 0b - No such event 1b - Non-corrected error overflow detected on PE DRAM
12 DRCE_OF	DRAM Corrected Error Overflow Flag Becomes 1 when at least one of the following events occurs:

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Memory errors are detected and corrected on PE DRAM and DRCE_IF is already 1. • Memory errors are detected and corrected on at least two banks of PE DRAM. 0b - No such event 1b - Corrected error overflow detected on PE DRAM
11 LRNE_IF	LRAM Non-Corrected Error Interrupt Flag Becomes 1 when a memory error is detected but not corrected on CHI LRAM. 0b - No such event 1b - Non-corrected error detected on CHI LRAM
10 LRCE_IF	LRAM Corrected Error Interrupt Flag Becomes 1 when a memory error is detected and corrected on CHI LRAM. **NOTE** Error correction not implemented on CHI LRAM; flag will never be asserted. 0b - No such event 1b - Corrected error detected on CHI LRAM
9 DRNE_IF	DRAM Non-Corrected Error Interrupt Flag Becomes 1 when a memory error is detected but not corrected on PE DRAM. 0b - No such event 1b - Non-corrected error detected on PE DRAM
8 DRCE_IF	DRAM Corrected Error Interrupt Flag Becomes 1 when a memory error is detected and corrected on PE DRAM. 0b - No such event 1b - Corrected error detected on PE DRAM
7-4 —	Reserved
3 LRNE_IE	LRAM Non-Corrected Error Interrupt Enable Enables and disables the assertion of the LRAM non-corrected error interrupt line when LRNE_IF = 1. 0b - Disable 1b - Enable
2 LRCE_IE	LRAM Corrected Error Interrupt Enable Enables and disables the assertion of the LRAM corrected error interrupt line when LRCE_IF = 1.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disable 1b - Enable
1 DRNE_IE	DRAM Non-Corrected Error Interrupt Enable Enables and disables the assertion of the DRAM non-corrected error interrupt line when DRNE_IF = 1. 0b - Disable 1b - Enable
0 DRCE_IE	DRAM Corrected Error Interrupt Enable Enables and disables the assertion of the DRAM corrected error interrupt line when DRCE_IF = 1. 0b - Disable 1b - Enable

56.5.108 ECC Error Report And Injection Control (EERICR)

Offset

Register	Offset
EERICR	F2h

Function

Configures the error injection and error reporting and provides the selector for the content of the report registers.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BSY	0					ERS		0			ERM	0		EIM	EIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 BSY	Register Update Busy Indicates the current state of the ECC configuration update and controls the register write access condition IDL specified in Register write access. 0b - Idle

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Running
14-10 —	Reserved
9-8 ERS	Error Report Select Selects the content of the ECC error reporting registers. 00b - Show PE DRAM noncorrected error information. 01b - Show PE DRAM corrected error information. 10b - Show CHI LRAM noncorrected error information. 11b - Show CHI LRAM corrected error information.
7-5 —	Reserved
4 ERM	Error Report Mode Configures the type of data written into the internal error report registers on the detection of a memory error. 0b - Store data and code as delivered by ECC decoding logic. 1b - Store data and code as read from the memory.
3-2 —	Reserved
1 EIM	Error Injection Mode Configures the ECC error injection mode. 0b - Use EEIDR[DATA] and EEICR[CODE] as XOR distortion pattern for error injection. 1b - Use EEIDR[DATA] and EEICR[CODE] as write value for error injection.
0 EIE	Error Injection Enable Enables and disables the ECC error injection on the memories. 0b - Disabled 1b - Enabled

56.5.109 ECC Error Report Address (EERAR)

Offset

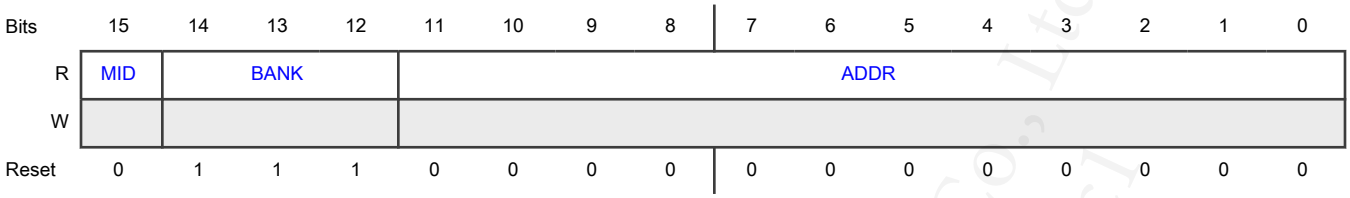
Register	Offset
EERAR	F4h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-b251-390d3ab624c1

Function

Provides the memory identifier, bank, and address for which the memory error is reported.

Diagram



Fields

Field	Function
15 MID	Memory Identifier Provides the memory instance for which the memory error is reported. 0b - PE DRAM 1b - CHI LRAM
14-12 BANK	Memory Bank Provides the BANK for which the memory error is reported. BANK = 111b indicates that no error was found after reset. When MID = 0, BANK values indicate the following: 000b - PE DRAM [7:0] 001b - PE DRAM [15:8] 010b–111b - Not used When MID = 1, see the following table for the assignment of the LRAM banks.

BANK	Register		
000	MBCCFR(2n)	MBDOR(6n)	LEETR0
001	MBFIDR(2n)	MBDOR(6n + 1)	LEETR1
010	MBIDXR(2n)	MBDOR(6n + 2)	LEETR2
011	MBCCFR(2n+1)	MBDOR(6n + 3)	LEETR3
100	MBFIDR(2n+1)	MBDOR(6n + 4)	LEETR4
101	MBIDXR(2n+1)	MBDOR(6n + 5)	LEETR5
110	Not used	Not used	Not used
111			

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-48a1-48c8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
11-0 ADDR	Memory Address Provides the address of the failing memory location.

56.5.110 ECC Error Report Data (EERDR)

Offset

Register	Offset
EERDR	F6h

Function

Provides the data-related information of the reported memory read access. The bit assignment depends on the selected memory and memory bank as shown in Table 466.

Table 466. Valid bits in the EERDR[DATA] and EEIDR[DATA] fields

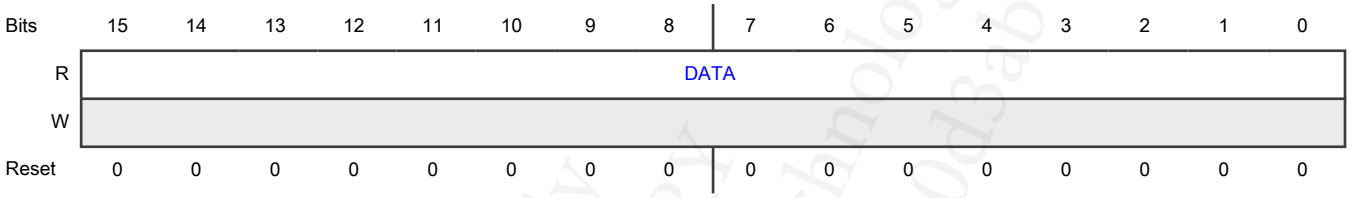
MEM	BANK	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PE DRAM	0									PE DRAM[7:0]							
PE DRAM	1									PE DRAM[15:8]							
CHI LRAM	0	MBCCFR(2n)															
CHI LRAM	0	MBDOR(6n)															
CHI LRAM	0	LEETR0															
CHI LRAM	1						MBFIDR(2n)[FID]										
CHI LRAM	1	MBDOR(6n+1)															
CHI LRAM	1	LEETR1															
CHI LRAM	2								MBIDXR(2n)[MBIDX]								
CHI LRAM	2	LEETR2															
CHI LRAM	3	MBCCFR(2n+1)															
CHI LRAM	3	MBDOR(6n+3)															
CHI LRAM	3	LEETR3															
CHI LRAM	4						MBFIDR(2n+1)[FID]										

Table continues on the next page...

Table 466. Valid bits in the EERDR[DATA] and EEIDR[DATA] fields (continued)

MEM	BANK	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHI LRAM	4	MBDOR(6n+4)															
CHI LRAM	4	LEETR4															
CHI LRAM	5								MBIDXR(2n+1)[MBIDX]								
CHI LRAM	5	MBDOR(6n+5)															
CHI LRAM	5	LEETR5															

Diagram



Fields

Field	Function
15-0 DATA	Data The content of this field depends on the report mode selected by EERICR[ERM]: • When ERM = 0, DATA represents ECC data as generated by the ECC decoding logic. • When ERM = 1, DATA represents data as read from the memory.

56.5.111 ECC Error Report Code (EERCR)

Offset

Register	Offset
EERCR	F8h

Function

Provides ECC-related information of the reported memory read access.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d8a6624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											CODE				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-5 —	Reserved
4-0 CODE	Code The content of this field depends on the report mode selected by EERICR[ERM]: • When ERM = 0, CODE represents the ECC syndrome generated by the ECC decoding logic. — PE DRAM Syndrome shows the associated encoding for the PE DRAM syndrome. — CHI LRAM Syndrome shows the associated encoding for the CHI LRAM syndrome. • When ERM = 1, CODE represents the ECC checkbits read from the memory.

56.5.112 ECC Error Injection Address Register (EEIAR)

Offset

Register	Offset
EEIAR	FAh

Function

Defines the memory module, bank, and address where the ECC error must be injected.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MID			BANK					ADDR							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15	Memory Identifier

Table continues on the next page...

Table continued from the previous page...

Field	Function
MID	Defines the memory instance for ECC error injection. 0b - PE DRAM 1b - CHI LRAM
14-12 BANK	Memory Bank This field defines the memory bank for ECC error injection. When MID = 0, BANK values indicate the following: 000b - PE DRAM [7:0] 001b - PE DRAM [15:8] 010b–111b - Reserved When MID = 1, see ECC Error Report Address (EERAR) for the assignment of the LRAM banks.
11-0 ADDR	Memory Address Defines the memory address for ECC error injection.

56.5.113 ECC Error Injection Data (EEIDR)

Offset

Register	Offset
EEIDR	FCh

Function

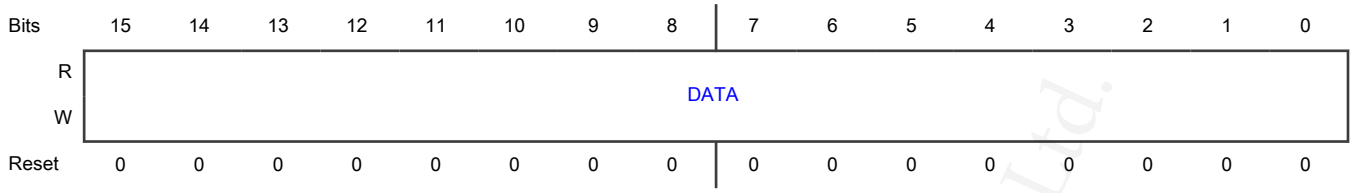
Defines the data distortion pattern for the error injection write. The number of valid bits depends on the selected memory and memory bank as shown in ECC Error Report Data (EERDR).

NOTE

The effect of the error injected depends from the LRAM content at the address accessed and from the module internal usage of the data. See Memory Error Response for details.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-48a2-48a8-b251-390d2ab624c1

Diagram



Fields

Field	Function
15-0 DATA	Data The content of this field depends on the error injection mode selected by EERICR[ERM]: • When EIM = 0, DATA defines the XOR distortion pattern for the data written into the memory. • When EIM = 1, DATA defines the data to be written into the memory.

56.5.114 ECC Error Injection Code (EEICR)

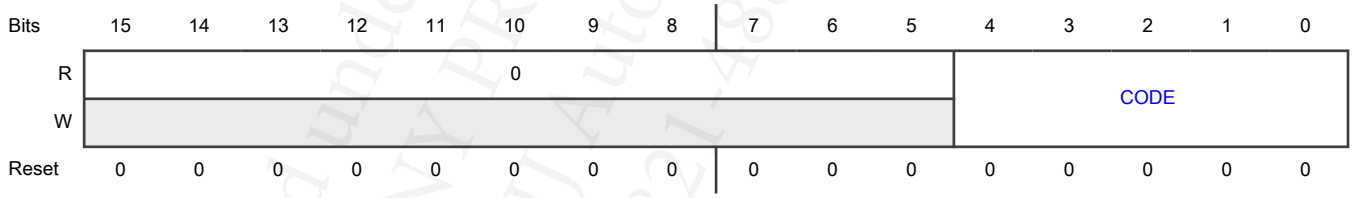
Offset

Register	Offset
EEICR	FEh

Function

Defines the ECC code distortion pattern for the error injection write.

Diagram



Fields

Field	Function
15-5 —	Reserved
4-0 CODE	Code The content of this field depends on the error injection mode selected by EERICR[ERM]:

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3a1624d1

Table continued from the previous page...

Field	Function
	• When EIM = 0, CODE defines the XOR distortion pattern for the ECC checkbits written into the memory. • When EIM = 1, CODE defines the ECC checkbits written into the memory.

56.5.115 MB Configuration Control Status (MBCCSR0 - MBCCSR255)

Offset

For n = 0 to 255:

Register	Offset
MBCCSRn	800h + (n × 8h)

Function

Contains:

- MB configuration data
- MB control data
- MB status information
- MB interrupt flags

A detailed description of all flags can be found in Individual MB functional description.

If you write 1 to EDT, no write access to the other register fields is performed.

If you write 0 to EDT and 1 to LCKT, no write access to the other fields is performed.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			MTD	CMT		0	MBIE	0			DUP	DVAL	EDS	LCKS	MBIF
W						EDT	LCKT									W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-13 —	Reserved
12 MTD	MB Transfer Direction Configures the MB transfer direction.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive MB 1b - Transmit MB
11 CMT	Commit To Transmission Indicates whether the transmit MB data is ready for transmission. **NOTE** This field is read/write but may be modified by hardware other than by a reset. 0b - Not ready 1b - Ready
10 EDT	Enable And Disable Trigger If you write 1 to this field, a MB enable or disable is triggered, depending on the current value of EDS. 0b - No effect 1b - MB enable or disable is triggered
9 LCKT	Lock And Unlock Trigger If you write 1 to this field and write 0 to EDT bit, a MB lock or unlock is triggered, depending on the current LCKS value. 0b - No effect 1b - MB lock or unlock is triggered
8 MBIE	MB Interrupt Enable Defines whether the MB generates an interrupt request when the MB's MBIF field is 1. 0b - Disabled 1b - Enabled
7-5 —	Reserved
4 DUP	Data Updated Indicates whether the frame header in the MB header field and the data in the MB data field are updated after a frame reception. 0b - Not updated 1b - Updated
3 DVAL	Data Valid For receive MBs, indicates whether the MB data field contains valid frame data. For transmit MBs, indicates if a message is transferred again because of the state transmission mode of the MB.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Receive MB contains no valid frame data, or message is transmitted for the first time 1b - Receive MB contains valid frame data, or message will be transferred again
2 EDS	Enable And Disable Status Indicates whether the MB is enabled or disabled. 0b - Disabled 1b - Enabled
1 LCKS	Lock Status Indicates the current lock status of the MB. 0b - You have not locked the MB 1b - You have locked the MB
0 MBIF	MB Interrupt Flag Becomes 1 when the MB slot status field is updated after frame transmission or reception, or when you just enabled a transmit MB. 0b - No such event 1b - Slot status field updated or transmit MB just enabled

56.5.116 MB Cycle Counter Filter (MBCCFR0 - MBCCFR255)

Offset

For n = 0 to 255:

Register	Offset
MBCCFRn	802h + (n × 8h)

Function

Contains MB configuration data for the:

- Transmission mode
- Channel assignment
- Cycle counter filtering

For detailed information on cycle counter filtering, see MB cycle counter filtering.

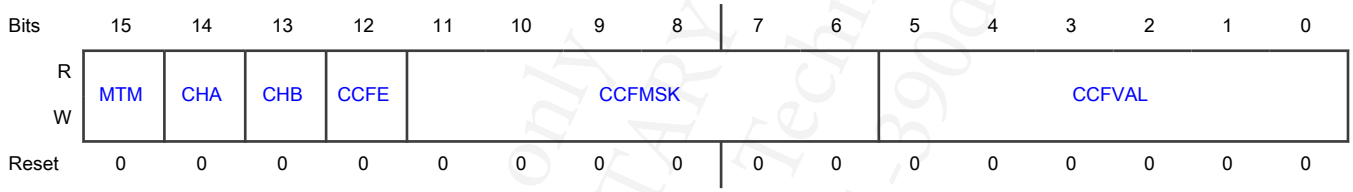
NOTE

- If at least one MB assigned to a certain slot is assigned to both channels, then all MBs assigned to that slot must be assigned to both channels. Otherwise, the MB configuration is illegal and the result of the MB search is not defined.
- After Phase3 reset, LRAM is not initialized. LRAM is initialized when CC leaves Disabled mode. For details, see CHI LRAM Initialization.

Table 467. Channel assignment description

CHA	CHB	Transmit MB		Receive MB	
		Static segment	Dynamic segment	Static segment	Dynamic segment
1	1	Transmit on both channel A and channel B	Transmit on channel A only	Store first valid frame received on either channel A or channel B	Store first valid frame received on channel A; ignore channel B
0	1	Transmit on channel B	Transmit on channel B	Store first valid frame received on channel B	Store first valid frame received on channel B
1	0	Transmit on channel A	Transmit on channel A	Store first valid frame received on channel A	Store first valid frame received on channel A
0	0	No frame transmission	No frame transmission	No frame stored	No frame stored

Diagram



Fields

Field	Function
15 MTM	MB Transmission Mode Applies only to transmit MBs and defines the transmission mode. 0b - Event 1b - State
14 CHA	Channel Assignment A In conjunction with the CHB field, defines the channel assignment and controls the receive and transmit behavior of the MB according to MB Cycle Counter Filter Register.
13 CHB	Channel Assignment B In conjunction with the CHB field, defines the channel assignment and controls the receive and transmit behavior of the MB according to MB Cycle Counter Filter Register.
12 CCFE	Cycle Counter Filtering Enable Enables and disables the cycle counter filtering. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
11-6 CCFMSK	Cycle Counter Filtering Mask Defines the filter mask for the cycle counter filtering.
5-0 CCFVAL	Cycle Counter Filtering Value Defines the filter value for the cycle counter filtering.

56.5.117 MB Frame ID (MBFIDR0 - MBFIDR255)

Offset

For n = 0 to 255:

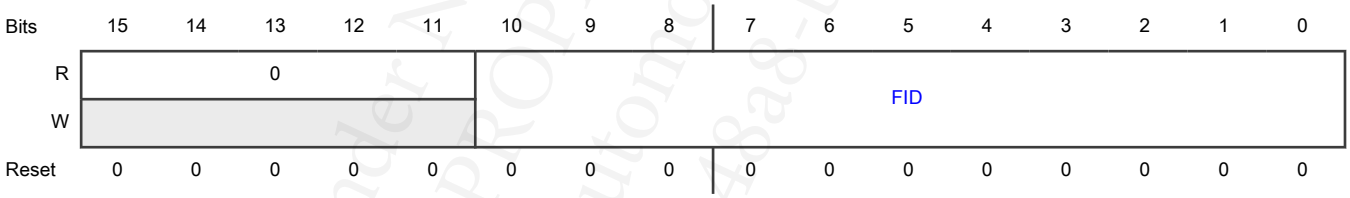
Register	Offset
MBFIDRn	804h + (n × 8h)

Function

NOTE

After Phase3 reset, LRAM is not initialized. LRAM is initialized when CC leaves Disabled mode. For details, see CHI LRAM Initialization.

Diagram



Fields

Field	Function
15-11 —	Reserved
10-0 FID	Frame ID The meaning of this field depends on the MB transfer type. • For a receive MB, this field is used as a filter value to determine if the MB is used for reception of a message received in a slot with the slot ID equal to FID. • For a transmit MB, this field is used to determine the slot in which the message in this MB must be transmitted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca0c2e5-4821-48a8-b251-390d3ab624c1

56.5.118 MB Index (MBIDXR0 - MBIDXR255)

Offset

For n = 0 to 255:

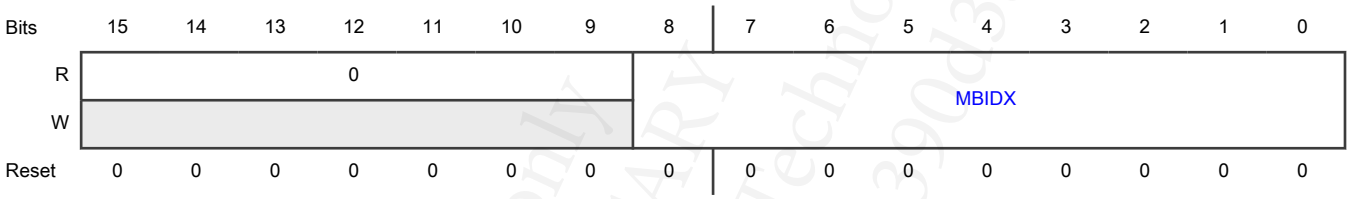
Register	Offset
MBIDXRn	806h + (n × 8h)

Function

NOTE

After Phase3 reset, LRAM is not initialized. LRAM is initialized when CC leaves Disabled mode. For details, see CHI LRAM Initialization.

Diagram



Fields

Field	Function
15-9 —	Reserved
8-0 MBIDX	MB Index Provides the index of the MB header field of the physical MB that is currently associated with this MB. You write the index of the initially associated MB header field into this register. CC updates this register after frame reception or transmission. Legal values are 0 ≤ i ≤ 259. Illegal values are detected during the MB search.

56.5.119 MB Data Field Offset (MBDOR0 - MBDOR259)

Offset

For n = 0 to 259:

Register	Offset
MBDORn	1000h + (n × 2h)

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-8a8-b251-390d3ab624c1

Function

NOTE

After Phase3 reset, LRAM is not initialized. LRAM is initialized when CC leaves Disabled mode. For details, see CHI LRAM Initialization.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MBDO															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 MBDO	MB Data Field Offset Provides the data field offset belonging to a particular MB index. For configuration constraints, see Configure Data Field Offsets.

56.5.120 LRAM ECC Error Test Register (LEETR0 - LEETR5)

Offset

Register	Offset
LEETR0	1210h
LEETR1	1212h
LEETR2	1214h
LEETR3	1216h
LEETR4	1218h
LEETR5	121Ah

Function

NOTE

After Phase3 reset, LRAM is not initialized. LRAM is initialized when CC leaves Disabled mode. For details, see CHI LRAM Initialization.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LEETD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-0 LEETD	LRAM ECC Error Test Data Contains the LRAM data belonging to the test register located in LRAM bank *n*.

56.6 Functional description

This section provides a detailed description of the implemented CC functionality.

56.6.1 MB concept

The CC uses MBs—data structures that store frame data, configuration, control, and status data. Each MB consists of two parts—the MB control data and the physical MB. The MB control data is located in dedicated registers. The structure of the MB control data depends on the MB type and is described in MB types. The physical MB is located in the FlexRay memory area and is described in the next section.

56.6.2 Physical MB

All FlexRay messages and related frame and slot status information of received frames and of frames to be transmitted to the FlexRay bus are stored in data structures called physical MBs. The physical MBs are located in the FlexRay memory area. The following figure shows the structure of a physical MB.

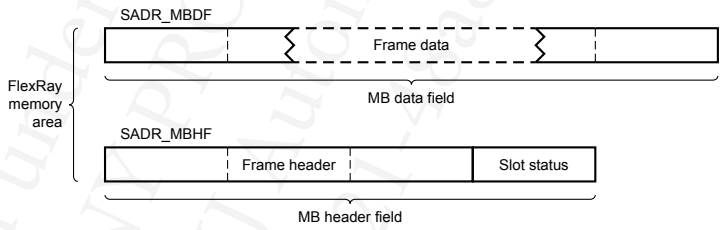


Figure 473. Physical MB structure

A physical MB consists of two fields—the MB header field and the MB data field. The MB header field contains the frame header and the slot status. The MB data field contains the frame data.

The connection between the two fields is established by the data field offset.

56.6.2.1 MB header field

The MB header field is a contiguous region in the FlexRay memory area and occupies eight bytes. It contains the frame header and the slot status. Its structure is shown in Figure 473.

The physical start address SADR_MBHF of the MB header field must be 16-bit aligned.

56.6.2.1.1 Frame header

The frame header occupies the first six bytes in the MB header field. It contains all information related to the FlexRay frame header, according to FlexRay Communications System Protocol Specification, Version 2.1 Rev A.

A detailed description of the usage and the content of the frame header is provided in Frame header description.

56.6.2.1.2 Slot status

The slot status occupies the last two bytes of the MB header field. It provides the slot and frame status related information according to FlexRay Communications System Protocol Specification, Version 2.1 Rev A.

A detailed description of the content and usage of the slot status is provided in Slot status description.

56.6.2.2 MB data field

The MB data field is a contiguous area of 2-byte entities. This field contains all or part of the frame payload data of the frame to be transmitted or received on the FlexRay bus. The minimum length of this field depends on the specific MB configuration and is specified in the MB descriptions given in the following section.

56.6.3 MB types

CC provides three different MB types:

- Individual MBs
- Receive shadow buffers
- Receive FIFO buffers

For each MB type, the structure of the physical MB is identical. The MB types differ only in the structure and content of MB control data, which control the related physical MB. The MB control data is described in the following sections.

56.6.3.1 Individual MBs

The individual MBs are used for all types of frame transmission and for dedicated frame reception based on individual filter settings for each MB. CC supports three types of individual MBs, which are described in Individual MB functional description.

Each individual MB consists of two parts:

- The physical MB, located in the FlexRay memory area
- The MB control data, located in dedicated registers

This figure shows the structure of an individual MB.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-8253-390d3ab624c1

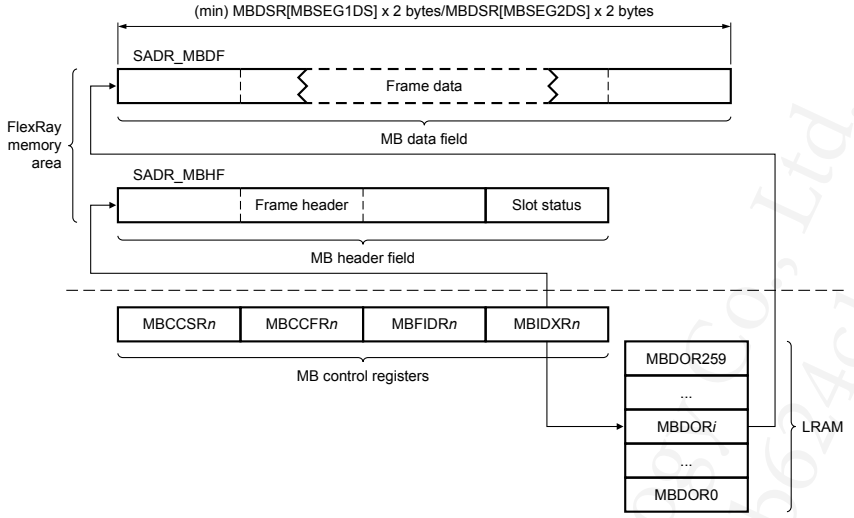


Figure 474. Individual MB structure

Each MB has an MB number *n* assigned. This number determines the set of MB control registers associated with this individual MB. The following registers control the individual MB with number *n*:

- MB Configuration Control Status (MBCCSR0 - MBCCSR255)
- MB Cycle Counter Filter (MBCCFR0 - MBCCFR255)
- MB Frame ID (MBFIDR0 - MBFIDR255)
- MB Index (MBIDXR0 - MBIDXR255)

The MBIDX field in MB Index (MBIDXR0 - MBIDXR255) establishes the connection between the MB control registers and the physical MB. The start address SADR_MBHF of the related MB header field in the FlexRay memory area is determined according to Equation 1.

$$SADR\_MBHF = (MBIDXRn[MBIDX] \times 8) + SMBA$$

Equation 1. Equation for SADR_MBHF

The data field belonging to a particular physical MB is characterized by the data field offset. For each physical MB with MBIDX *i*, MBDOR*i* contains the offset of the corresponding MB data field with respect to the CC FlexRay memory area base address as provided by the SMBA field in the System Memory Base Address Register (SYMBADR).

The data field offset is used to determine the start address SADR_MBDF of the corresponding MB data field in the FlexRay memory area according to Equation 2.

$$SADR\_MBDF = [DataFieldOffset] + SMBA$$

Equation 2. Equation for SADR_MBDF

MBDOR*n* are stored in the module internal memory LRAM. See CHI LRAM Initialization for the setup of the data field offset values.

56.6.3.1.1 Individual MB segments

The set of individual MBs can be split in two MB segments using MB Segment Size and Utilization (MBSSUTR). All individual MBs with a MB number n ≤ MBSSUTR[LAST_MB_SEG1] belong to the first MB segment. All individual MBs with a MB number n > MBSSUTR[LAST_MB_SEG1] belong to the second MB segment. The following rules apply to the length of the MB data field:

- All physical MBs associated with individual MBs that belong to the same MB segment must have MB data fields of the same length.
- The minimum length of the MB data field for individual MBs in the first MB segment is 2 × MBDSR[MBSEG1DS] bytes.

- The minimum length of the MB data field for individual MBs in the second segment is 2 × MBDSR[MBSEG2DS] bytes.

56.6.3.2 Receive shadow buffers

The receive shadow buffers are required for the frame reception process for individual MBs. CC provides four receive shadow buffers, one receive shadow buffer per channel and per MB segment.

Each receive shadow buffer consists of two parts—the physical MB located in the FlexRay memory area and the receive shadow buffer control registers located in dedicated registers. The structure of a receive shadow buffer is shown in the figure below. The four internal shadow buffer control registers can be accessed by Receive Shadow Buffer Index (RSBIR).

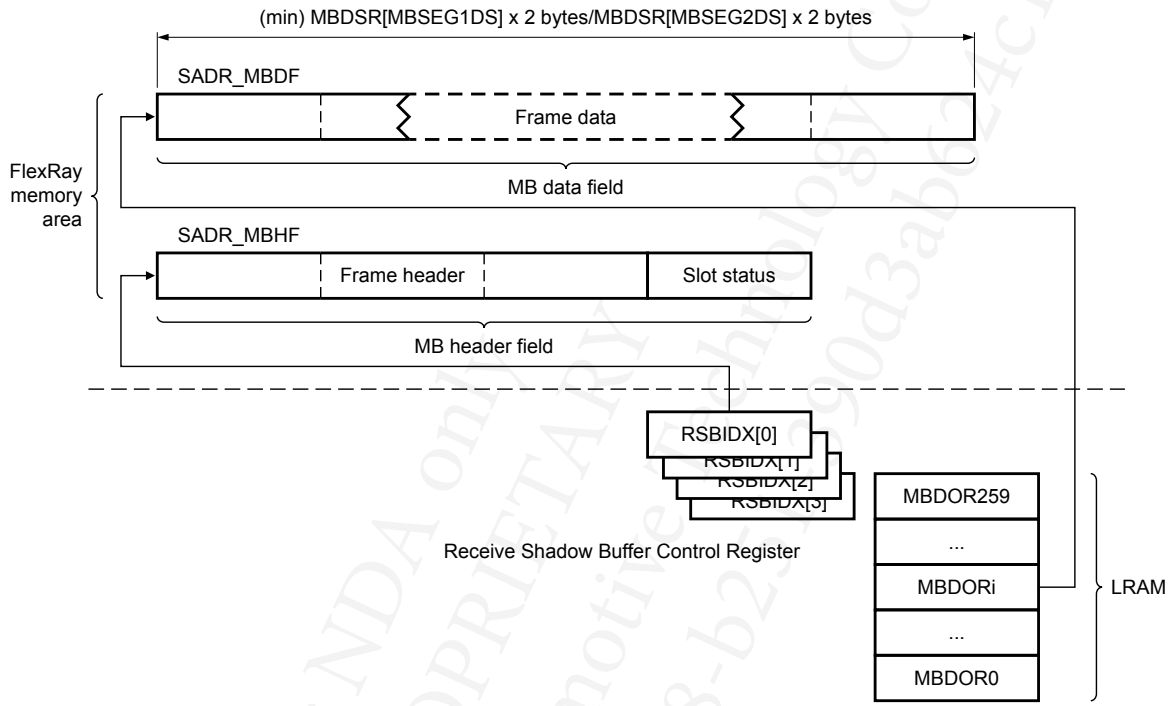


Figure 475. Receive shadow buffer structure

RSBIR[RSBIDX] establishes the connection between the receive shadow buffer control register and the physical MB for the selected receive shadow buffer. The start address SADR_MBHF of the related MB header field in the FlexRay memory area is determined according to this equation:

$$SADR\_MBHF = (RSBIR[RSBIDX] \times 8) + SMBA$$

Equation 3. Equation for SADR_MBHF

The length required for the MB data field depends on the MB segment that the receive shadow buffer is assigned to. For the receive shadow buffers assigned to the first MB segment, the length must be the same as for the individual MBs assigned to the first MB segment. For the receive shadow buffers assigned to the second MB segment, the length must be the same as for the individual MBs assigned to the second MB segment. The receive shadow buffer assignment is described in Receive Shadow Buffer Index (RSBIR).

56.6.3.3 Receive FIFO

The receive FIFO implements a frame reception system based on the FIFO concept. CC provides two independent receive FIFOs, one per channel.

A receive FIFO consists of a set of physical MBs in the FlexRay memory area and a set of receive FIFO control registers located in dedicated registers. The following figure shows the structure of a receive FIFO.

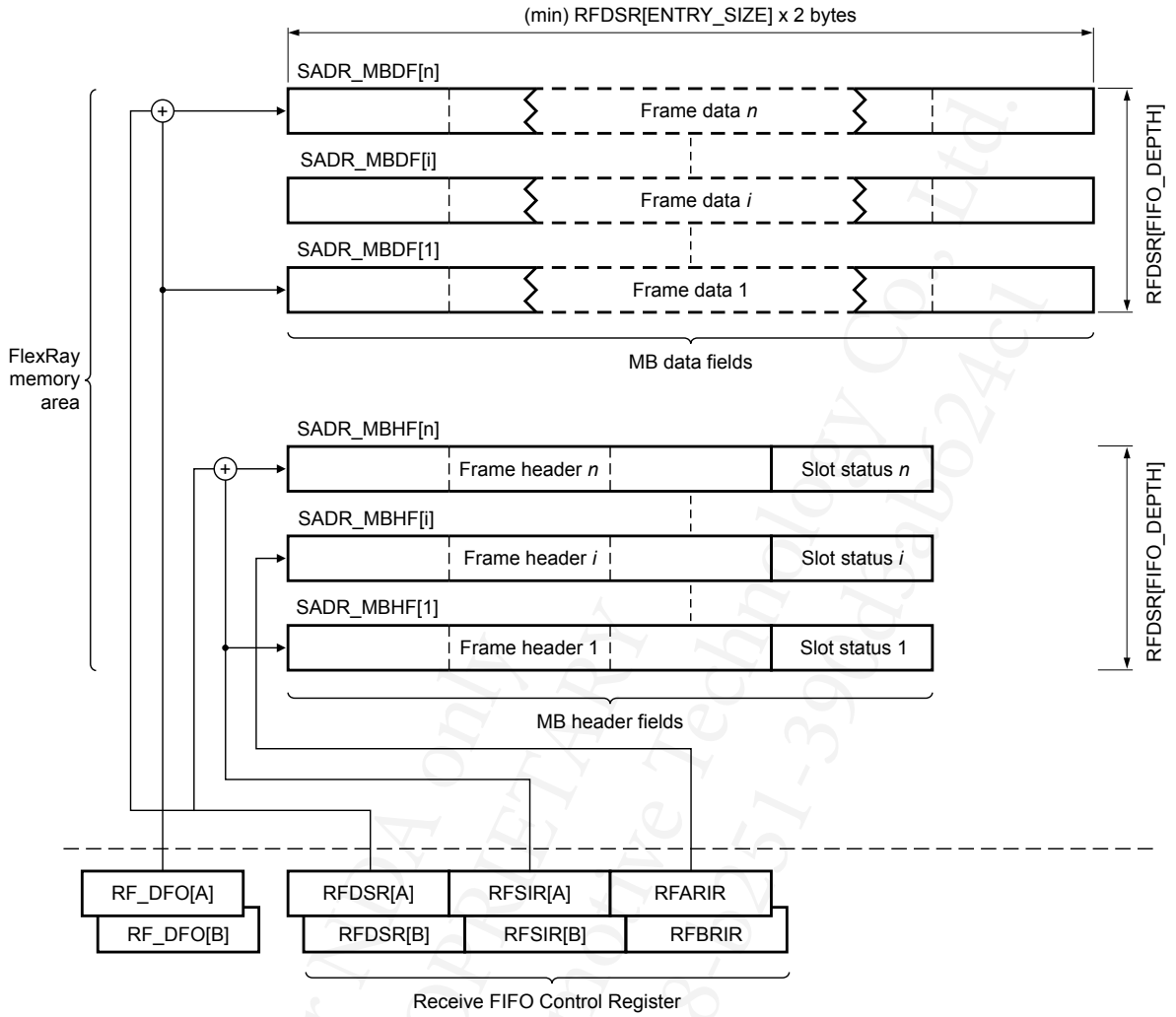


Figure 476. Receive FIFO structure

The following registers establish the connection between the receive FIFO control registers and the set of physical MBs:

- Receive FIFO Start Index (RFSIR)
- Receive FIFO Depth And Size (RFDSR)
- Receive FIFO A Read Index (RFARIR)
- Receive FIFO B Read Index (RFBRIR)

The system memory base address SMBA valid for the receive FIFOs is defined by the system memory base address register that MCR[FAM] selects.

Equation 4 determines the start byte address SADR_MBHF[1] of the first MB header field that belongs to the receive FIFO.

$$SADR\_MBHF[1] = (8 \times RFSIR[SIDX]) + SMBA$$

Equation 4. Equation for SADR_MBHF[1]

Equation 5 determines the start byte address SADR_MBHF[n] of the last MB header field that belongs to the receive FIFO in the FlexRay memory area.

$$SADR\_MBHF[n] = (8 \times (RFSIR[SIDX] + RFDSR[FIFO\_DEPTH])) + SMBA$$

Equation 5. Equation for SADR_MBHF[n]

Receive FIFO A Read Index (RFARIR) and Receive FIFO B Read Index (RFBRIR) provide:

- The required information to access the current entry of the FIFO
- The index of the physical MB belonging to the current entry

You must calculate the data field offset belonging to the current FIFO entry RF_DFO[X] using the current read index *i* according to Equation 6.

$$RF\_DFO[X] = FR\_RFSDOR[X] + (FR\_RFDSR[X][ENTRY\_SIZE] \times 2) \times i - FR\_RFSIR[X][SID])$$

Equation 6. Calculating data field offset

The current read index loops up starting at the number given in the RD[A/B]RDIDX register for the required number of entries.

See FIFO Update for details about updating the FIFO read pointer.

All MB header fields assigned to a receive FIFO are within a contiguous region defined by RFSIR[SIDX] and RFDSR[FIFO_DEPTH].

The data sections of all FIFO entries within on receive FIFO are of the same length defined by RFDSR[FIFO_SIZE].

You must calculate the actual values of the data field offsets RF_DFO[A/B] according to Equation 6. They are not stored in a register.

56.6.3.4 MB configuration and control data

This section describes the configuration and control data for each MB type.

56.6.3.4.1 Individual MB configuration data

Before an individual MB can be used for transmission or reception, you must configure it. There is a set of common configuration parameters that applies to all individual MBs and a set of configuration parameters that applies to each MB individually.

56.6.3.4.1.1 Common configuration data

The set of common configuration data for individual MB is located in the following registers:

- MB Data Size (MBDSR): The MBSEG2DS and MBSEG1DS fields define the minimum length of the MB data field with respect to the MB segment.
- MB Segment Size and Utilization (MBSSUTR): The LAST_MB_SEG1 and LAST_MB_UTIL fields define the segmentation of the individual MBs and the number of individual MBs that are used. For more details, see Individual MB segments.

56.6.3.4.1.2 Specific configuration data

The set of MB-specific configuration data for individual MBs is located in the following registers:

- MB Configuration Control Status (MBCCSR0 - MBCCSR255): The MTD field configures the MB type.
- MB Cycle Counter Filter (MBCCFR0 - MBCCFR255):
  - The MTM, CHA, and CHB fields configure the transmission mode and the channel assignment.
  - The CCFE, CCFMSK, and CCFVAL fields configure the cycle counter filter.
- MB Frame ID (MBFIDR0 - MBFIDR255): For a transmit MB, the FID field determines the slot in which the message in this MB is transmitted.
- MB Index (MBIDXR0 - MBIDXR255): The MBIDX field provides the index of the MB header field of the physical MB that is currently associated with this MB.

56.6.3.5 Individual MB control data

During normal operation, each individual MB can be controlled by the control and trigger fields CMT, LCKT, EDT, and MBIE in MB Configuration Control Status (MBCCSR0 - MBCCSR255).

56.6.3.6 Receive shadow buffer configuration data

Before frame reception into the individual MB can be performed, you must configure the receive shadow buffers. Receive Shadow Buffer Index (RSBIR) provides the configuration data. For each receive shadow buffer, the application provides the MB header index. When the protocol is in the "POC:normal active" or "POC:normal passive" state, the receive shadow buffers are under full CC control.

56.6.3.7 Receive FIFO control and configuration data

This section describes the configuration and control data for the two receive FIFOs.

56.6.3.7.1 Receive FIFO configuration data

CC provides two functional independent receive FIFOs, one per channel. The FIFOs have a common subset of configuration data in Receive FIFO Periodic Timer (RFPTR). Each FIFO has its own set of configuration data that is located in the following registers:

- Receive FIFO Watermark And Selection (RFWMSR)
- Receive FIFO Start Index (RFSIR)
- Receive FIFO Start Data Offset (RFSDOR)
- Receive FIFO Depth And Size (RFDSR)
- Receive FIFO Message ID Acceptance Filter Value (RFMIDAFVR)
- Receive FIFO Message ID Acceptance Filter Mask (RFMIDAFMR)
- Receive FIFO Frame ID Rejection Filter Value (RFFIDRFVR)
- Receive FIFO Frame ID Rejection Filter Mask (RFFIDRFMR)
- Receive FIFO Range Filter Configuration (RFRFCFR)

56.6.3.7.2 Receive FIFO control data

You can access the FIFOs at any time using the control fields in the following registers:

- Global Interrupt Flag And Enable (GIFER)
- Receive FIFO Fill Level and Pop Count (RFFLPCR)

56.6.3.7.3 Receive FIFO status data

The current status of the receive FIFO is provided in the following registers:

- Global Interrupt Flag And Enable (GIFER)
- Receive FIFO A Read Index (RFARIR)
- Receive FIFO B Read Index (RFBRIR)
- Receive FIFO Fill Level and Pop Count (RFFLPCR)

56.6.4 FlexRay memory area layout

CC supports a wide range of possible layouts for the FlexRay memory area. You can select two basic layout modes using MCR[FAM].

Preliminary under NDA only
COMPANY CONFIDENTIAL
Beijing CJJ Automotive Technology Co., Ltd.
ca2462e5-4821-4da8-b251-390d3b624c1

56.6.4.1 FlexRay memory area layout when MCR[FAM] = 0

The figure provided below shows an example layout for the FIFO address mode associated with MCR[FAM] = 0. In this mode, the following set of rules applies to the layout of the FlexRay memory area:

- It is one contiguous region.
- Its size is a maximum of 64 KB.
- It starts at a 16-byte boundary.

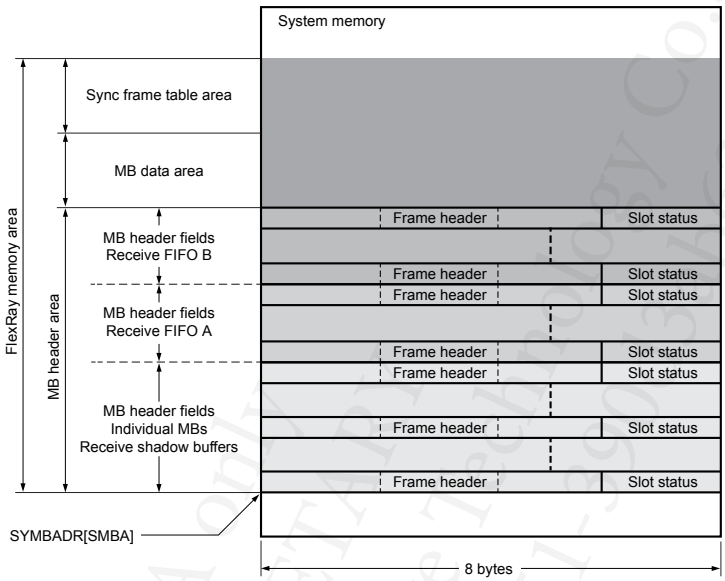


Figure 477. Example of FlexRay memory area layout when MCR[FAM] = 0

The FlexRay memory area contains three areas:

- MB header area
- MB data area
- Sync frame table area

56.6.4.2 FlexRay memory area layout when MCR[FAM] = 1

The following figure shows an example layout for the FIFO address mode associated with MCR[FAM] = 1. The following set of rules applies to the layout of the FlexRay memory area:

- The FlexRay memory area consists of two contiguous regions.
- The size of each region is a maximum of 64 KB.
- Each region starts on a 16-byte boundary.

Provided under NDA only
COMMERCIAL PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c462e5-4621-48a8-b251-9903b624c1

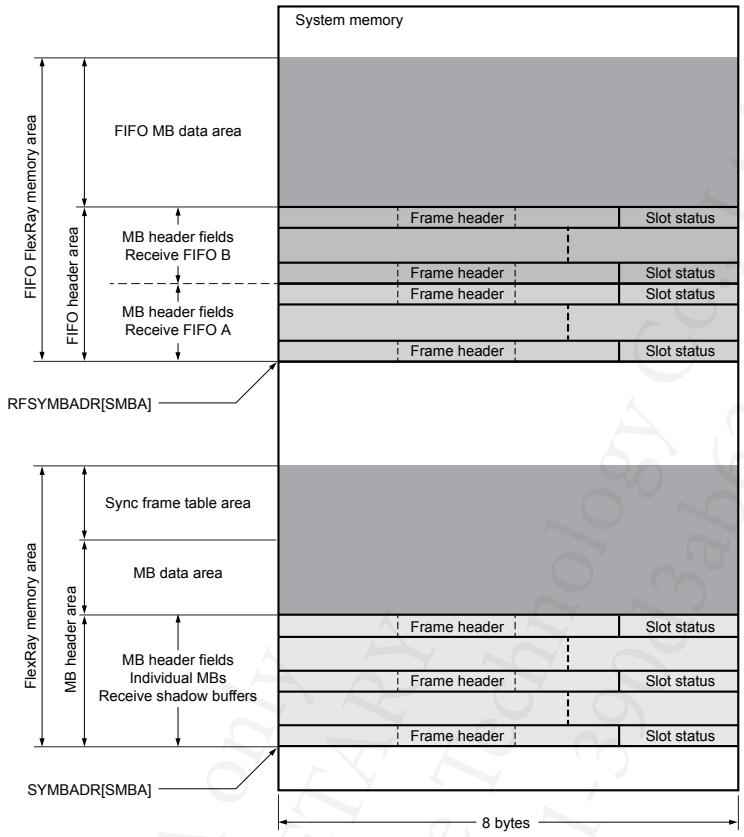


Figure 478. Example of FlexRay memory area layout when MCR[FAM] = 1

56.6.4.3 MB header area when MCR[FAM] = 0

The MB header area contains all MB header fields of the physical MBs for all MB types. The following rules apply to the MB header fields for the three type of MBs.

- The start byte address SADR_MBHF of each MB header field for individual MBs and receive shadow buffers must fulfill Equation 7:

$$SADR\_MBHF = (i \times 8) + SYMBADR[SMBA]\ ;\ (0 \le i \le 259)$$

Equation 7. Equation for SADR_MBHF of individual MBs and receive shadow buffers

- The start byte address SADR_MBHF of each MB header field for the FIFO must fulfill Equation 8:

$$SADR\_MBHF = (i \times 8) + SYMBADR[SMBA]\ ;\ (0 \le i \le 1023)$$

Equation 8. Equation for SADR_MBHF of the FIFO

- The MB header fields for each FIFO must be a contiguous area.

56.6.4.4 MB header area with MCR[FAM] = 1

The MB header area contains all MB header fields of the physical MBs for the individual MBs and receiver shadow buffers.

For the two type of MBs, the start address SADR_MBHF of each MB header field for individual MBs and receive shadow buffers must fulfill Equation 9:

$$SADR\_MBHF = (i \times 8) + SYMBADR[SMBA]\ ;\ (0 \le i \le 259)$$

Equation 9. Equation for SADR_MBHF

56.6.4.5 FIFO MB header area with MCR[FAM] = 1

The FIFO MB header area contains all MB header fields of the physical MBs for the FIFO. The following rules apply to the FIFO MB header fields:

- The start byte address SADR_MBHF of each MB header field for the FIFO must fulfill the following equation:

$$SADR\_MBHF = (i\gamma 8) + RFSYMBADR[SMBA];(0 \le i \le 1023)$$

Equation 10. Equation for SADR_MBHF

- The MB header fields for each FIFO must be a contiguous area.

56.6.4.6 MB data area

The MB data area contains all the MB data fields of the physical MBs. Each MB data field must start at a 16-bit boundary.

56.6.4.7 Sync frame table area

The sync frame table area provides a copy of the internal sync frame tables for application access. See Sync Frame ID and Sync Frame Deviation Tables for the description of the sync frame table area.

56.6.5 Physical MB description

This section provides a detailed description of the usage and the content of the two parts of a physical MB—the MB header field and the MB data field.

56.6.5.1 MB protection and data consistency

The physical MBs are located in the FlexRay memory area. CC provides no means to protect the FlexRay memory area from uncontrolled or illegal host or other client write access. To ensure data consistency of the physical MBs, you must follow the write access scheme that is given in the description of each of the physical MB fields.

56.6.5.2 MB header field description

This section provides a detailed description of the usage and content of the MB header field. A description of the structure of the MB header fields is provided in MB header field. Each MB header field consists of two sections—frame header and slot status.

56.6.5.2.1 Frame header description

56.6.5.2.1.1 Frame header content

The semantic and content of the frame header section depends on the MB type.

For individual receive MBs and receive FIFOs, the frame header receives the frame header data of the first valid frame received on the assigned channels.

For receive shadow buffers, the frame header receives the frame header data of the current frame received regardless of whether the frame is valid or not.

For transmit MBs, you write the frame header of the to-be-transmitted frame to this location. The frame header is read out when the frame is transferred to the FlexRay bus.

The structure of the frame header in the MB header field for receive MBs and the receive FIFO is given in Table 468. A detailed description is given in Table 472.

Table 468. Frame header structure (receive MB and receive FIFO)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0h	R	PPI	NUF	SYF	SUF	FID										
2h	0	0	CYCCNT						0	PLDLEN						
4h	0	0	0	0	0	HDCRC										

The structure of the frame header in the MB header field for transmit MBs is given in Table 469. A detailed description is given in Table 473. The checks to be performed are described in Frame header checks.

Table 469. Frame header structure (transmit MB)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0h	R	PPI	NUF	SYF	SUF	FID (checked)										
2h			CYCCNT							PLDLEN (checked if not static)						
4h						HDCRC										

= Not used

The structure of the frame header in the MB header field for transmit MBs assigned to key slot is provided in Table 470.

Table 470. Frame header structure (transmit MB for key slot)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0h	R	PPI	NUF	SYF	SUF	FID										
2h			CYCCNT							PLDLEN						
4h						HDCRC										

= Not used

56.6.5.2.1.2 Frame header access

The frame header is located in the FlexRay memory area. To ensure data consistency, you must follow the write access scheme described below.

For receive MBs, receive shadow buffers, and receive FIFOs, you must not write to the frame header field.

For transmit MBs, you must follow the write access restrictions given in the following table. This table shows the condition under which you can write to the frame header entries without corrupting the FlexRay message transmission.

Table 471. Frame header write access constraints (transmit MB)

Field	Static segment	Dynamic segment
FID	POC:config or MB_DIS	

Table continues on the next page...

Table 471. Frame header write access constraints (transmit MB) (continued)

Field	Static segment	Dynamic segment
PPI, PLDLEN, HDCRC	POC:config or MB_DIS	
		MB_LCK

56.6.5.2.1.3 Frame header checks

As shown in Table 469 and Table 470, not all fields in the MB frame header are used for transmission. Some fields in the MB frame header are ignored, some are used for transmission, and some of them are checked for correct values. All checks that are performed are described below.

For MBs assigned to the key slot, no checks are performed.

The value of the FID field must be equal to the value of the corresponding MB Frame ID (MBFIDR0 - MBFIDR255). If CC detects a mismatch while transmitting the frame header, it causes CHIERFR[FID_EF] to become 1. The value of the FID field is ignored and replaced by the value provided in MB Frame ID (MBFIDR0 - MBFIDR255).

For transmit MBs assigned to the static segment, the PLDLEN value must be equal to PCR19[payload_length_static]. If this is not fulfilled, CHIERFR[SPL_EF] becomes 1 when the MB is under transmission. A syntactically and semantically correct frame is generated with PCR19[payload_length_static] payload words and the payload length field in the transmitted frame header set to payload_length_static.

For transmit MBs assigned to the dynamic segment, the PLDLEN value must be less than or equal to PCR24[max_payload_length_dynamic]. If this is not fulfilled, CHIERFR[DPL_EF] becomes 1 when the MB is under transmission. A syntactically and semantically correct dynamic frame is generated with PLDLEN payload words and the payload length field in the frame header set to PLDLEN.

Table 472. Frame header field descriptions (receive MB and receive FIFO)

Field	Description
R	Reserved field—This is the value of the reserved field of the received frame stored in the MB.
PPI	Payload preamble indicator—This is the value of the payload preamble indicator of the received frame stored in the MB.
NUF	Null frame indicator—This is the value of the null frame indicator of the received frame stored in the MB.
SYF	Sync frame indicator—This is the value of the sync frame indicator of the received frame stored in the MB.
SUF	Startup frame indicator—This is the value of the startup frame indicator of the received frame stored in the MB.
FID	Frame ID—This is the value of the frame ID field of the received frame stored in the MB.
CYCCNT	Cycle count—This is the number of the communication cycle in which the frame stored in the MB was received.
PLDLEN	Payload length—This is the value of the payload length field of the received frame stored in the MB.
HDCRC	Header CRC—This is the value of the header CRC field of the received frame stored in the MB.

Table 473. Frame header field descriptions (transmit MB)

Field	Description
R	Reserved field—Not used; the value of the reserved field is generated internally according to *FlexRay Communications System Protocol Specification, Version* 2.1 Rev A.
PPI	Payload preamble indicator—This field provides the value of the payload preamble indicator for the frame transmitted from the MB.
NUF	Null frame indicator—Not used; the value of the null frame indicator is generated internally according to *FlexRay Communications System Protocol Specification, Version 2.1 Rev A*.
SYF	Sync frame indicator—Not used; the value of the sync frame indicator is generated internally according to *FlexRay Communications System Protocol Specification, Version 2.1 Rev A*.
SUF	Startup frame indicator—Not used; the value of the startup frame indicator is generated internally according to *FlexRay Communications System Protocol Specification, Version 2.1 Rev A*.
FID	Frame ID—This field is checked as described in Frame Header Checks.
CYCCNT	Cycle count—Not used; the value of the transmitted cycle count field is taken from the internal communication cycle counter.
PLDLEN	Payload length—This field is checked and used as described in Frame Header Checks.
HDCRC	Header CRC—This field provides the value of the header CRC field for the frame transmitted from the MB.

56.6.5.2.2 Slot status description

The slot status is a read-only structure for you and a write-only structure for CC. The meaning and content of the slot status in the MB header field depends on the MB type.

56.6.5.2.2.1 Slot status description for the receive MBs and receive FIFOs

The content of the slot status structure for receive MBs depends on the MB type and on the channel assignment for individual receive MBs as given in Table 474.

Table 474. Receive MB slot status content

Receive MB type	MBCCFR*n*[CHA]	MBCCFR*n*[CHB]	Slot status content
Individual receive MB assigned to both channels	1	1	See Table 475.
Individual receive MB assigned to channel A	1	0	See Table 476.
Individual receive MB assigned to channel B	0	1	See Table 477.
Receive FIFO channel A MB	—	—	See Table 476.
Receive FIFO channel B MB	—	—	See Table 477.

Table 478 provides the meaning of the fields in the slot status structure.

Table 475. Receive MB slot status structure (ChAB)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	CH	VFA	SYA	NFA	SUA	SEA	CEA	BVA	0
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Table 476. Receive MB slot status structure (ChA)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	VFA	SYA	NFA	SUA	SEA	CEA	BVA	0
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Table 477. Receive MB slot status structure (ChB)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	1	0	0	0	0	0	0	0	0
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Table 478. Receive MB slot status field descriptions

Field	Description
	Common MB status fields
VFB	Valid Frame On Channel B Protocol-related variable: vSS!ValidFrame channel B 0: vSS!ValidFrame = 0 1: vSS!ValidFrame = 1
SYB	Sync Frame Indicator Channel B Protocol-related variable: vRF!Header!SyFIndicator channel B 0: vRF!Header!SyFIndicator = 0 1: vRF!Header!SyFIndicator = 1
NFB	Null Frame Indicator Channel B Protocol-related variable: vRF!Header!NFIndicator channel B 0: vRF!Header!NFIndicator = 0 1: vRF!Header!NFIndicator = 1
SUB	Startup Frame Indicator Channel B Protocol-related variable: vRF!Header!SuFIndicator channel B 0: vRF!Header!SuFIndicator = 0

Table continues on the next page...

Table 478. Receive MB slot status field descriptions (continued)

Field	Description
	1: vRF!Header!SuFIndicator = 1
SEB	Syntax Error On Channel B Protocol-related variable: vSS!SyntaxError channel B 0: vSS!SyntaxError = 0 1: vSS!SyntaxError = 1
CEB	Content Error On Channel B Protocol-related variable: vSS!ContentError channel B 0: vSS!ContentError = 0 1: vSS!ContentError = 1
BVB	Boundary Violation On Channel B Protocol-related variable: vSS!BViolation channel B 0: vSS!BViolation = 0 1: vSS!BViolation = 1
CH	Channel First Valid Received Applies only to receive MBs assigned to the static segment and to both channels. It indicates the channel that has received the first valid frame in the slot. This field becomes 0 if no valid frame was received in the subscribed slot. 0: First valid frame received on channel A, or no valid frame received 1: First valid frame received on channel B
VFA	Valid Frame On Channel A Protocol-related variable: vSS!ValidFrame channel A 0: vSS!ValidFrame = 0 1: vSS!ValidFrame = 1
SYA	Sync Frame Indicator Channel A Protocol-related variable: vRF!Header!SyFIndicator channel A 0: vRF!Header!SyFIndicator = 0 1: vRF!Header!SyFIndicator = 1
NFA	Null Frame Indicator Channel A Protocol-related variable: vRF!Header!NFIndicator channel A 0: vRF!Header!NFIndicator = 0 1: vRF!Header!NFIndicator = 1
SUA	Startup Frame Indicator Channel A

Table continues on the next page...

Table 478. Receive MB slot status field descriptions (continued)

Field	Description
	Protocol-related variable: vRF!Header!SuFIndicator channel A 0: vRF!Header!SuFIndicator = 0 1: vRF!Header!SuFIndicator = 1
SEA	Syntax Error On Channel A Protocol-related variable: vSS!SyntaxError channel A 0: vSS!SyntaxError = 0 1: vSS!SyntaxError = 1
CEA	Content Error On Channel A Protocol-related variable: vSS!ContentError channel A 0: vSS!ContentError = 0 1: vSS!ContentError = 1
BVA	Boundary Violation On Channel A Protocol-related variable: vSS!BViolation channel A 0: vSS!BViolation = 0 1: vSS!BViolation = 1

56.6.5.2.2.2 Slot status description for transmit MBs

Only the TCA and TCB status fields are directly related to the transmission process. All other status fields in this structure are related to a receive process that may have occurred. The content of the slot status structure for transmit MBs depends on the channel assignment as given in Table 479.

Table 479. Transmit MB slot status content

Transmit MB type	MBCCFR*n*[CHA]	MBCCFR*n*[CHB]	Slot status content
Individual transmit MB assigned to both channels	1	1	See Table 480.
Individual transmit MB assigned to channel A	1	0	See Table 481.
Individual transmit MB assigned to channel B MBCCFRn[CHA]=0 and MBCCFRn[CHB]=1	0	1	See Table 482.

Table 483 provides the meaning of the fields in the slot status structure.

Freescale Confidential Proprietary
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c402e5-4821-42a8-9251-390d3ab624c1

Table 480. Transmit MB slot status structure (ChAB)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Table 481. Transmit MB slot status structure (ChA)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Table 482. Transmit MB slot status structure (ChB)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB	0	0	0	0	0	0	0	0
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Table 483. Transmit MB slot status structure field descriptions

Field	Description
VFB	Valid Frame On Channel B Protocol-related variable: vSS!ValidFrame channel B 0: vSS!ValidFrame = 0 1: vSS!ValidFrame = 1
SYB	Sync Frame Indicator Channel B Protocol-related variable: vRF!Header!SyFIndicator channel B 0: vRF!Header!SyFIndicator = 0 1: vRF!Header!SyFIndicator = 1
NFB	Null Frame Indicator Channel B Protocol-related variable: vRF!Header!NFIndicator channel B 0: vRF!Header!NFIndicator = 0 1: vRF!Header!NFIndicator = 1
SUB	Startup Frame Indicator Channel B Protocol-related variable: vRF!Header!SuFIndicator channel B 0: vRF!Header!SuFIndicator = 0 1: vRF!Header!SuFIndicator = 1

Table continues on the next page...

Table 483. Transmit MB slot status structure field descriptions (continued)

Field	Description
SEB	Syntax Error On Channel B Protocol-related variable: vSS!SyntaxError channel B 0: vSS!SyntaxError = 0 1: vSS!SyntaxError = 1
CEB	Content Error On Channel B Protocol-related variable: vSS!ContentError channel B 0: vSS!ContentError = 0 1: vSS!ContentError = 1
BVB	Boundary Violation On Channel B Protocol-related variable: vSS!BViolation channel B 0: vSS!BViolation = 0 1: vSS!BViolation = 1
TCB	Transmission Conflict On Channel B Protocol-related variable: vSS!TxConflict channel B 0: vSS!TxConflict = 0 1: vSS!TxConflict = 1
VFA	Valid Frame On Channel A Protocol-related variable: vSS!ValidFrame channel A 0: vSS!ValidFrame = 0 1: vSS!ValidFrame = 1
SYA	Sync Frame Indicator Channel A Protocol-related variable: vRF!Header!SyFIndicator channel A 0: vRF!Header!SyFIndicator = 0 1: vRF!Header!SyFIndicator = 1
NFA	Null Frame Indicator Channel A Protocol-related variable: vRF!Header!NFIndicator channel A 0: vRF!Header!NFIndicator = 0 1: vRF!Header!NFIndicator = 1
SUA	Startup Frame Indicator Channel A Protocol-related variable: vRF!Header!SuFIndicator channel A 0: vRF!Header!SuFIndicator = 0 1: vRF!Header!SuFIndicator = 1

Table continues on the next page...

Table 483. Transmit MB slot status structure field descriptions (continued)

Field	Description
SEA	Syntax Error On Channel A Protocol-related variable: vSS!SyntaxError channel A 0: vSS!SyntaxError = 0 1: vSS!SyntaxError = 1
CEA	Content Error On Channel A Protocol-related variable: vSS!ContentError channel A 0: vSS!ContentError = 0 1: vSS!ContentError = 1
BVA	Boundary Violation On Channel A Protocol-related variable: vSS!BViolation channel A 0: vSS!BViolation = 0 1: vSS!BViolation = 1

56.6.5.3 MB data field description

The MB data field is used to store the payload data—or a part of that data—of the frame that you want to transmit or receive on the FlexRay bus. The minimum required length of this field depends on the MB type that the physical MB is assigned to and is given in Table 484. Table 485 shows the structure of the MB data field.

Table 484. MB data field minimum length

Physical MB assigned to	Minimum length defined by
Individual MB in segment 1	MBDSR[MBSEG1DS]
Receive shadow buffer in segment 1	
Individual MB in segment 2	MBDSR[MBSEG2DS]
Receive shadow buffer in segment 2	
Receive FIFO for channel A	RFDSR[ENTRY_SIZE] (RFWMSR[SEL] = 0)
Receive FIFO for channel B	RFDSR[ENTRY_SIZE] (RFWMSR[SEL] = 1)

NOTE

CC does not access any locations outside the MB data field boundaries given in Table 484.

Table 485. MB data field structure

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0h	DATA0 / MID0 / NMV0								DATA1 / MID1 / NMV1							
2h	DATA2 / NMV2								DATA3 / NMV3							
…	…								…							

Table continues on the next page...

Table 485. MB data field structure (continued)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
(N-2)h	DATA N-2								DATA N-1							

The MB data field is located in the FlexRay memory area; thus, CC has no means to control your write access to the field. To ensure data consistency, you must follow a write and read access scheme.

56.6.5.3.1 MB data field read access

For transmit MBs, CC does not modify the content of the MB data field. Thus you can read back the data at any time without any impact on data consistency.

For receive MBs, you must lock the related receive MB and retrieve the MB header index from MB Index (MBIDXR0 - MBIDXR255). While the MB is locked, CC does not update the MB data field.

For receive FIFOs, you can read the MB indicated by Receive FIFO A Read Index (RFARIR) or Receive FIFO B Read Index (RFBRIR) when the related fill levels in Receive FIFO Fill Level and Pop Count (RFFLPCR) indicate a non-empty FIFO.

56.6.5.3.2 MB data field write access

For receive MBs, receive shadow buffers, and receive FIFOs, you must not write to the MB data field.

For transmit MBs, you must follow the write access restrictions provided in Table 486.

Table 486. Frame data write access constraints

Field	CC/MB state
DATA, MID, NMV	POC:config or MB_DIS or MB_LCK

Table 487. Frame data field descriptions

Field	Description
DATA 0–DATA N-1	Message Data Provides the message data received or to be transmitted. For receive MBs and receive FIFOs, this field provides the message data received for this MB. For transmit MBs, the field provides the message data to be transmitted.
MID 0–1	Message Identifier If the MB frame header's PPI field is 1, MID *n* holds the message ID of a dynamic frame located in the MB. The receive FIFO filter uses the received message ID for message ID filtering.
NMV 0–11	NMV If the MB frame header's PPI field is 1, NMV *n* holds the NMV of a static frame located in the MB. **NOTE** The MID and NMV bytes replace the corresponding DATA bytes.

56.6.6 Individual MB functional description

CC provides these basic types of individual MBs:

- Transmit MBs

- Receive MBs

Before an individual MB can be used, you must configure it. After the initial configuration, the MB can be reconfigured later. Individual MB configuration data provides the set of configuration data for individual MBs.

56.6.6.1 Individual MB configuration

The individual MB configuration consists of two steps:

1. Allocation of the required amount of memory for the FlexRay memory area
2. Programming of the MB configuration registers (described in the following subsections)

56.6.6.1.1 Common configuration data

One part of the MB configuration data is common to all individual MBs and the receive shadow buffers. This data can only be set when the protocol is in the POC:config state.

Table 488. Common configuration data

To configure this:	Do this:
The number of used MBs	Write the MB number of the last-used MB to MBSSUTR[LAST_MB_UTIL].
The size of the two segments of individual MBs	Write the MB number of the last MB in the first segment to MBSSUTR[LAST_MB_SEG1].
The length of the MB data fields for both MB segments	Write to MBDSR[MBSEG1DS] and MBDSR[MBSEG2DS].
The receive shadow buffers (depending on the current CC receive functionality)	Write to Receive Shadow Buffer Index (RSBIR) for each segment and for each channel with at least one individual receive MB assigned.

56.6.6.1.2 Specific configuration data

The second part of the MB configuration data is specific for each MB.

This data can be changed only when either of the following is true:

- The protocol is in the POC:config state.
- The MB is disabled—that is, MBCCSR*n*[EDS] = 0.

The individual MB type is defined by MBCCSR*n*[MTD] as given in the following table.

Table 489. Individual MB types

MBCCSR*n*[MTD]	Individual MB description
0	Receive MB
1	Transmit MB

The MB-specific configuration data is:

- MBCCSR*n*[MTD]
- All fields in MB Cycle Counter Filter (MBCCFR0 - MBCCFR255)
- All fields in MB Frame ID (MBFIDR0 - MBFIDR255)
- All fields in MB Index (MBIDXR0 - MBIDXR255)

The meaning of the specific configuration data depends on the MB type, as described in Transmit MBs and Receive MBs.

Provided under NDA
COMPANY PROPRIETARY
Beijing CEI Automotive Technology Co., Ltd.
ce42e5-1121-48a8-b25f-8905ab624c1

56.6.6.2 Transmit MBs

The section provides a detailed description of the functionality of single-buffered transmit MBs.

You use a transmit MB to provide message data to CC that is transmitted over the FlexRay bus. CC uses the transmit MBs to provide information about the transmission process and status information about the slot in which the message was transmitted.

The individual MB with MB number n is configured to be a transmit MB by the following settings:

- MBCCSR*n*[MBT] = 0 (single-buffered MB)
- MBCCSR*n*[MDT] = 1 (transmit MB)

56.6.6.2.1 Access regions

Both you and CC have access to certain MB fields. To ensure data consistency, an MB locking scheme controls the access to the MB data, control, and status fields. The access regions for transmit MBs are depicted in the following figure. A description of the regions is given in Table 491. If a region is active as indicated in Table 492, the access scheme given for that region applies to the MB.

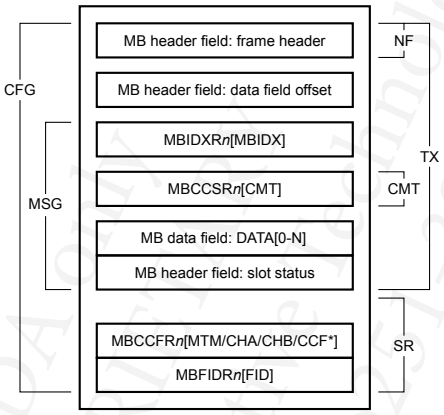


Figure 479. Transmit MB access regions

Table 490. Transmit MB access region description

Region	Access from		Region used for
	Application	Module	
CFG	Read/write	—	MB configuration
MSG	Read/write	—	Message data and slot status access
NF	—	Read-only	Message header access for null frame transmission
TX	—	Read/write	Message transmission and slot status update
CM	—	Read-only	MB validation
SR	—	Read-only	MB search

The EDT and LCKT trigger fields, and the MBIE interrupt-enable field, in MB Configuration Control Status (MBCCSR0 - MBCCSR255) are not under access control and can be accessed from the application at any time. The EDS and LCKS status fields are not under access control and can be accessed from CC at any time.

MBCCSR*n*[MBIF] is not under access control and can be accessed from the application and CC at any time. CC clear access has higher priority.

CC restricts its access to the regions depending on the current state of the MB. You must comply with these restrictions to ensure data consistency. The transmit MB states are given in Figure 480. A description of the states is given in Table 491, which also provides the access scheme for the access regions.

The EDS and LCKS status fields in MB Configuration Control Status (MBCCSR0 - MBCCSR255) provide you with the required MB status information. The internal status information is not visible to you.

56.6.6.2.2 MB states

This section describes the transmit MB states and provides a state diagram.

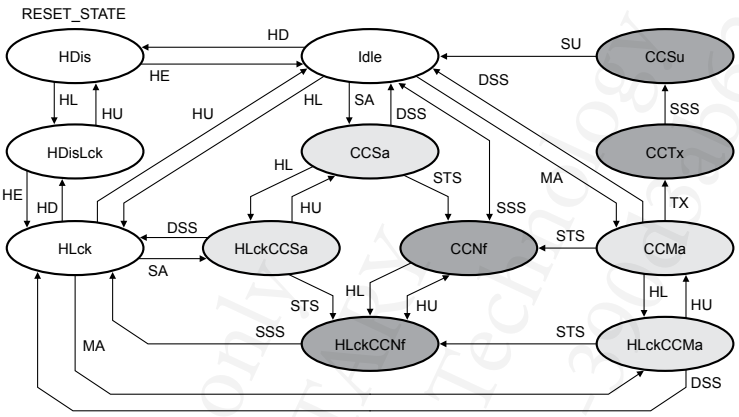


Figure 480. Transmit MB states

Table 491. Transmit MB state description

State	MBCCSRn		Access region		Description
	EDS	LCKS	Application	Module	
Idle	1	0	—	CM, SR	Idle MB is idle. Included in MB search.
HDis	0	0	CFG	—	Disabled MB under configuration. Excluded from MB search.
HDisLck	0	1	CFG	—	Disabled and locked MB under configuration. Excluded from MB search.
HLck	1	1	MSG	SR	Locked Applications access to data, control, and status. Included in MB search.
CCSa	1	0	—	—	Slot assigned MB assigned to next static slot. Ready for null frame transmission.
HLckCCSa	1	1	MSG	—	Locked and slot assigned

Table continues on the next page...

Table 491. Transmit MB state description (continued)

State	MBCCSRn		Access region		Description
	EDS	LCKS	Application	Module	
					Applications access to data, control, and status. MB assigned to next static slot.
CCNf	1	0	—	NF	Null frame transmission Header is used for null frame transmission.
HLckCCNf	1	1	MSG	NF	Locked and null frame transmission Applications access to data, control, and status. Header is used for null frame transmission.
CCMa	1	0	—	CM	Message available MB is assigned to next slot and cycle counter filter matches.
HLckCCMa	1	1	MSG	—	Locked and message available Applications access to data, control, and status. MB is assigned to next slot and cycle counter filter matches.
CCTx	1	0	—	TX	Message transmission MB data transmit. Payload data from buffer transmitted.
CCSu	1	0	—	TX	Status update MB status update. Update of status flags, the slot status field, and the header index.

56.6.6.2.3 MB transitions

This section discusses:

- Application transitions
- Module transitions
- Transition priorities

56.6.6.2.3.1 Application transitions

You can trigger the application transitions using the commands described in Table 492. You issue the commands by writing to MB Configuration Control Status (MBCCSR0 - MBCCSR255). Only one command can be issued with one write access. Each command executes immediately. If the command is ignored, you must issue it again.

56.6.6.2.3.1.1 MB enable and disable

You issue the enable and disable commands by writing 1 to MBCCSR*n*[EDT]. The transition that each of these commands triggers depends on the current value of MBCCSR*n*[EDS].

If the command triggers the disable transition HD and the MB is in one of the states:

- CCSa
- HLckCCSa

- CCMa
- HLckCCMa
- CCNf
- HLckCCNf
- CCTx

then the disable transition has no effect (the command is ignored) and the MB state does not change. You do not receive any notification.

NOTE

Do not disable an MB when FlexRay is in the STARTUP protocol state.

56.6.6.2.3.1.2 MB lock and unlock

You issue the lock and unlock commands by writing 1 to MBCCSR*n*[LCKT]. The transition that each of these commands triggers depends on the current value of MBCCSR*n*[LCKS]. If the command triggers the HL lock transition and the MB is in the CCTx state, the lock transition has no effect (command is ignored) and the MB state does not change. In this case, CHIERFR[LCK_EF] becomes 1.

Table 492. Transmit MB application transitions

Transition	Command	Condition	Description
HE	Write 1 to MBCCSR*n*[EDT]	MBCCSR*n*[EDS] = 0	Application triggers MB enable.
HD		MBCCSR*n*[EDS] = 1	Application triggers MB disable.
HL	Write 1 to MBCCSR*n*[LCKT]	MBCCSR*n*[LCKS] = 0	Application triggers MB lock.
HU		MBCCSR*n*[LCKS] = 1	Application triggers MB unlock.

56.6.6.2.3.2 Module transitions

The following table describes the module transitions that CC can trigger. Each transition is triggered for certain MBs when the related condition is fulfilled.

Table 493. Transmit MB module transitions

Transition	Condition	Description
SA	Slot match and static slot	Slot assigned MB is assigned to the next static slot.
MA	Slot match and cycle counter match	Message available MB is assigned to the next slot and cycle counter filter matches.
TX	Slot start and MBCCSR*n*[CMT] = 1	Transmission slot start Slot start and MBCCSR*n*[CMT] = 1. In case of a dynamic slot, pLatestTx is not exceeded.
SU	Status updated	Status updated Slot status field and MB status flags updated. Interrupt flag becomes 1.

Table continues on the next page...

Table 493. Transmit MB module transitions (continued)

Transition	Condition	Description
STS	Static slot start	Static slot start
DSS	Dynamic slot start, symbol window start, or NIT start	Dynamic slot or segment start
SSS	Slot start, symbol window start, or NIT start	Slot or segment start

56.6.6.2.3.3 Transition priorities

You can trigger only one transition at a time. There is no need to specify priorities among them.

As shown in the first part of the following table, the module transitions have a higher priority than the application transitions. For all states except the CCMa state, both a lock/unlock transition HL/HD and a module transition can be executed at the same time. The result state is reached by first applying the application transition and subsequently the module transition to the intermediately reached state. For example, if all of the following is true:

- The MB is in the HLck state
- Your application unlocks the MB by using the HU transition
- The module triggers the slot assigned transition SA

then the intermediate state is Idle and the resulting state is CCSa.

The second part of the following table gives the priorities among the module transitions.

Table 494. Transmit MB transition priorities

State	Priorities	Description
Module vs. application		
Idle, HLck	SA > HD	Slot assigned > MB disable
	MA > HD	Message available > MB disable
CCMa	TX > HL	Transmission start > MB lock
Module internal		
Idle, HLck	MA > SA	Message available > slot assigned
CCMa	TX > STS	Transmission slot start > static slot start
	TX > DSS	Transmission slot start > dynamic slot start

56.6.6.2.4 Transmit message setup

To transmit a message over the FlexRay bus, you:

- Write the message data into the MB data field
- Write 1 to MBCCSR*n*[CMT]

Individual MBs describes the physical access to the MB data field.

As indicated by Table 483, you must write to the MB data field and change MBCCSR*n*[CMT] only if the transmit MB is in one of these states:

- HDis

- HDisLck
- HLck
- HLckCCSa
- HLckCCMa
- HLckCCMa

You can change the MB state if you issue the appropriate commands shown in Table 483. The EDS and LCKS fields of MB Configuration Control Status (MBCCSR0 - MBCCSR255) indicate the state change.

If the transmit MB enters one of these states:

- HDis
- HDisLck
- HLck
- HLckCCSa
- HLckCCMa

then MBCCSR*n*[DVAL] becomes 0.

56.6.6.2.5 Message transmission

As a result of the MB search described in Individual MB search, CC triggers the message-available transition (MA) for up to two transmit MBs. This changes the MB state from Idle to CCMa and the MBs can be used for message transmission in the next slot.

CC transmits a message from an MB if both of the following conditions are fulfilled at the start of the transmission slot:

- The MB is in the message available state CCMa
- The message data is still valid—that is, MBCCSR*n*[CMT] = 1

In this case, CC triggers the TX transition and changes the MB state to CCTx.

Figure 481 shows transmit MB timing and state changes. In this example:

- The MB with MB number n is idle at the start of the search slot.
- The MB with MB number n matches the slot and cycle number of the next slot.
- MB data is valid—that is, MBCCSR*n*[CMT] = 1.

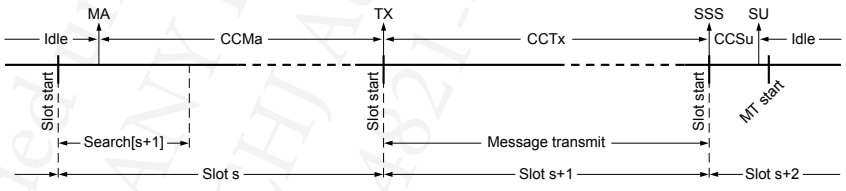


Figure 481. Message transmission timing

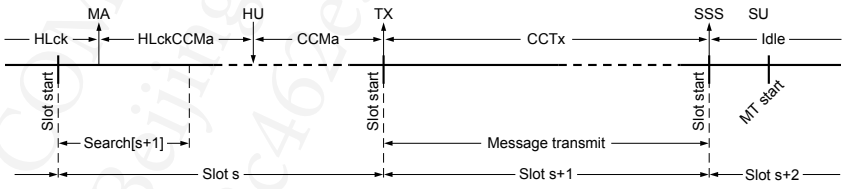


Figure 482. Message transmission from HLck state with unlock

The following items determine the amount of message data read from the FlexRay memory area and transferred to the FlexRay bus:

- The MB segment that the MB is assigned to, as defined by MB Segment Size and Utilization (MBSSUTR)
- The MB data field size, as defined by the related field of MB Data Size (MBDSR)
- The value of the PLDLEN field in the MB header field, as described in Frame header description

If an MB is assigned to MB segment 1, and PLDLEN > MBSEG1DS, then 2 × MBSEG1DS bytes are read from the MB data field and zero padding is used for the remaining bytes for the FlexRay bus transfer. If PLDLEN ≤ MBSEG1DS, CC reads and transfers 2 × PLDLEN bytes. The same is true for segment 2 and MBSEG2DS.

56.6.6.2.6 Null frame transmission

A static slot with slot number S is assigned to CC for channel A if at least one transmit MB is configured with MBFIDR*n*[FID] = S and MBCCFR*n*[CHA] = 1. A null frame is transmitted in the static slot S on channel A if both of the following are true:

- This slot is assigned to CC for channel A.
- One of the following is true for all transmit MBs with MBFIDR*n*[FID] = S and MBCCFR*n*[CHA] = 1:
  - Not committed (that is, MBCCSR*n*[CMT] = 0)
  - You locked them (that is, MBCCSR*n*[LCKS] = 1)
  - The cycle counter filter is enabled and does not match

Additionally, you can write 0 to the commit field of an MB that is in the CCMa state, which is called "uncommit" or "transmit abort". This MB is used for null frame transmission.

As a result of the MB search described in Individual MB search, CC triggers the slot assigned transition SA for up to two transmit MBs if at least one of the conditions mentioned above is fulfilled for these MBs. The transition SA changes the MB states from either Idle to CCSa or from HLck to HLckCCSa. In each case, these MBs are used for null frame transmission in the next slot.

Figure 483 shows an MB timing and state-change diagram for null frame transmission from Idle state.

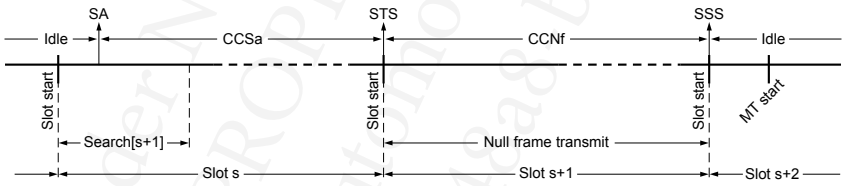


Figure 483. Null frame transmission from Idle state

Figure 484 shows an MB timing and state-change diagram for null frame transmission from HLck state.

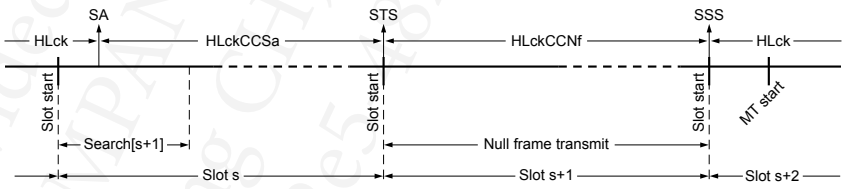


Figure 484. Null frame transmission from HLck state

If a transmit MB is in the CCSa or HLckCCSa state at the start of the transmission slot, a null frame is transmitted in any case, even if the MB is unlocked or committed before the transmission slot starts. Figure 485 shows a transmit MB timing and state-change diagram for null frame transmission for this case.

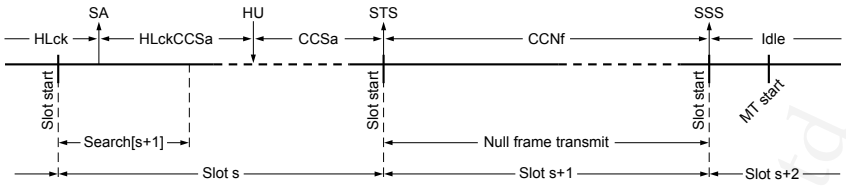


Figure 485. Null frame transmission from HLck state with unlock

Because the null frame transmission does not use the MB data, you can lock and unlock the MB during null frame transmission. Figure 486 shows a transmit MB timing and state-change diagram for null frame transmission for this case.

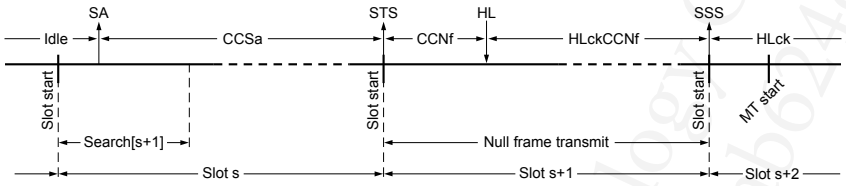


Figure 486. Null frame transmission from Idle state with locking

56.6.6.2.7 MB status update

After the end of each slot, PE generates the slot status vector. Depending on the status, the transmitted frame type, and the amount of transmitted data, the MB status is updated.

56.6.6.2.7.1 MB status update after complete message transmission

The term "complete message transmission" refers to the fact that all payload data stored in the MB were sent to the FlexRay bus. In this case, CC updates the MB slot status field and triggers the status-updated transition SU. With the SU transition, CC sets the MB interrupt flag (MBCCSR*n*[MBIF]) to indicate successful message transmission.

Depending on MBCCFR*n*[MTM], CC changes the commit flag (MBCCSR*n*[CMT]) and the valid flag (MBCCSR*n*[DVAL]). If MBCCFR*n*[MTM] = 0, the MB is in Event Transmission mode. In this case, each committed message is transmitted only one time. The commit flag is cleared with the SU transition. If MBCCFR*n*[MTM] = 1, the MB is in State Transmission mode. In this case, each committed message is transmitted as long as you provide new data or lock the MBs. CC does not clear the commit flag at the end of transmission and sets the valid flag to indicate that the message is transmitted again.

56.6.6.2.7.2 MB status update after incomplete message transmission

The term "incomplete message transmission" refers to the fact that not all payload data that must be transmitted was sent to the FlexRay bus. The following regular conditions in the dynamic segment may cause incomplete message transmission:

- The transmission slot starts in a minislot with a minislot number greater than pLatestTx.
- The transmission slot never existed in the dynamic segment.

If either of these conditions occur, the MB status does not change with the SU transition. The slot status field is not updated, the status and control flags are not changed, and the interrupt flag is not set.

Additionally, an incomplete message transmission can be caused by internal communication errors. If those errors occur, CC sets the PIFR1[PECF_IF].

56.6.6.2.7.3 MB status update after null frame transmission

After the transmission of a null frame, the status of the MB that was used for the null frame transmission does not change. The slot status field is not updated, the status and control flags are not changed, and the interrupt flag is not set.

56.6.6.3 Receive MBs

This section provides a detailed description of the functionality of the receive MBs. If you use receive MBs, you must configure the related receive shadow buffer as described in Receive shadow buffers.

You use a receive MB to receive a message from the FlexRay Bus based on individual filter criteria. CC uses the receive MB to provide the following data to you:

- Message data received
- Information about the reception process
- Status information about the slot in which the message was received

An individual MB with number *n* is configured as a receive MB by the following configuration settings shown in Equation 11:

$$MBCCSRn[MTD] = 0(receivemessagebuffer)$$

Equation 11. Configuration equation

Both you and CC have access to certain MB fields, To ensure data consistency, CC implements an MB locking scheme that controls the access to the MB data, control, and status fields.

Figure 487 shows the access regions for receive MBs. Table 495 describes the regions. If a region is active as indicated in Table 496, the access scheme given for that region applies to the MB.

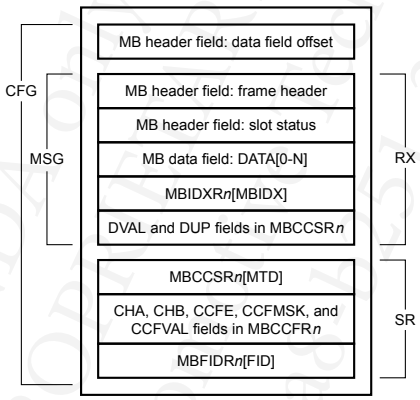


Figure 487. Receive MB access regions

Table 495. Receive MB access region description

Region	Access from		Region used for
	Application	CC	
CFG	Read and write	—	MB configuration, message data, and status access
MSG	Read and write	—	Message data, header, and status access
RX	—	Write-only	Message reception and status update
SR	—	Read-only	MB search data

In MB Configuration Control Status (MBCCSR0 - MBCCSR255):

- The EDT, LCKT, and MBIE fields in are not under access control and can be accessed from the application at any time.

- The EDS and LCKS fields are not under access control and can be accessed from CC at any time. They provide you with the required status information. The internal status information is not visible to you.
- The MBIF field is not under access control and can be accessed from the application and CC at any time. CC set access has higher priority.

CC restricts its access to the regions depending on the current MB state. You must comply with these restrictions to ensure data consistency.

Figure 488 shows the receive MB states. Table 491 describes the MB states and provides the access scheme for the access regions.

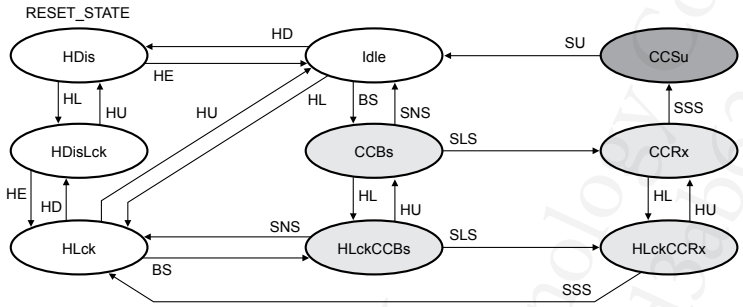


Figure 488. Receive MB states

Table 496. Receive MB states and access

State	MBCCSR*n*		Access from		Description
	EDS	LCKS	Application	Module	
Idle	1	0	—	SR	Idle MB is idle. Included in MB search.
HDis	0	0	CFG	—	Disabled MB under configuration. Excluded from MB search.
HDisLck	0	1	CFG	—	Disabled And Locked MB under configuration. Excluded from MB search.
HLck	1	1	MSG	—	Locked Application access to data, control, and status. Included in MB search.
CCBs	1	0	—	—	Buffer Subscribed MB subscribed to reception. Filter matches next (slot, cycle, channel) tuple.
HLckCCBs	1	1	MSG	—	Locked And Buffer Subscribed Application access to data, control, and status. MB subscribed to reception.
CCRx	1	0	—	—	Message Receive

Table continues on the next page...

Table 496. Receive MB states and access (continued)

State	MBCCSR*n*		Access from		Description
	EDS	LCKS	Application	Module	
					Message data received in the related shadow buffer.
HLckCCRx	1	1	MSG	—	Locked And Message Receive Application access to data, control, and status. Message data received in the related shadow buffer.
CCSu	1	0	—	RX	Status Update MB status update. Update of status flags, the slot status field, and the header index.

56.6.6.3.1 MB transitions

56.6.6.3.1.1 Application transitions

The application transitions that can be triggered by the application using the commands described in Table 492. You issue the commands by writing to MB Configuration Control Status (MBCCSR0 - MBCCSR255). One write access can issue only one command. Each command executes immediately. If the command is ignored, you must issue it again.

56.6.6.3.1.2 MB enable and disable

You issue the enable and disable commands by writing 1 to MBCCSR*n*[EDT]. The transition that each of these commands triggers depends on the current value of MBCCSR*n*[EDS]. If the command triggers the disable transition HD and the MB is in the CCBs, HLckCCBs, or CCRx state, the disable transition has no effect (the command is ignored) and the MB state does not change. You do not receive any notification.

56.6.6.3.1.3 MB lock and unlock

You issue the lock and unlock commands by writing 1 to MBCCSR*n*[LCKT]. The transition that each of these commands triggers depends on the current value of the status bit MBCCSR*n*[LCKS]. If the command triggers the HL lock transition HL while the MB is in the CCRx state, the lock transition has no effect (the command is ignored) and the MB state does not change. In this case, CC sets the MB lock error flag (CHIERFR[LCK_EF]).

Table 497. Receive MB application transitions

Transition	Host command	Condition	Description
HE	Write 1 to MBCCSR*n*[EDT]	MBCCSR*n*[EDS] = 0	Application triggers MB enable.
HD		MBCCSR*n*[EDS] = 1	Application triggers MB disable.
HL	Write 1 to MBCCSR*n*[LCKT]	MBCCSR*n*[LCKS] = 0	Application triggers MB lock.
HU		MBCCSR*n*[LCKS] = 1	Application triggers MB unlock.

56.6.6.3.1.4 Module transitions

The following table describes the module transitions that CC can trigger. Each transition triggers for certain MBs when the related condition is fulfilled.

Table 498. Receive MB module transitions

Transition	Condition	Description
BS	Slot and CycleCounter match	Buffer Subscribed The MB filter matches the next slot and cycle.
SLS	Slot start	Slot Start Start of either a static or dynamic slot.
SNS	Start of symbol window or NIT	Symbol Window Or NIT Start Start of either the symbol window or NIT
SSS	Start of slot, symbol window, or NIT	Slot Or Segment Start Start of static slot, dynamic slot, symbol window, or NIT
SU	Status updated	Status Updated Slot Status field, MB status flags, and header index updated. Interrupt flag set.

56.6.6.3.1.5 Transition priorities

You can trigger only one transition at a time. You do not need to specify priorities among them.

As shown in the following table, the module transitions have a higher priority than the application transitions. For all states except CCRx, a module transition and the application lock/unlock transition HL/HU and can execute at the same time. You reach the resulting state by applying the module transition and the application transition, in that order, to the intermediately-reached state. For example, if the MB is in the Buffer Subscribed CCBs state and the module triggers the slot start transition SLS at the same time as the application locks the MB by the HL transition, the intermediate state is CCRx and the resulting state is locked Buffer Subscribed state HLckCCRx.

Table 499. Receive MB transition priorities

State	Priorities	Description
Module vs. application		
Idle	BS > HD	Buffer Subscribed > MB Disable
HLck	BS > HD	Buffer Subscribed > MB Disable
CCRx	SSS > HL	Slot or Segment Start > MB Lock

56.6.6.3.2 Message reception

As a result of the MB search, CC changes the state of up to two enabled receive MBs from Idle or HLck state to either CCBs or HLckCCBs state by triggering the BS transition.

If the receive MBs for the next slot are assigned to both channels, then CC changes at most one receive MB to a buffer-subscribed state.

If more than one matching MB is assigned to a certain channel, then only the MB with the lowest MB number is in one of the states mentioned above.

With the start of the next static or dynamic slot, the module triggers the SLS transition. This changes the state of the subscribed receive MBs from either CCBs to CCRx or from HLckCCBs to HLckCCRx, respectively.

During the reception slot, the received frame data is written into the shadow buffers. For details on receive shadow buffers, see Receive shadow buffer concept. The data and status of the receive MBs that are in the CCRx or HLckCCRx states are not modified in the reception slot.

56.6.6.3.3 MB update

CC triggers the SSS transition at the start of any of these:

- Next static slot
- Next dynamic slot
- The symbol window
- NIT

This transition changes the state of the receiving receive MBs from either CCRx to CCSu or from HLckCCRx to HLck.

If an MB is in the HLckCCRx state, no update is performed. The received data is lost. CC indicates this by setting the FRLA_EF or FRLB_EF flags in CHI Error Flag (CHIERFR).

If an MB was in the CCRx state, it is now in the CCSu state. CC updates the MB after evaluating the PE-provided slot status. The MB update depends on the slot-status bits and the segment the MB is assigned to. This is described in Table 503.

Table 500. Receive MB update

vSS!ValidFrame	vRF!Header!NFIndicator	Update description
1	1	Valid non-null frame received • MB data field updated • Frame header field updated • Slot status field updated • DUP = 1 • DVAL = 1 • MBIF = 1
1	0	Valid null frame received • MB data field not updated • Frame header field not updated • Slot status field updated • DUP = 0 • DVAL not changed • MBIF = 1
0	0 or 1	No valid frame received

Table continues on the next page...

Table 500. Receive MB update (continued)

vSS!ValidFrame	vRF!Header! NFIndicator	Update description
		• MB data field not updated • Frame header field not updated • Slot status field updated • DUP = 0 • DVAL not changed. • MBIF = 1 if the slot was not an empty dynamic slot An empty dynamic slot occurs when all of the following are 0: • vSS!ValidFrame • vSS!SyntaxError • vSS!ContentError • vSS!BViolation

NOTE

If the number of the last slot in the current communication cycle on a given channel is *n*, then all receive MBs assigned to this channel with MBFIDR*n*[FID] > *n* are not updated.

When the receive MB update completes, CC triggers the SU transition. This changes the buffer state from CCSu to Idle. The next figure shows an example receive MB timing and state change diagram for a normal frame reception.

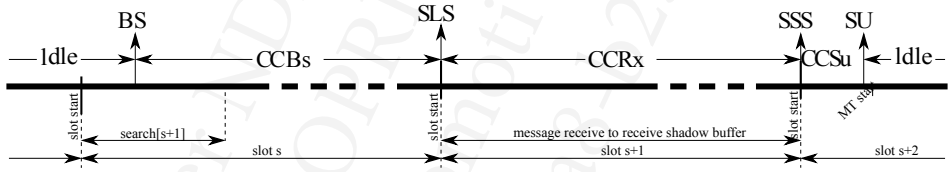


Figure 489. Message reception timing

The following considerations determine the amount of message data written into the MB data field of the receive shadow buffer:

- The MB segment that the MB is assigned to, as defined by MB Segment Size and Utilization (MBSSUTR)
- The MB data field size, as defined by the related field in MB Data Size (MBDSR)
- The number of bytes received over the FlexRay bus

If the MB is assigned to MB segment 1, and the number of received bytes is greater than 2 × MBDSR[MBSEG1DS], CC writes only 2 × MBDSR[MBSEG1DS] bytes into the MB data field of the receive shadow buffer. If the number of received bytes is less than 2 × MBDSR[MBSEG1DS], CC writes only the received number of bytes and does not change the trailing bytes in the MB data field of the receive shadow buffer. The same holds for the message buffer segment 2 with MBDSR[MBSEG2DS].

56.6.6.3.4 Received message access

To access the message data received over the FlexRay bus, you read the message data stored in the MB data field of the corresponding receive MB. Individual MBs describes the access to the MB data field.

You can read the MB data field if the receive MB is in one of these states:

- HDis
- HDisLck

- HLck

If the MB is in one of these states, CC does not change the MB content.

56.6.6.3.5 Receive shadow buffer concept

The receive shadow buffer concept applies only to individual receive MBs. The concept ensures that the receive MB presents only syntactically- and semantically-valid received non-null frames to you. Receive shadow buffers describes the basic structure of a receive shadow buffer.

The receive shadow buffers temporarily store the received frame header and message data. The slot-status information is generated after the slot boundary. If the slot-status information indicates the reception of the valid non-null frame (see Table 500), CC writes the slot status into the Slot Status field of the receive shadow buffer and exchanges the content of MB Index (MBIDXR0 - MBIDXR255) with the content of the corresponding internal shadow buffer index register. In all other cases, CC writes the slot status into the identified receive MB, depending on the slot status and the FlexRay segment the MB is assigned to.

The shadow buffer concept, with its index exchange, results in the FlexRay memory area located MB associated to an individual receive MB changes after successful reception of a valid frame. This means that the MB area in the FlexRay memory area that you access for reading the received message is different from the initial MB setting. Therefore, you must not rely on the index information written initially into MB Index (MBIDXR0 - MBIDXR255). Instead, you must fetch the index of the MB header field from MB Index (MBIDXR0 - MBIDXR255).

56.6.7 Individual MB search

This section provides a detailed description of the MB search algorithm.

For each channel, the MB search determines whether a slot *s* in a communication cycle *c* is assigned to frame or null-frame transmission or is subscribed to frame reception on that channel.

The MB search is a sequential algorithm that is invoked at the following protocol-related events:

- NIT start
- Slot start in the static segment
- Minislot start in the dynamic segment

The MB search within the NIT searches for MBs assigned or subscribed to slot 1. The MB search within slot *n* searches for MBs assigned or subscribed to slot *n+1*.

In general, the MB search for the next slot *n* considers only MBs that are both of the following:

- Enabled—that is, MBCCSR*n*[EDS] = 1
- Match the next slot *n*—that is, MBFIDRn[FID] = *n*

Table 501 introduces and defines matching MBs. For each enabled channel, the MB search may identify multiple matching MBs. CC considers all matching MBs and selects the MBs with highest priority according to Table 502 and Table 503.

Table 501. Matching MBs

For this segment:	An MB is a matching MB if it:
Static	Matches the condition of at least one row of Table 502
Dynamic	Matches the condition of at least one row of Table 503

Table 502. MB search priority (static segment)

Priority	MTD	LCKS	CMT	CCFM[1]	Description	Transition
0 (highest)	1	0	1	1	Transmit buffer, matches cycle count, not locked and committed	MA

Table continues on the next page...

Table 502. MB search priority (static segment) (continued)

Priority	MTD	LCKS	CMT	CCFM[1]	Description	Transition
1	1	—	0	1	Transmit buffer, matches cycle count, not committed	SA
	1	1	—	1	Transmit buffer, matches cycle count, locked	
2	1	—	—	—	Transmit buffer	
3	0	0	—	1	Receive buffer, matches cycle count, not locked	SB
4 (lowest)	0	1	—	1	Receive buffer, matches cycle count, locked	

1. Cycle counter filter match; see MB cycle counter filtering.

Table 503. MB search priority (dynamic segment)

Priority	MTD	LCKS	CMT	CCFM[1]	Description	Transition
0 (highest)	1	0	1	1	Transmit buffer, matches cycle count, not locked and committed	MA
1	0	0	—	1	Receive buffer, matches cycle count, not locked	SB
2 (lowest)	0	1	—	1	Receive buffer, matches cycle count, locked	

1. Cycle Counter Filter Match, see MB cycle counter filtering.

If there are multiple MBs with highest priority, CC selects the MB with the lowest MB number. All MBs that have the highest priority must have a consistent channel assignment as specified in MB channel assignment consistency.

Depending on the MB channel assignment, the same MB can be found for both channel A and channel B. In this case, use this MB as described in Individual MBs.

56.6.7.1 MB cycle counter filtering

The MB cycle counter filter is a value-mask filter defined by the CCFE, CCFMSK, and CCFVAL fields in MB Cycle Counter Filter (MBCCFR0 - MBCCFR255). This filter determines a set of communication cycles in which the MB is considered for message reception or transmission. If the cycle counter filter is disabled (MBCCFR*n*[CCFE] = 0), this set of cycles consists of all communication cycles.

If the MB cycle counter filter does not match a certain communication cycle number, this MB is not considered for message transmission or reception in that communication cycle. In case of a transmit MB assigned to a slot in the static segment, though, this buffer is added to the matching MBs to indicate the slot assignment and to trigger the null-frame transmission.

The MB cycle counter filter matches the communication cycle with the number CYCCNT if at least one of the following conditions evaluates to true:

- MBCCFR*n*[CCFE] = 0
- CYCCNT & MBCCFR*n*[CCFMSK] = MBCCFR*n*[CCFVAL] & MBCCFR*n*[CCFMSK]

56.6.7.2 MB channel assignment consistency

The MB channel assignment given by the CHA and CHB fields in MB Cycle Counter Filter (MBCCFR0 - MBCCFR255) defines the channels on which the MB receives or transmits. The MB with number *n* transmits or receives:

- On channel A if MBCCFR*n*[CHA] = 1
- On channel B if MBCCFR*n*[CHB] = 1

To ensure correct MB operation, all MBs assigned to the same slot and with the same priority must have a consistent channel assignment. That means they must be either assigned to one channel only, or must be assigned to both channels. The behavior

of the MB search is undefined if both types of channel assignments occur for one slot and priority. The following figure illustrates an inconsistent channel assignment for MB 0 and MB 1.

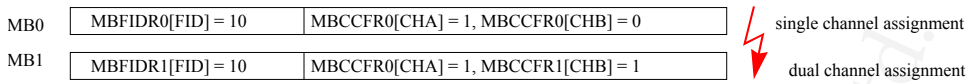


Figure 490. Inconsistent channel assignment

56.6.7.3 Node-related slot multiplexing

Node-related slot multiplexing applies only to the dynamic segment and refers to the functionality in which transmit as well as receive MBs are configured for the same slot.

According to Table 503, the transmit buffer is found only if the cycle counter filter matches and the buffer is not locked and committed. In all other cases, the receive buffer is found. Thus, if the block has no data to transmit in a dynamic slot, it can receive frames on that slot.

56.6.7.4 MB search error

This section describes the types of errors that may occur during the MB search.

FIFO reception is not affected by the search errors. Additionally, if no receive buffer has been found because of a search error, the received frame is considered for FIFO reception.

56.6.7.4.1 MB search start while running

If the MB search is running in slot *n-1* and the next MB search start event appears because of the start of slot *n*, the MB search engine stops and CC sets the Message Buffer Search Error flag (CHIERFR[MBS_EF]). As a result of this stop, no individual MB is identified for transmission or reception in slot *n*. Additionally, the search engine does not start in slot *n*, and consequently no individual MB is identified for transmission or reception in slot *n+1*.

An MB search error appears only if the CHI frequency is too slow to allow the search through all MBs to be completed within the NIT or a minislot.

For more details of the minimum required CHI frequency, see Number of Usable Message Buffers.

56.6.7.4.2 Illegal MB index found

If the MB search has finished the MB search in slot *n-1*, it retrieves the data offset values for the found MBs and the receive shadow buffers. If one of these MBs contains an illegal MB index, CC sets the Message Buffer Search Error flag (CHIERFR[MBS_EF]) and no individual MB is identified for transmission or reception in slot *n*. Receive Shadow Buffer Index (RSBIR) and MB Index (MBID XR0 - MBIDXR255) specify the legal MB index values for the individual and receive shadow buffers.

56.6.8 Individual MB reconfiguration

Individual MB reconfiguration is the ability to change the initial configuration of each individual MB even when the protocol is not in the POC:config state. The configuration bits and fields that can be changed are given in the section on specific configuration data. The common configuration data given in the section on specific configuration data cannot be reconfigured when the protocol is out of the POC:config state.

56.6.8.1 Reconfiguration Schemes

Depending on the target and destination basic state of the message buffer that is to be reconfigured, there are three reconfiguration schemes.

56.6.8.1.1 Basic Type Not Changed (RC1)

A reconfiguration will not change the basic type of the individual message buffer, if the message buffer transfer direction bit MBCCSRn[MTD] are not changed. This type of reconfiguration is denoted by RC1 in Figure 491. Transmit and receive message buffers can be RC1-reconfigured when in the HDis or HDisLck state.

56.6.8.1.2 Buffer Type Not Changed (RC2)

A reconfiguration will not change the buffer type of the individual message buffer. This type of reconfiguration is denoted by RC2 in the figure below. It applies to transmit and receive message buffers. Transmit and receive message buffers can be RC2-reconfigured when in the HDis or HDisLck state.

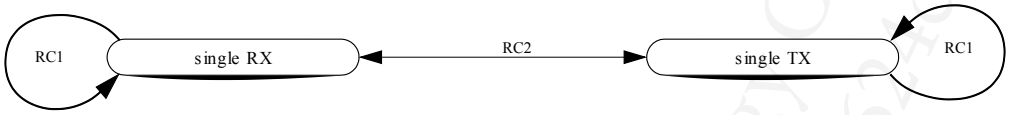


Figure 491. Message Buffer Reconfiguration Scheme

56.6.9 Receive FIFOs

This section provides the functional description of the two receive FIFOs.

56.6.9.1 Overview

The two receive FIFOs implement the queued message buffer concept defined by the *FlexRay Communications System Protocol Specification, Version 2.1 Rev A*. One FIFO is assigned to channel A, the other FIFO is assigned to channel B. Both FIFOs work completely independent from each other.

The message buffer structure of each FIFO is described in Receive FIFO. The area in the FlexRay memory area for each of the two FIFOs is characterized by:

- *The FIFO system memory base address*
- *The index of the first FIFO entry given by Receive FIFO Start Index Register (RFSIR)*
- *The data field offset of the data field belonging to the first FIFO entry given by Receive FIFO Start Data Offset Register (RFSDOR)*
- *The number of FIFO entries and the length of each FIFO entry as given by Receive FIFO Depth and Size Register (RFDSR)*

56.6.9.2 FIFO Configuration

The FIFOs can be configured for two different locations of the system memory base address via the FIFO address mode bit FAM in the Module Configuration Register (MCR).

56.6.9.2.1 Single System Memory Base Address Mode

This mode is configured, when the FIFO address mode flag MCR[FAM] is set to 0. In this mode, the location of the system memory base address for the FIFO buffers is System Memory Base Address Register (SYMBADR).

56.6.9.2.2 Dual System Memory Base Address Mode

This mode is configured, when the FIFO address mode flag MCR[FAM] is set to 1. In this mode, the location of the system memory base address for the FIFO buffers is Receive FIFO System Memory Base Address Register (RFSYMBADR).

The FIFO control and configuration data are given in Receive FIFO control and configuration data. The configuration of the FIFOs consists of two steps.

The first step is the allocation of the required amount of FlexRay memory area for the FlexRay window. This includes the allocation of the message buffer header area and the allocation of the message buffer data fields. For more details see FlexRay memory area layout.

The second step is the programming of the configuration data register while the PE is in *POC:config*.

The following steps configure the layout of the FIFO.

- Configure the FIFO update and address modes in Module Configuration Register (MCR)
- Configure the FIFO system memory base address
- Configure the Receive FIFO Start Index Register (RFSIR) with the first message buffer header index that belongs to the FIFO
- Configure the Receive FIFO Start Data Offset Register (RFSDOR) with the data field offset of the data field belonging to the first message buffer that belongs to the FIFO
- Configure the Receive FIFO Depth and Size Register (RFDSR) with FIFO entry size
- Configure the Receive FIFO Depth and Size Register (RFDSR) with FIFO depth
- Configure the FIFO Filters

56.6.9.3 FIFO Periodic Timer

The FIFO periodic timer is used to generate an FIFO almost-full interrupt at certain point in time, if the almost-full watermark is not reached, but the FIFO is not empty. This can be used to prevent frames from get stuck in the FIFO for a long time.

The FIFO periodic timer is configured via the Receive FIFO Periodic Timer Register (RFPTR). If the periodic timer duration RFPTR[PTD] is configured to 0x0000, the periodic timer is continuously expired. If the periodic timer duration RFPTR[PTD] is configured to 0x3FFF, the periodic timer never expires. If the periodic timer is configured to a value *ptd*, greater than 0x0000 and smaller 0x3FFF, the periodic timer expires and is restarted at the start of every communication cycle, and expires and is restarted after *ptd* macroticks have been elapsed.

56.6.9.4 FIFO Reception

The FIFO reception is a CC internal operation.

A message frame reception is directed into the FIFO, if no individual message buffer is assigned for transmission or subscribed for reception for the current slot. In this case the FIFO filter path shown in Figure 492 is activated.

If the FIFO filter path indicates that the received frame has to be appended to the FIFO and the FIFO is not full, the CC writes the received frame header into the message buffer header field indicated by the CC internal FIFO write index. The frame payload data are written into the corresponding message buffer data field. If the status of the received frame indicates a valid non-null frame, the slot status information is written into the message buffer header field and the CC internal FIFO write index is updated by 1 and the fifo fill level FLA (FLB) in the Receive FIFO Fill Level and POP Count Register (RFFLPCR) is incremented. If the status of the received frame indicates an invalid or null frame, the frame is not appended to the FIFO.

56.6.9.5 FIFO Almost-Full Interrupt Generation

If the fifo fill level FLA (FLB) is updated after a frame reception and exceeds the FIFO watermark level WM, i.e. FLA>WM_A (FLB>WM_B), then the FIFO almost-full interrupt flag GIFER[FAFAIF] (GIFER[FAFBIF]) is asserted.

If the periodic timer expires, and FIFOA (FIFOB) is not empty, i.e. FLA>0 (FLB>0), then the FIFO almost-full interrupt flag GIFER[FAFAIF] (GIFER[FAFBIF]) is asserted.

56.6.9.6 FIFO Overflow Error Generation

If the FIFOA (FIFOB) is full, i.e. FLA=$FIFO\_DEPTH_A$ (FLB=$FIFO\_DEPTH_B$) and the conditions for a FIFO reception as described in FIFO Reception are fulfilled, then the fifo overflow error flag CHIERFR[FOVA_EF] (CHIERFR[FOVB_EF]) is asserted.

56.6.9.7 FIFO Message Access

The FIFOA (FIFOB) contains valid messages if the FIFO fill level given in the fields FLA (FLB) in the Receive FIFO Fill Level and POP Count Register (RFFLPCR) is greater than 0. The Receive FIFO A Read Index Register (RFARIR) and the (Receive

FIFO B Read Index Register (RFBRIR)) point to a message buffer with valid content and the oldest frames stored in the FIFO.The respective read data field offsets can be calculated according to Equation 6.

If the FIFO fill level FLA (FLB) in the Receive FIFO Fill Level and POP Count Register (RFFLPCR) is 0, than the FIFOA (FIFOB) contains no valid messages and the corresponding read index register Receive FIFO A Read Index Register (RFARIR) or (Receive FIFO B Read Index Register (RFBRIR) point to a message buffer with invalid content. In this case the application must not read data from this FIFO.

To access the oldest message in the FIFOA (FIFOB), the application first reads the read index RDIDX out of the Receive FIFO A Read Index Register (RFARIR) (Receive FIFO B Read Index Register (RFBRIR)). This read index points to the message buffer header field of the oldest message buffer that contains valid received message data. The data field offset belonging to this message buffer must be calculated by the application according to Equation 6. The application can access the message data as described in Receive FIFO. When the application has read the message buffer data and status information, it can update the FIFO as described in FIFO Update.

56.6.9.8 FIFO Update

The application updates the FIFOA (FIFOB) by writing a pop count value *pc* different from 0 to the PCA (PCB) field in the Receive FIFO Fill Level and POP Count Register (RFFLPCR).

As a result of the this operation, the CC removes the oldest *pc* entries from FIFOA (FIFOB).

If the specified pop count value *pc* is greater than the current fill level *fl* provided in FLA (FAB) field, then only *fl* entries are removed from the FIFOA (FIFOB), the remaining *fl-pc* requested pop operations are discarded without any notification. In this case FIFOA (FIFOB) is empty after the update operation.

The read index in the Receive FIFO A Read Index Register (RFARIR) (Receive FIFO B Read Index Register (RFBRIR)) is incremented by the number of removed items. If the read index reaches the top of the FIFO, it wraps around to the FIFO start index defined in Receive FIFO Start Index Register (RFSIR) automatically.

56.6.9.8.1 FIFO Interrupt Flag Update

The FIFO Interrupt Flag Update mode is configured, when the FIFO update mode flag MCR[FUM] is set to 0. In this mode FIFOA (FIFOB) will be updated by 1 entry, when the interrupt flag GIFER[FAFAIF] (GIFER[FAFBIF]) is written with 1 by the application.

If the FIFO is empty, the update request is ignored without any notification.

The read index in the Receive FIFO A Read Index Register (RFARIR) (Receive FIFO B Read Index Register (RFBRIR)) is incremented by 1, if the FIFO was not empty. If the read index reaches the top of the FIFO, it wraps around to the FIFO start index automatically.

56.6.9.9 FIFO Filtering

The FIFO filtering is activated after all enabled individual receive message buffers have been searched without success for a message buffer to receive the current frame.

The CC provides three sets of FIFO filters. The FIFO filters are applied to valid non-null frames only. The FIFO will not receive invalid or null-frames. For each FIFO filter, the pass criteria is specified in the related section given below. Only frames that have passed all filters will be appended to the FIFO. The FIFO filter path is depicted in the figure below.

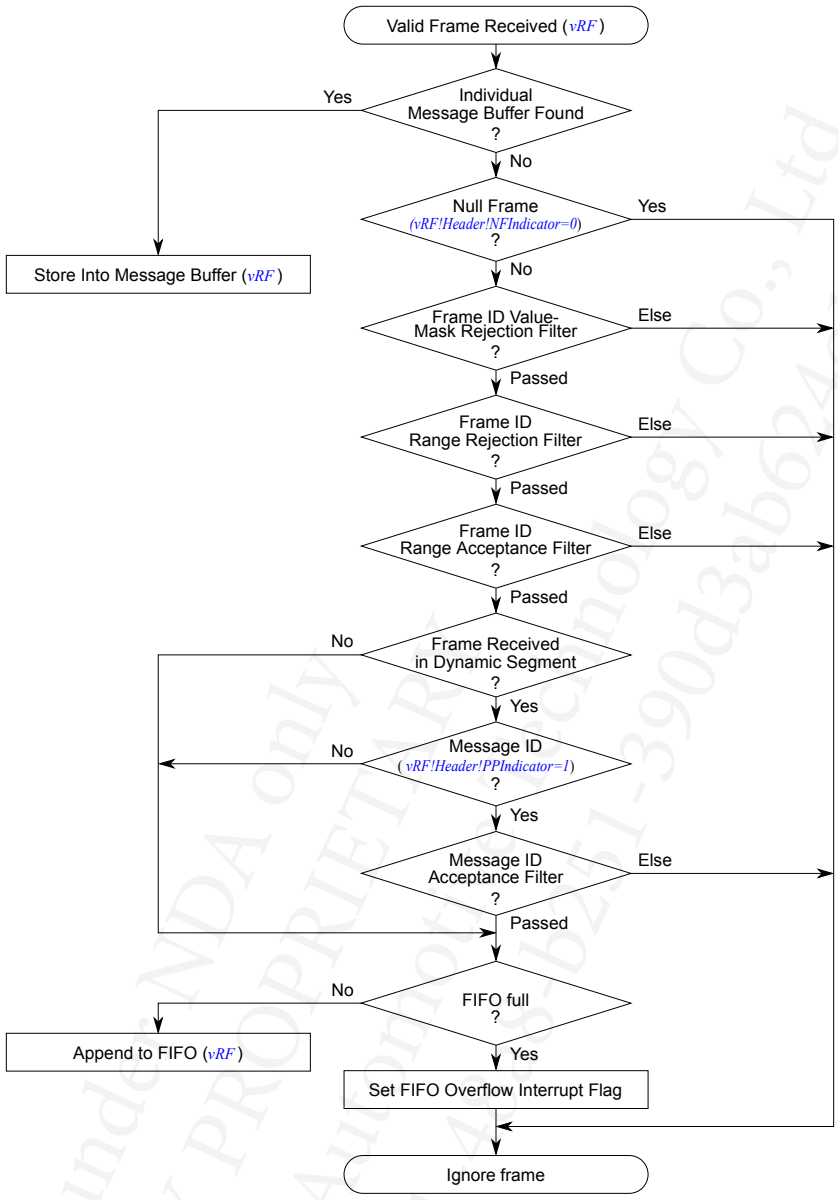


Figure 492. Received Frame FIFO Filter Path

A received frame passes the FIFO filtering if it has passed all three type of filter.

56.6.9.9.1 RX FIFO Frame ID Value-Mask Rejection Filter

The frame ID value-mask rejection filter is a value-mask filter and is defined by the fields in the Receive FIFO Frame ID Rejection Filter Value Register (RFFIDRFVR) and the Receive FIFO Frame ID Rejection Filter Mask Register (RFFIDRFMR). Each received frame with a frame ID FID that does not match the value-mask filter value passes the filter, i.e. is not rejected.

Consequently, a received valid frame with the frame ID FID passes the RX FIFO Frame ID Value-Mask Rejection Filter if the following equation is fulfilled.

$$FID \,\&\, RFFIDRFMR[FIDRFMSK] \neq RFFIDRFVR[FIDRFVAL] \,\&\, RFFIDRFMR[FIDRFMSK]$$

The RX FIFO Frame ID Value-Mask Rejection Filter can be configured to pass all frames by the following settings.

$$RFFIDRFVR[FIDRFVAL] = 0x000 \text{ and } RFFIDRFMR[FIDRFMSK] = 0x7FF$$

Using the settings above, only the frame with frame ID 0 will be rejected, which is an invalid frame. All other frames will pass.

The RX FIFO Frame ID Value-Mask Rejection Filter can be configured to reject all frames by the following settings.

$$RFFIDRFMR[FIDRFMSK] = 0x000$$

Using these settings, the first equation can never be fulfilled (0!= 0) and thus all frames are rejected; no frame will pass. This is the reset value for the RX FIFO.

56.6.9.9.2 RX FIFO Frame ID Range Rejection Filter

Each of the four RX FIFO Frame ID Range filters can be configured as a rejection filter. The filters are configured by the Receive FIFO Range Filter Configuration Register (RFRFCFR) and controlled by the Receive FIFO Range Filter Control Register (RFRFCTR). The RX FIFO Frame ID range filters apply to all received valid frames. A received frame with the frame ID FID passes the RX FIFO Frame ID Range rejection filters if either no rejection filter is enabled, or, for all of the enabled RX FIFO Frame ID Range rejection filters, i.e. RFRFCTR[FiMD] = 1 and RFRFCTR[FiEN] = 1, fulfilling the following equation: .

$$FID < RFRFCFR_{SEL}[SID_{IBD=0}]) or (RFRFCFR_{SEL}[SID_{IBD=1}] < FID)$$

Consequently, all frames with a frame ID that fulfills the following equation for at least one of the enabled rejection filters will be rejected and thus not pass.

$$RFRFCFR_{SEL}[SID_{IBD=0}] \leq FID \leq RFRFCFR_{SEL}[SID_{IBD=1}]$$

56.6.9.9.3 RX FIFO Frame ID Range Acceptance filter

Each of the four RX FIFO Frame ID Range filters can be configured as an acceptance filter. The filters are configured by the Receive FIFO Range Filter Configuration Register (RFRFCFR) and controlled by the Receive FIFO Range Filter Control Register (RFRFCTR). The RX FIFO Frame ID range filters apply to all received valid frames. A received frame with the frame ID FID passes the RX FIFO Frame ID Range acceptance filters if either no acceptance filter is enabled, or, for at least one of the enabled RX FIFO Frame ID Range acceptance filters, i.e. RFRFCTR[FiMD] = 0 and RFRFCTR[FiEN] = 1, the following equation is fulfilled.

$$RFRFCFR_{SEL}[SID_{IBD=0}] \leq FID \leq RFRFCFR_{SEL}[SID_{IBD=1}]$$

56.6.9.9.4 RX FIFO Message ID Acceptance Filter

The RX FIFO Message ID Acceptance Filter is a value-mask filter and is defined by the Receive FIFO Message ID Acceptance Filter Value Register (RFMIDAFVR) and the Receive FIFO Message ID Acceptance Filter Mask Register (RFMIDAFMR). This filter applies only to valid frames received in the dynamic segment with the payload preamble indicator bit PPI set to 1. All other frames will pass this filter.

A received valid frame in the dynamic segment with the payload preamble indicator bit PPI set to 1 and with the message ID MID (the first two bytes of the payload) will pass the RX FIFO Message ID Acceptance Filter if the following equation is fulfilled.

$$MID \& RFMIDAFMR[MIDAFMSK] = RFMIDAFMR[MIDAFVAL] \& RFMIDAFMR[MIDAFMSK]$$

The RX FIFO Message ID Acceptance Filter can be configured to accept all frames by setting

$$RFMIDAFMR[MIDAFMSK]: = 0x000$$

Using these settings, the first equation is always fulfilled and all frames will pass.

56.6.10 Channel Device Modes

This section describes the two FlexRay channel device modes that are supported by the CC.

56.6.10.1 Dual Channel Device Mode

In the dual channel device mode, both FlexRay ports are connected to physical FlexRay bus lines. The FlexRay port consisting of FR_A_RX, FR_A_TX, and FR_A_TX_EN_b is connected to the physical bus channel A and the FlexRay port consisting of FR_B_RX, FR_B_TX, and FR_B_TX_EN_b is connected to the physical bus channel B. The dual channel system is shown in the following figure.

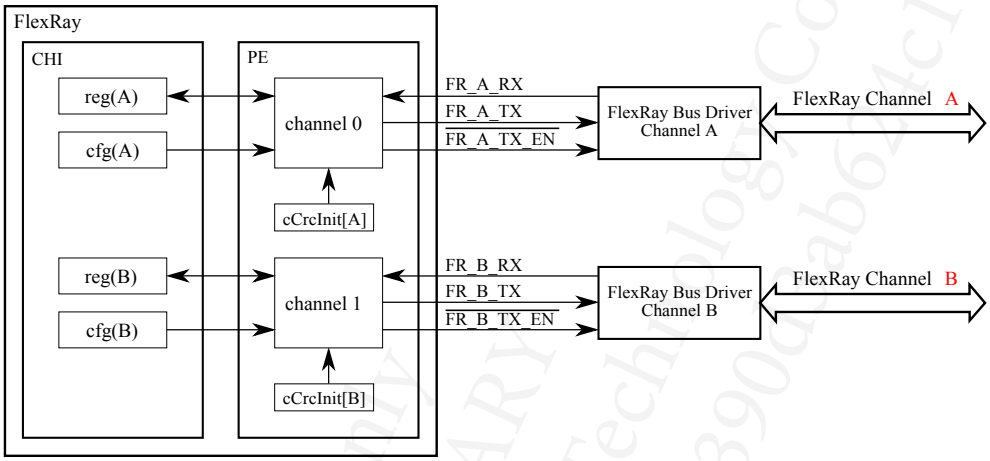


Figure 493. Dual Channel Device Mode

56.6.10.2 Single Channel Device Mode

The single channel device mode supports devices that have only one FlexRay port available. This FlexRay port consists of the signals FR_A_RX, FR_A_TX, and FR_A_TX_EN_b and can be connected to either the physical bus channel A (shown in the "Single Channel Device Mode (Channel A)" figure below) or the physical bus channel B (shown in the "Single Channel Device Mode (Channel B)" figure below).

If the device is configured as a single channel device by setting MCR[SCM] to 1, only the internal channel A and the FlexRay Port A is used. Depending on the setting of MCR[CHA] and MCR[CHB], the internal channel A behaves either as a FlexRay Channel A or FlexRay Channel B. The bit MCR[CHA] must be set, if the FlexRay Port A is connected to a FlexRay Channel A. The bit MCR[CHB] must be set if the FlexRay Port A is connected to a FlexRay Channel B. The two FlexRay channels differ only in the initial value for the frame CRC *cCrcInit*. For a single channel device, the application can access and configure only the registers related to internal channel A.

Provided by IHS Markit
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a8-251-390dab624c1

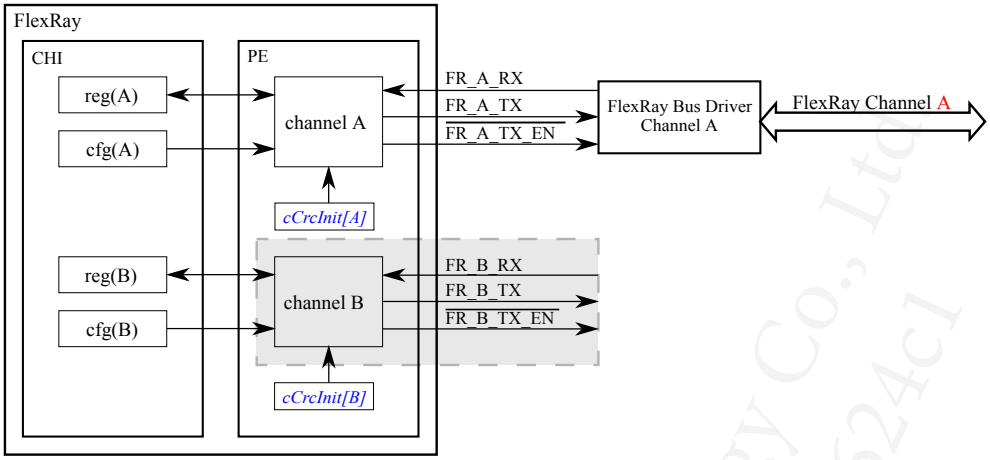


Figure 494. Single Channel Device Mode (Channel A)

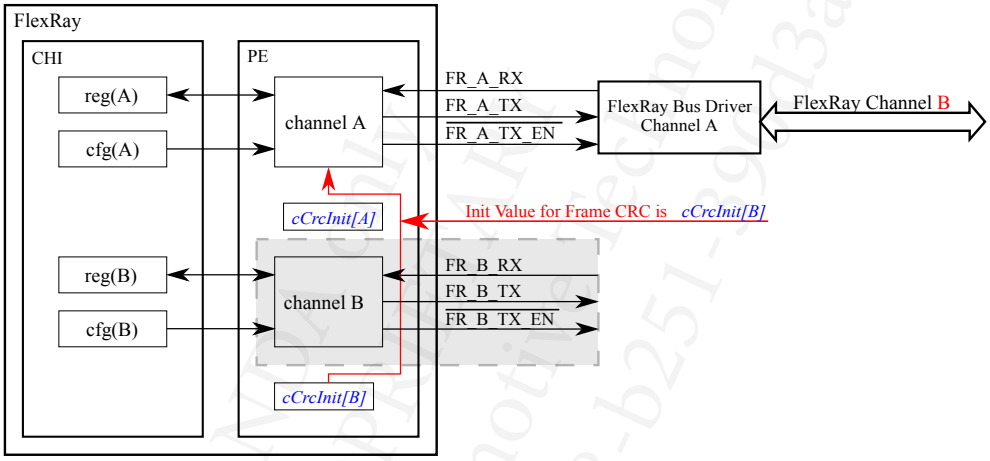


Figure 495. Single Channel Device Mode (Channel B)

56.6.11 External Clock Synchronization

The application of the external rate and offset correction is triggered when the application writes to the EOC_AP and ERC_AP fields in the Protocol Operation Control Register (POCR). The PE applies the external correction values in the next even-odd cycle pair as shown in the "External Offset Correction Write and Application Timing" figure and the "External Rate Correction Write and Application Timing" figures below.

NOTE

The values provided in the EOC_AP and ERC_AP fields are the values that were written from the application most recently. If these value were already applied, they will not be applied in the current cycle pair again.

If the offset correction applied in the NIT of cycle 2n+1 shall be affect by the external offset correction, the EOC_AP field must be written to after the start of cycle 2n and before the end of the static segment of cycle 2n+1. If this field is written to after the end of the static segment of cycle 2n+1, it is not guaranteed that the external correction value is applied in cycle 2n+1. If the value is not applied in cycle 2n+1, then the value will be applied in the cycle 2n+3. Refer to the following for timing details.

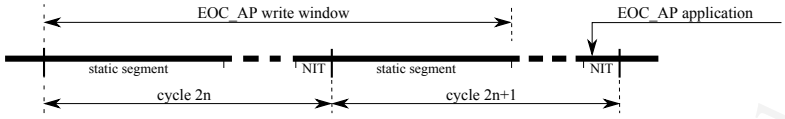


Figure 496. External Offset Correction Write and Application Timing

If the rate correction for the cycle pair [2n+2, 2n+3] shall be affect by the external offset correction, the ERC_AP field must be written to after the start of cycle 2n and before the end of the static segment start of cycle 2n+1. If this field is written to after the end of the static segment of cycle 2n+1, it is not guaranteed that the external correction value is applied in cycle pair [2n+2, 2n+3]. If the value is not applied for cycle pair [2n+2, 2n+3], then the value will be applied for cycle pair [2n+4, 2n+5]. Refer to the following for details.

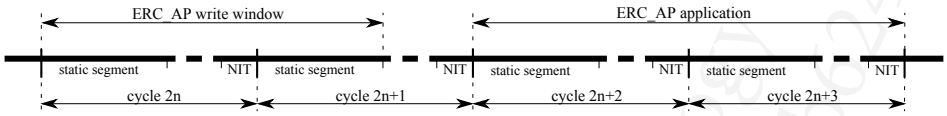


Figure 497. External Rate Correction Write and Application Timing

56.6.12 Sync Frame ID and Sync Frame Deviation Tables

The FlexRay protocol requires the provision of a snapshot of the Synchronization Frame ID tables for the even and odd communication cycle for both channels. The CC provides the means to write a copy of these internal tables into the FlexRay memory area and ensures application access to consistent tables by means of table locking. Once the application has locked the table successfully, the CC will not overwrite these tables and the application can read a consistent snapshot.

NOTE

Only synchronization frames that have passed the synchronization frame filters are considered for clock synchronization and appear in the sync frame tables.

56.6.12.1 Sync Frame ID Table Content

The Sync Frame ID Table is a snapshot of the protocol related variables *vsSyncIdListA* and *vsSyncIdListB* for each even and odd communication cycle. This table provides a list of the frame IDs of the synchronization frames received on the corresponding channel and cycle that are used for the clock synchronization.

56.6.12.2 Sync Frame Deviation Table Content

The Sync Frame Deviation Table is a snapshot of the protocol related variable zsDev(id)(oe)(ch)!Value. Each Sync Frame Deviation Table entry provides the deviation value for the sync frame, with the frame ID presented in the corresponding entry in the Sync Frame ID Table.

Provided under NXP SLA CLICK-THROUGH LICENSE
COMPANY CONFIDENTIAL PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482a-482a-b251-0d3ab624c

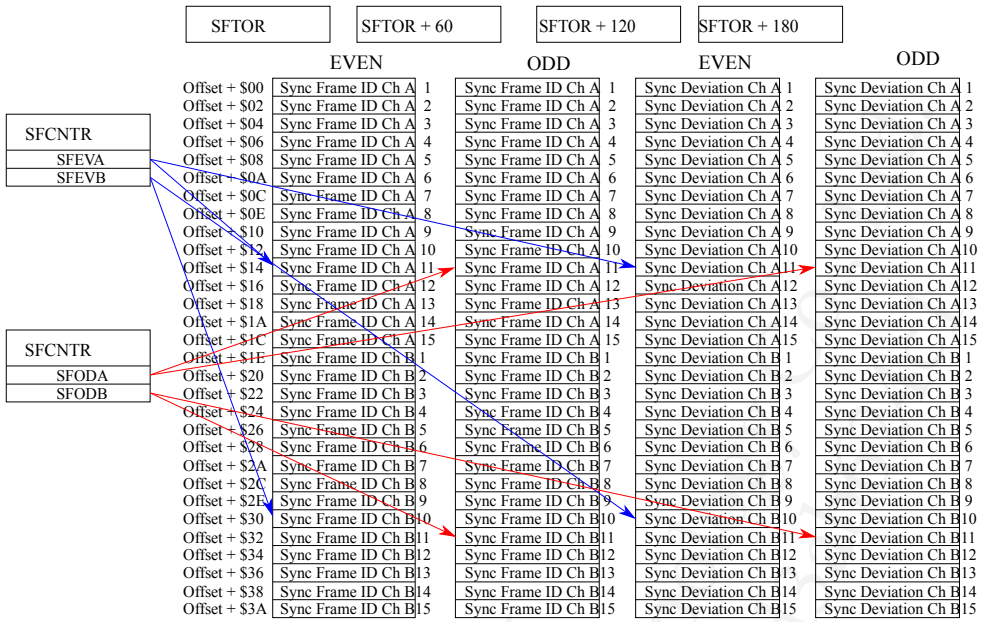


Figure 498. Sync Table Memory Layout

56.6.12.3 Sync Frame ID and Sync Frame Deviation Table Setup

The CC writes a copy of the internal synchronization frame ID and deviation tables into the FlexRay memory area if requested by the application. The application must provide the appropriate amount of FlexRay memory area for the tables. The memory layout of the tables is given in Figure 498. Each table occupies 120 16-bit entries.

While the protocol is in *POC:config* state, the application must program the offsets for the tables into the Sync Frame Table Offset Register (SFTOR).

56.6.12.4 Sync Frame ID and Sync Frame Deviation Table Generation

The application controls the generation process of the Sync Frame ID and Sync Frame Deviation Tables into the FlexRay memory area using the Sync Frame Table Configuration, Control, Status Register (SFTCCSR). A summary of the copy modes is given in the table below.

Table 504. Sync Frame Table Generation Modes

SFTCCSR			Description
OPT	SDVEN	SIDEN	
0	0	0	No Sync Frame Table copy
0	0	1	Sync Frame ID Tables will be copied continuously
0	1	0	Reserved
0	1	1	Sync Frame ID Tables and Sync Frame Deviation Tables will be copied continuously
1	0	0	No Sync Frame Table copy
1	0	1	Sync Frame ID Tables for next even-odd-cycle pair will be copied

Table continues on the next page...

Table 504. Sync Frame Table Generation Modes (continued)

SFTCCSR			Description
OPT	SDVEN	SIDEN	
1	1	0	Reserved
1	1	1	Sync Frame ID Tables and Sync Frame Deviation Tables for next even-odd-cycle pair will be copied

The Sync Frame Table generation process is described in the following for the even cycle. The same sequence applies to the odd cycle.

If the application has enabled the sync frame table generation by setting SFTCCSR[SIDEN] to 1, the CC starts the update of the even cycle related tables after the start of the NIT of the next even cycle. The CC checks if the application has locked the tables by reading the SFTCCSR[ELKS] lock status bit. If this bit is set, the CC will not update the table in this cycle. If this bit is cleared, the CC locks this table and starts the table update. To indicate that these tables are currently updated and may contain inconsistent data, the CC clears the even table valid status bit SFTCCSR[EVAL]. Once all table entries related to the even cycle have been transferred into the FlexRay memory area, the CC sets the even table valid bit SFTCCSR[EVAL] and the Even Cycle Table Written Interrupt Flag EVT_IF in the Protocol Interrupt Flag Register 1 (PIFR1). If the interrupt enable flag EVT_IE is set, an interrupt request is generated.

To read the generated tables, the application must lock the tables to prevent the CC from updating these tables. The locking is initiated by writing a 1 to the even table lock trigger SFTCCSR[ELKT]. When the even table is not currently updated by the CC, the lock is granted and the even table lock status bit SFTCCSR[ELKS] is set. This indicates that the application has successfully locked the even sync tables and the corresponding status information fields SFRA, SFRB in the Sync Frame Counter Register (SFCNTR). The value in the SFTCCSR[CYCNUM] field provides the number of the cycle that this table is related to.

The number of available table entries per channel is provided in the SFCNTR[SFEVA] and SFCNTR[SFEVB] fields. The application can now start to read the sync table data from the locations given in Figure 498.

After reading all the data from the locked tables, the application must unlock the table by writing to the even table lock trigger SFTCCSR[ELKT] again. The even table lock status bit SFTCCSR[ELKS] is reset immediately.

If the sync frame table generation is disabled, the table valid bits SFTCCSR[EVAL] and SFTCCSR[EVAL] are reset when the counter values in the Sync Frame Counter Register (SFCNTR) are updated. This is done because the tables stored in the FlexRay memory area are no longer related to the values in the Sync Frame Counter Register (SFCNTR).

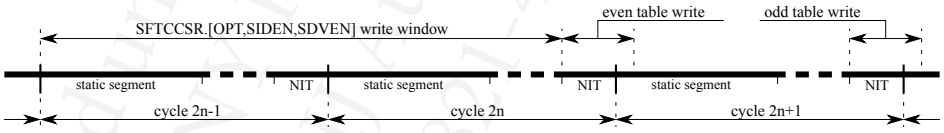


Figure 499. Sync Frame Table Trigger and Generation Timing

56.6.12.5 Sync Frame Table Access

The sync frame tables will be transferred into the FlexRay memory area during the table write windows shown in Figure 498. During the table write, the application can not lock the table that is currently written. If the application locks the table outside of the table write window, the lock is granted immediately.

56.6.12.5.1 Sync Frame Table Locking and Unlocking

The application locks the even/odd sync frame table by writing 1 to the lock trigger bit ELKT/OLKT in the Sync Frame Table Configuration, Control, Status Register (SFTCCSR). If the affected table is not currently written to the FlexRay memory area, the lock is granted immediately, and the lock status bit ELKS/OLKS is set. If the affected table is currently written to the FlexRay memory area, the lock is not granted. In this case, the application must issue the lock request again until the lock is granted.

The application unlocks the even/odd sync frame table by writing 1 to the lock trigger bit ELKT/OLKT. The lock status bit ELKS/OLKS is cleared immediately.

56.6.13 MTS Generation

The CC provides a flexible means to request the transmission of the Media Access Test Symbol MTS in the symbol window on channel A or channel B.

The application can configure the set of communication cycles in which the MTS will be transmitted over the FlexRay bus by programming the CYCCNTMSK and CYCCNTVAL fields in the MTS A Configuration Register (MTSACFR) and MTS B Configuration Register (MTSBCFR).

The application enables or disables the generation of the MTS on either channel by setting or clearing the MTE control bit in the MTS A Configuration Register (MTSACFR) or MTS B Configuration Register (MTSBCFR). If an MTS is to be transmitted in a certain communication cycle, the application must set the MTE control bit during the static segment of the preceding communication cycle.

The MTS is transmitted over channel A in the communication cycle with number CYCCNT, if the following equations are fulfilled.

PSR0[PROTSTATE] = POC: normalactive
MTSACRF[MTE] = 1
CYCCNT & MTSACFR[CYCCNTMSK] = MTSACFR[CYCCNTVAL] & MTSACFR[CYCCNTMSK]

The MTS is transmitted over channel B in the communication cycle with number CYCCNT, if the following equations are fulfilled.

MTSBCRF[MTE] = 1
CYCCNT & MTSBCFR[CYCCNTMSK] = MTSBCFR[CYCCNTVAL] & MTSBCFR[CYCCNTMSK]

56.6.14 Key Slot Transmission

Key slot assignment, transmission in POC:startup state, and transmission in POC:normal active state are discussed in this section.

56.6.14.1 Key Slot Assignment

A key slot is assigned to the CC if the key_slot_id field in the Protocol Configuration Register 18 (PCR18) is configured with a value greater than 0 and less or equal to number_of_static_slots in Protocol Configuration Register 2 (PCR2), otherwise no key slot is assigned.

56.6.14.2 Key Slot Transmission in POC:startup

If a key slot is assigned and the CC is in the *POC:startup* state, startup null frames will be transmitted as specified by FlexRay Communications System Protocol Specification, Version *2.1 Rev A*.

56.6.14.3 Key Slot Transmission in POC:normal active

If a key slot is assigned and the CC is in *POC:normal active*, a frame of the type as shown in the following table is transmitted. If a transmit message buffer is configured for the key slot and a valid message is available, a message frame is transmitted (see Message transmission). If no transmit message buffer is configured for the key slot or no valid message is available, a null frame is transmitted (see Null frame transmission).

Table 505. Key Slot Frame Type

PCR11[key_slot_used_for_sync]	PCR11[key_slot_used_for_startup]	key slot frame type
0	0	normal frame
0	1	normal frame1[1]
1	0	sync frame
1	1	startup frame

1. The frame transmitted has an semantically incorrect header and will be detected as an invalid frame at the receiver.

56.6.15 Sync Frame Filtering

Each received synchronization frame must pass the Sync Frame Acceptance Filter and the Sync Frame Rejection Filter before it is considered for clock synchronization. If the synchronization frame filtering is globally disabled, i.e. the SFFE control bit in the Module Configuration Register (MCR) is cleared, all received synchronization frames are considered for clock synchronization. If a received synchronization frame did not pass at least one of the two filters, this frame is processed as a normal frame and is not considered for clock synchronization.

56.6.15.1 Sync Frame Acceptance Filtering

The synchronization frame acceptance filter is implemented as a value-mask filter. The value is configured in the Sync Frame ID Acceptance Filter Value Register (SFIDAFVR) and the mask is configured in the Sync Frame ID Acceptance Filter Mask Register (SFIDAFMR). A received synchronization frame with the frame ID FID passes the sync frame acceptance filter, if the following equations evaluates to true.

$MCR[SFFE] = 0$
$FID \,\&\, SFIDAFMR[FMSK] = SFIDAFVR[FVAL] \,\&\, SFIDAFMR[FMSK]$

NOTE

Sync frames are transmitted in the static segment only. Thus FID <= 1023.

56.6.15.2 Sync Frame Rejection Filtering

The synchronization frame rejection filter is a comparator. The compare value is defined by the Sync Frame ID Rejection Filter Register (SFIDRFR). A received synchronization frame with the frame ID FID passes the sync frame rejection filter if the following equations evaluates to true.

$MCR[SFFE] = 0$
$FID \neq SFIDRFR[SYNFRID]$

NOTE

Sync frames are transmitted in the static segment only. Thus FID <= 1023.

56.6.16 Strobe Signal Support

The CC provides a number of strobe signals for observing internal protocol timing related signals in the protocol engine. The signals are listed and described in the Strobe Signal Mapping table, located in the Register Descriptions section.

56.6.16.1 Strobe Signal Assignment

Each of the strobe signals listed in the Strobe Signal Mapping table, located in the Register Descriptions section, can be assigned to one of the four strobe ports using the Strobe Signal Control Register (STBSCR). To assign multiple strobe signals, the application must write multiple times to the Strobe Signal Control Register (STBSCR) with appropriate settings.

To read out the current settings for a strobe signal with number N, the application must execute the following sequence.

1. Write to STBSCR with WMD = 1 and SEL = N. (updates SEL field only)
2. Read STBCSR.

   The SEL field provides N and the ENB and STBPSEL fields provides the settings for signal N.

56.6.16.2 Strobe Signal Timing

This section provides detailed timing information of the strobe signals with respect to the protocol engine clock.

The strobe signals display internal PE signals. Due to the internal architecture of the PE, some signals are generated several PE clock cycles before the actual action is performed on the FlexRay Bus. These signals are listed in the Strobe Signal Mapping table, located in the Register Descriptions section, with a negative clock offset. An example waveform is given in the following figure.

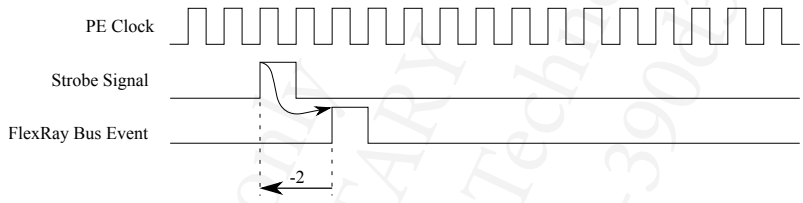


Figure 500. Strobe Signal Timing (type = pulse, clk_offset = -2)

Other signals refer to events that occurred on the FlexRay Bus some cycles before the strobe signal is changed. These signals are listed in the Strobe Signal Mapping table, located in the Register Descriptions section, with a positive clock offset. An example waveform is given in the figure below.

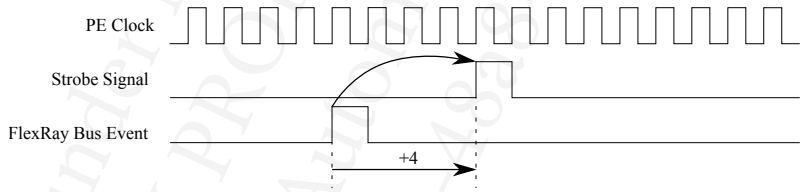


Figure 501. Strobe Signal Timing (type = pulse, clk_offset = +4)

56.6.17 Timer Support

The CC provides two timers, which run on the FlexRay time base. Each timer generates a maskable interrupt when it reaches a configured point in time. Timer T1 is an absolute timer. Timer T2 can be configured to be an absolute or a relative timer. Both timers can be configured to be repetitive. In the non-repetitive mode, timer stops if it expires. In repetitive mode, timer is restarted when it expires.

Both timers are active only when the protocol is in *POC:normal active* or *POC:normal passive* state. If the protocol is not in one of these modes, the timers are stopped. The application must restart the timers when the protocol has reached the *POC:normal active* or *POC:normal passive* state.

56.6.17.1 Absolute Timer T1

The absolute timer T1 has the protocol cycle count and the macrotick count as the time base. The timer 1 interrupt flag TI1_IF in the Protocol Interrupt Flag Register 0 (PIFR0) is set at the macrotick start event, if the following equations are fulfilled.

CYCTR[CTCCNT] & T1ICYSR[T1_CYC_MSK] = T1ICYSR[T1_CYC_VAL] & T1ICYSR[T1_CYC_MSK]
MTCTR[MTCT] = T1IMTOR[T1_MTOFFSET]

If the timer 1 interrupt enable bit TI1_IE in the Protocol Interrupt Enable Register 0 (PIER0) is asserted, an interrupt request is generated.

The status bit T1ST is set when the timer is triggered, and is cleared when the timer expires and is non-repetitive. If the timer expires but is repetitive, the T1ST bit is not cleared and the timer is restarted immediately. The T1ST is cleared when the timer is stopped.

The corresponding Interrupt condition (i.e., when the abovementioned equations are fulfilled) also leads to an event out indication on an output port (*fr_evt_tim1*) based on the [TIM1_EE] bit of the Protocol Event Output Enable Register (PEOER).

56.6.17.2 Absolute / Relative Timer T2

The timer T2 can be configured to be an absolute or relative timer by setting the T2_CFG control bit in the Timer Configuration and Control Register (TICCR). The status bit T2ST is set when the timer is triggered, and is cleared when the timer expires and is non-repetitive. If the timer expires but is repetitive, the T2ST bit is not cleared and the timer is restarted immediately. The T2ST is cleared when the timer is stopped.

56.6.17.2.1 Absolute Timer T2

If timer T2 is configured as an absolute timer, it has the same functionality timer T1 but the configuration from Timer 2 Configuration Register 0 (TI2CR0) and Timer 2 Configuration Register 1 (TI2CR1) is used. On expiration of timer T2, the interrupt flag TI2_IF in the Protocol Interrupt Flag Register 0 (PIFR0) is set. If the timer 1 interrupt enable bit TI1_IE in the Protocol Interrupt Enable Register 0 (PIER0) is asserted, an interrupt request is generated.

The corresponding Interrupt condition for T2 also leads to an event out indication on an output port (*fr_evt_tim2*) based on the [TIM2_EE] bit of the Protocol Event Output Enable Register (PEOER).

56.6.17.2.2 Relative Timer T2

If the timer T2 is configured as a relative timer, the interrupt flag TI2_IF in the Protocol Interrupt Flag Register 0 (PIFR0) is set, when the programmed amount of macroticks MT[31:0], defined by Timer 2 Configuration Register 0 (TI2CR0) and Timer 2 Configuration Register 1 (TI2CR1), has expired since the trigger or restart of timer 2. The relative timer is implemented as a down counter and expires when it has reached 0. At the macrotick start event, the value of MT[31:0] is checked and then decremented. Thus, if the timer is started with MT[31:0] == 0, it expires at the next macrotick start.

The corresponding Interrupt condition for T2 also leads to an event out indication on an output port (*fr_evt_tim2*) based on the [TIM2_EE] bit of the Protocol Event Output Enable Register (PEOER).

56.6.18 Slot Status Monitoring

The CC provides several means for slot status monitoring. All slot status monitors use the same slot status vector provided by the PE. The PE provides a slot status vector for each static slot, for each dynamic slot, for the symbol window, and for the NIT, on a per channel base. The content of the slot status vector is described in the "Slot Status Content" table below. The PE provides the slot status vector within the first macrotick after the end of the related slot/window/NIT, as shown in the below figure.

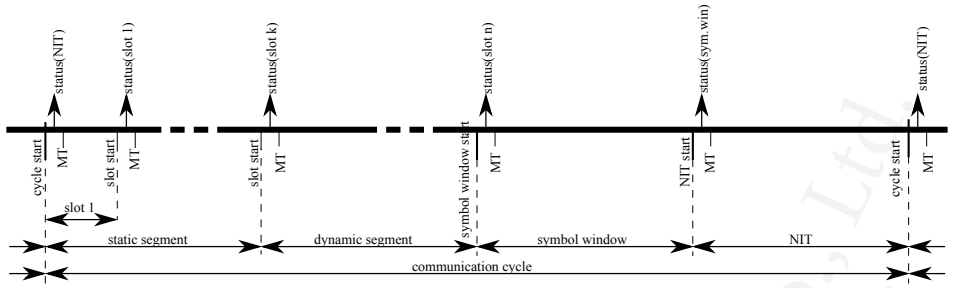


Figure 502. Slot Status Vector Update

NOTE

The slot status for the NIT of cycle n is provided after the start of cycle n+1.

Table 506. Slot Status Content

	Status Content
static /dynamic Slot	**slot related status** *vSS!ValidFrame* - valid frame received *vSS!SyntaxError* - syntax error occurred while receiving *vSS!ContentError* - content error occurred while receiving *vSS!BViolation* - boundary violation while receiving *for slots in which the module transmits:* *vSS!TxConflict* - reception ongoing while transmission starts *for slots in which the module does not transmit:* *vSS!TxConflict* - reception ongoing while transmission starts first valid - channel that has received the first valid frame **received frame related status** *extracted from* *a)header of valid frame, if vSS!ValidFrame = 1* *b) last received header, if vSS!ValidFrame = 0* *c) set to 0, if nothing was received* *vRF!Header!NFIndicator* - Null Frame Indicator (0 for null frame) *vRF!Header!SuFIndicator* - Startup Frame Indicator *vRF!Header!SyFIndicator* - Sync Frame Indicator
Symbol Window	**window related status** *vSS!ValidFrame* - always 0 *vSS!ContentError* - content error occurred while receiving *vSS!SyntaxError* - syntax error occurred while receiving *vSS!BViolation* - boundary violation while receiving

Table continues on the next page...

Table 506. Slot Status Content (continued)

	Status Content
	vSS!TxConflict - reception ongoing while transmission starts **received symbol related status** *vSS!ValidMTS* - valid Media Test Access Symbol received **received frame related status** *see static/dynamic slot*
NIT	**NIT related status** *vSS!ValidFrame* - always 0 *vSS!ContentError* - content error occurred while receiving *vSS!SyntaxError* - syntax error occurred while receiving *vSS!BViolation* - boundary violation while receiving *vSS!TxConflict* - always 0 **received frame related status** *see static/dynamic slot*

56.6.18.1 Channel Status Error Counter Registers

The two channel status error counter registers, Channel A Status Error Counter Register (CASERCR) and Channel B Status Error Counter Register (CBSERCR), incremented by one, if at least one of four slot status error bits, *vSS!SyntaxError, vSS!ContentError, vSS!BViolation*, or *vSS!TxConflict* is set to 1. The status vectors for all slots in the static and dynamic segment, in the symbol window, and in the NIT are taken into account. The counters wrap round after they have reached the maximum value.

56.6.18.2 Protocol Status Registers

The Protocol Status Register 2 (PSR2) provides slot status information about the Network Idle Time NIT and the Symbol Window. The Protocol Status Register 3 (PSR3) provides aggregated slot status information.

56.6.18.3 Slot Status Registers

The eight slot status registers, Slot Status Registers (SSR0–SSR7), can be used to observe the status of static slots, dynamic slots, the symbol window, or the NIT without individual message buffers. These registers provide all slot status related and received frame / symbol related status information, as given in Table 506, except of the *first valid* indicator for non-transmission slots.

56.6.18.4 Slot Status Counter Registers

The CC provides four slot status error counter registers, Slot Status Counter Registers (SSCR0–SSCR3). Each of these slot status counter registers is updated with the value of an internal slot status counter at the start of a communication cycle. The internal slot status counter is incremented if its increment condition, defined by the Slot Status Counter Condition Register (SSCCR), matches the status vector provided by the PE. All static slots, the symbol window, and the NIT status are taken into account. *Dynamic* slots are *excluded*. The internal slot status counting and update timing is shown in the figure below.

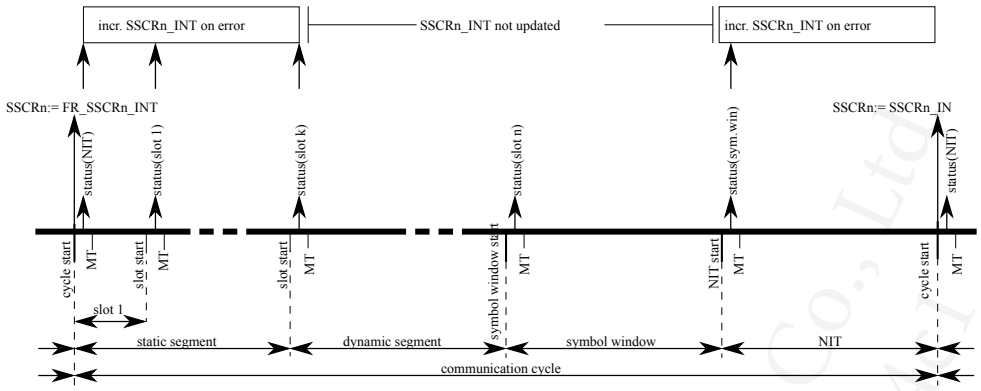


Figure 503. Slot Status Counting and SSCRn Update

The PE provides the status of the NIT in the first slot of the next cycle. Due to these facts, the SSCRn register reflects, in cycle n, the status of the NIT of cycle n-2, and the status of all static slots and the symbol window of cycle n-1.

The increment condition for each slot status counter consists of two parts, the frame related condition part and the slot related condition part. The internal slot status counter SSCRn_INT is incremented if at least one of the conditions is fulfilled:

1. frame related condition:

   (SSCCRn[VFR] | SSCCRn[SYF] | SSCCRn[NUF] | SSCCRn[SUF]) // count on frame condition

   = 1;

   and

   ((~SSCCRn[VFR] | vSS!ValidFrame) & // valid frame restriction

   (~SSCCRn[SYF] | vRF!Header!SyFIndicator) & // sync frame indicator restriction

   (~SSCCRn[NUF] | ~vRF!Header!NFIndicator) & // null frame indicator restriction

   (~SSCCRn[SUF] | vRF!Header!SuFIndicator)) // startup frame indicator restriction

   = 1;

   NOTE

   The indicator bits SYF, NUF, and SUF are valid only when a valid frame was received. Thus it is required to set the VFR always, whenever count on frame condition is used.

   slot related condition:

   ((SSCCRn[STATUSMASK[3]] & vSS!SyntaxError) | // increment on syntax error

   (SSCCRn[STATUSMASK[2]] & vSS!ContentError) | // increment on content error

   (SSCCRn[STATUSMASK[1]] & vSS!BViolation) | // increment on boundary violation

   (SSCCRn[STATUSMASK[0]] & vSS!TxConflict)) // increment on transmission conflict

   = 1;

If the slot status counter is in single cycle mode, i.e. SSCCRn[MCY] = 0, the internal slot status counter SSCRn_INT is reset at each cycle start. If the slot status counter is in the multicycle mode, i.e. SSCCRn[MCY] = 1, the counter is not reset and incremented, until the maximum value is reached.

56.6.18.5 Message Buffer Slot Status Field

Each individual message buffer and each FIFO message buffer provides a slot status field, which provides the information shown in Table 506 for the static/dynamic slot. The update conditions for the slot status field depend on the message buffer type. Refer to the Message Buffer Update Sections in Individual MB configuration data.

56.6.19 System Bus Access

This section provides a description of the system bus accesses failures and the related CC behavior. System bus access failures may occur when the CC transfers data to or from the FlexRay memory area.

The system bus access failure types are described in System Bus Access Failure Types.

The behavior of the CC after the occurrence of a system bus access failure is described in System Bus Access Failure Response.

56.6.19.1 System Bus Access Failure Types

This section describes the two types of system bus access failures.

56.6.19.1.1 System Bus Illegal Address Access

A system bus illegal address access is detected when the CC has used an illegal or invalid address to access the FlexRay system memory area. There are three conditions which are treated as a system bus illegal address access:

- The system bus subsystem detects an CC access to an illegal system memory address.
- The CC detects the usage of an data field offset with the value of 0.
- The CC detects a memory error while reading a data field offset from the CHI LRAM memory (see CHI LRAM Error Response after CC Read).

If a system bus illegal address access is detected, the CC sets the ILSA_EF flag in the CHI Error Flag Register (CHIERFR).

56.6.19.1.2 System Bus Access Timeout

A system bus access timout is detected if an access to the FlexRay memory area is not finished in time. The timeout value is derived from the SYMATOR[TIMEOUT] setting (see Configure System Memory Access Time-Out Register (SYMATOR)).

If a system bus access timout is detected, the CC sets the SBCF_EF flag in the CHI Error Flag Register (CHIERFR).

56.6.19.2 System Bus Access Failure Response

This section describes the two types of behavior of the CC after the occurrence of a system bus access failure. The actual behavior is defined by the SBFF bit in the Module Configuration Register (MCR).

56.6.19.2.1 Continue after System Bus Access Failure

If the SBFF bit in the Module Configuration Register (MCR) is 0, the CC will continue its operation after the occurrence of the system bus access failure, but will not generate any system bus accesses until the start of the next communication cycle.Since no data are read from or written to the FlexRay memory area, no messages are received or transmitted. Consequently, none of the individual message buffers or receive FIFOs will be updated until the next communication cycle starts.

If a frame is under transmission when the system bus failure occurs, a correct frame is generated with the remaining header and frame data are replaced by all zeros. Depending on the point in time this can affect the PPI bit, the Header CRC, the Payload Length in case of an dynamic slot, and the payload data. Starting from the next slot in the current cycle, no frames will be transmitted and received, except for the key slot, where a sync or startup null-frame is transmitted, if the key slot is assigned.

If a frame is received when the system bus failure occurs, the reception is aborted and the related receive message buffer is not updated.

Normal operation is resumed after the start of next communication cycle.

56.6.19.2.2 Freeze after System Bus Access Failure

If the SBFF bit in the Module Configuration Register (MCR) is set to 1, the CC will go into the freeze mode immediately after the occurrence of one of the system bus access failures.

56.6.20 Interrupt Support

The CC provides 300 individual interrupt sources and five combined interrupt sources.

56.6.20.1 Individual Interrupt Sources

Message buffer interrupts are discussed in this section.

56.6.20.1.1 Message Buffer Interrupts

The CC provides 256 message buffer interrupt sources.

Each individual message buffer provides an interrupt flag MBCCSRn[MBIF] and an interrupt enable bit MBCCSRn[MBIE]. The CC sets the interrupt flag when the slot status of the message buffer was updated. If the interrupt enable bit is asserted, an interrupt request is generated.

56.6.20.1.2 FIFO Interrupts

The CC provides 2 FIFO interrupt sources.

Each of the 2 FIFO provides a Receive FIFO Almost Full Interrupt Flag. The CC sets the Receive FIFO Almost Full Interrupt Flags (GIFER[FAFBIF], GIFER[FAFAIF]) in the Global Interrupt Flag and Enable Register (GIFER) if the corresponding Receive FIFO fill level exceeds the defined watermark.

56.6.20.1.3 Wakeup Interrupt

The CC provides one interrupt source related to the wakeup.

The CC sets the Wakeup Interrupt Flag GIFER[WUPIF] when it has received a wakeup symbol on the FlexRay bus. The CC generates an interrupt request if the interrupt enable bit GIFER[WUPIE] is asserted.

56.6.20.1.4 Protocol Interrupts

The CC provides 25 interrupt sources for protocol related events. For details, see Protocol Interrupt Flag Register 0 (PIFR0) and Protocol Interrupt Flag Register 1 (PIFR1). Each interrupt source has its own interrupt enable bit.

56.6.20.1.5 CHI Interrupts

The CC provides 16 interrupt sources for CHI related error events. For details, see CHI Error Flag Register (CHIERFR). There is one common interrupt enable bit GIFER[CHIE] for all CHI error interrupt sources.

56.6.20.2 Combined Interrupt Sources

Each combined interrupt source generates an interrupt request only when at least one of the interrupt sources that is combined generates an interrupt request.

56.6.20.2.1 Receive Message Buffer Interrupt

The Receive Message Buffer Interrupt request is generated when at least one of the individual receive message buffers generates an interrupt request MBXIRQ[n] and the interrupt enable bit GIFER[RBIE] is set.

56.6.20.2.2 Transmit Message Buffer Interrupt

The Transmit Message Buffer Interrupt request is generated when at least one of the individual transmit message buffers generates an interrupt request MBXIRQ[n] and the interrupt enable bit GIFER[TBIE] is asserted.

56.6.20.2.3 Protocol Interrupt

The Protocol Interrupt request is generated when at least one of the individual protocol interrupt sources generates an interrupt request and the interrupt enable bit GIFER[PRIE] is set.

56.6.20.2.4 CHI Interrupt

The CHI Interrupt request is generated when at least one of the individual chi error interrupt sources generates an interrupt request and the interrupt enable bit GIFER[CHIE] is set.

56.6.20.2.5 Module Interrupt

The Module Interrupt request is generated if at least one of the combined interrupt sources generates an interrupt request and the interrupt enable bit GIFER[MIE] is set.

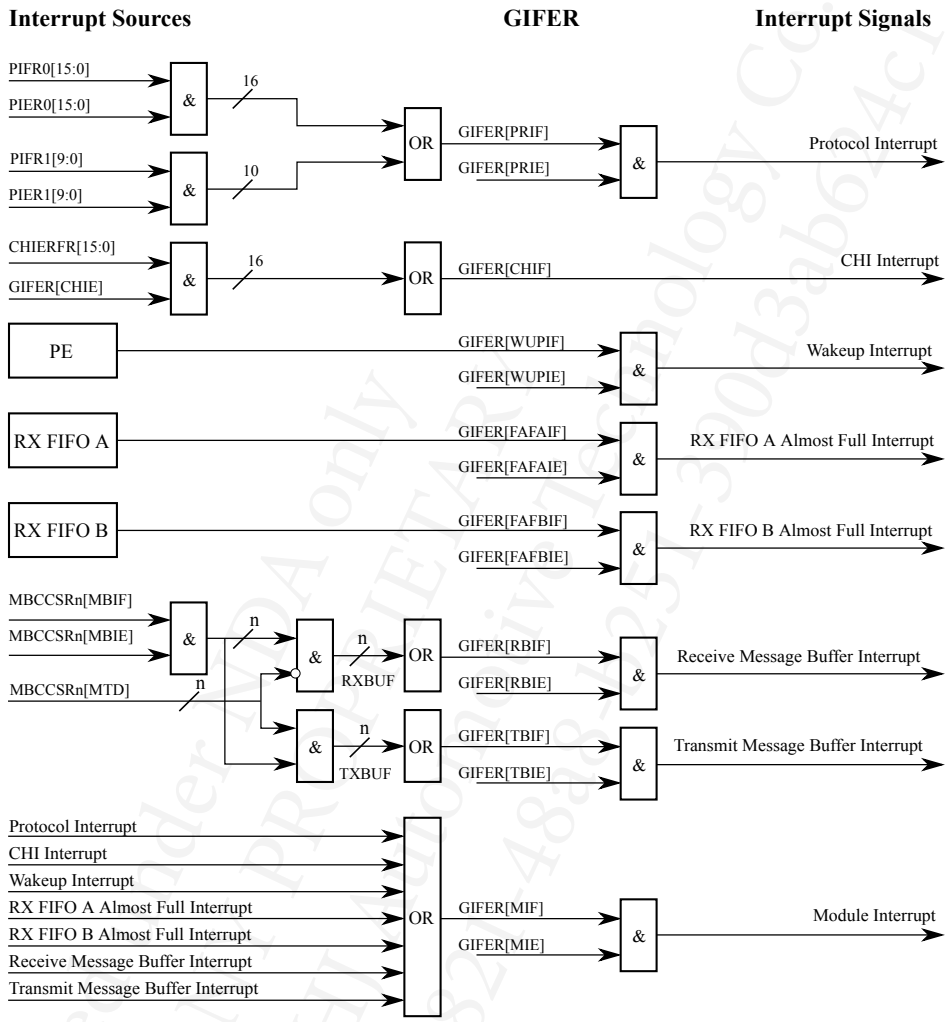


Figure 504. Scheme of GIFER interrupt signal generation

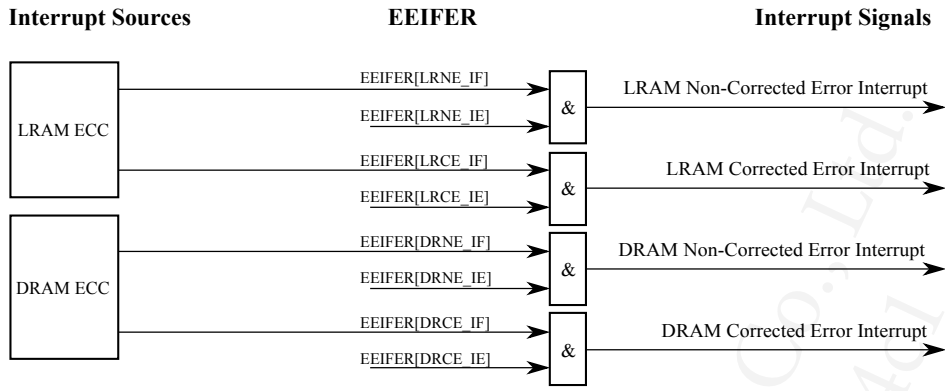


Figure 505. Scheme of EEIFER interrupt signal generation

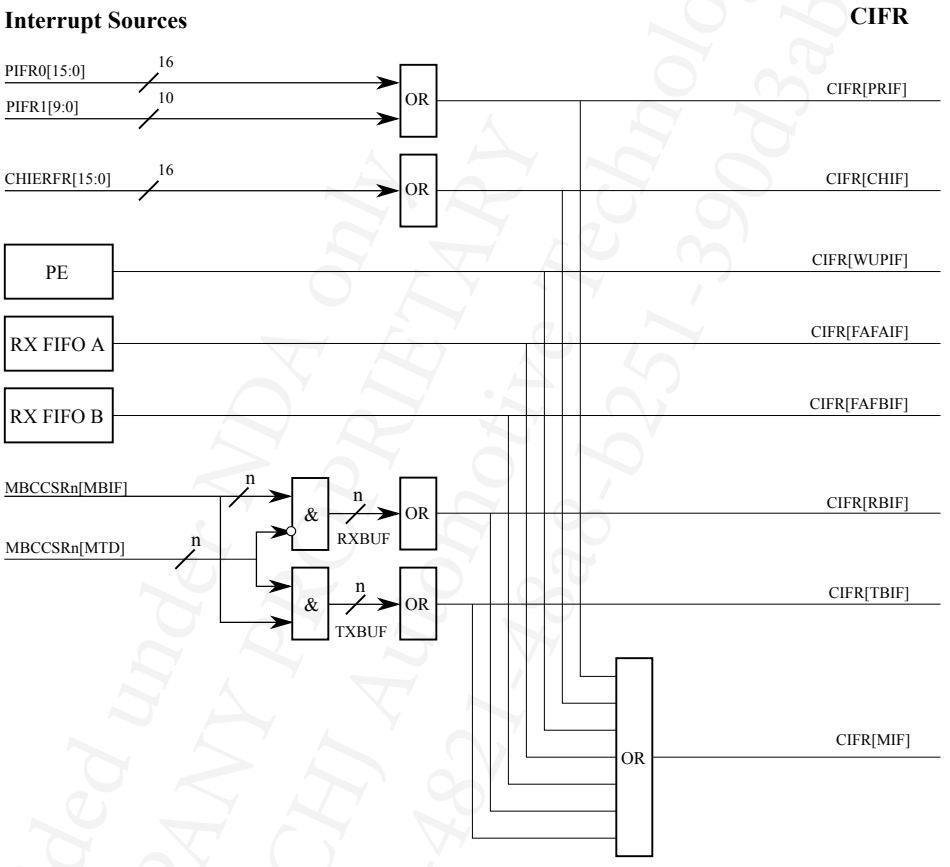


Figure 506. Scheme of CIFR flags generation

56.6.21 Lower Bit Rate Support

The CC supports a number of lower bit rates on the FlexRay bus channels. The lower bit rates are implemented by modifying the duration of the microtick *pdMicrotick*, the number of samples per microtick *pSamplesPerMicrotick*, the number of samples per bit *cSamplesPerBit*, and the strobe offset *cStrobeOffset*. The application configures the FlexRay channel bit rate by setting the BITRATE field in the Module Configuration Register (MCR). The protocol values are set internally. The available bit rates, the related BITRATE field configuration settings and related protocol parameter values are shown in the "FlexRay Channel Bit Rate Control" table below.

Table 507. FlexRay Channel Bit Rate Control

FlexRay Channel Bit Rate [Mbit/s]	MCR[BITRATE]	*pdMicrotick [ns]*	*gdSampleClockPeriod [ns]*	*pSamplesPerMicrotick*	*cSamplesPerBit*	*cStrobeOffset*
10.0	000	25.0	12.5	2	8	5
8.0	011	25.0	12.5	2	10	6
5.0	001	25.0	25.0	1	8	5
2.5	010	50.0	50.0	1	8	5

NOTE

The bit rate of 8 Mbit/s is not defined by the *FlexRay Communications System Protocol Specification, Version 2.1 Rev A*.

56.6.22 PE Data Memory (PE DRAM)

The PE Data Memory (PE DRAM) is 128 word, 16-bit wide memory with byte access, which contains the program data of the PE internal CPU. The PE DRAM is divided into two banks, 8-bit each. The memory data [7:0] are assigned to BANK0, the memory data [15:8] are assigned to BANK1.

Table 508. PE DRAM Layout

ADDR	BANK1	BANK0
0x00	byte1	byte0
0x01	byte3	byte2
	…	
0x7F	byte255	byte254

The FlexRay module provides means to access the PE DRAM from the application. The PE DRAM application access is initiated and controlled via PE DRAM Access Register (PEDRAR) and PE DRAM Access Register (PEDRAR). This functionality is used to check the memory error detection.

56.6.22.1 PE DRAM Read Access

A read access from the PE DRAM can be initiated in any protocol state. The following sequence describes a read access from the PE DRAM address 0x70.

1. PEDRAR:= 0x50E0;

   // INST=0x5; ADDR=070

2. wait until PEDRAR[DAD] == 1;

   // wait for end of PE DRAM access

3. val = PEDRDR[DATA];

   // read PE DRAM data

The read access is handled by the PE internal CPU with the lowest execution priority. This may cause an response delay with a maximum of 1000 PE clock cycle (25us).

56.6.22.2 PE DRAM Write Access

A write access into the PE DRAM can be initiated in any protocol state. The following sequence describes a write access to the PE DRAM address 0x70.

1. PEDRDR:= DATA;

   // write value to be written into data register

2. PEDRAR:= 0x30E0;

   // INST=0x3; ADDR=0x70

3. wait until PEDRAR[DAD] == 1;

   // wait for end of PE DRAM access

4. val = PEDRDR[DATA];

   // read back PE DRAM data

The write access is handled by the PE internal CPU with the lowest execution priority. This may causes an response delay with a maximum of 1000 PE clock cycle (25us).

If the conditions given in PE DRAM Write Access Limitations are fulfilled, the data provided in PE DRAM Access Regsiter (PEDRAR) are written into the PE DRAM, read back in the next clock cycle and stored into the PE DRAM Access Register (PEDRAR). Otherwise, data are not written into the PE DRAM and 0x0000 is stored into the PE DRAM Access Register (PEDRAR).

56.6.22.3 PE DRAM Write Access Limitations

The PE DRAM is used by the protocol engine if the module is not in *POC:default config* state. The only address not used by the protocol engine is 0x70. To prevent the corruption of protocol engine data the following PE DRAM write access limitations apply for application writes.

1. When the module is in *POC:default config* state, all PE DRAM addresses are writable.
2. When the module is not *in POC:default config* state, only PE DRAM address 0x70 is writable.

56.6.23 CHI Lookup-Table Memory (CHI LRAM)

The CHI Lookup-Table Memory (CHI LRAM) is an CHI internal memory which contains the message buffer configuration data and the data field offsets for the physical message buffers. The configuration data for two message buffers or 6 data field offsets are contained in one memory row. The CHI LRAM is divided into 6 memory BANKs.

Table 509. CHI LRAM Layout

ADR	BANK5	BANK4	BANK3	BANK2	BANK1	BANK0
0x00	MBIDXR1	MBFIDR1	MBCCFR1	MBIDXR0	MBFIDR0	MBCCFR0
0x01	MBIDXR3	MBFIDR3	MBCCFR3	MBIDXR2	MBFIDR2	MBCCFR2
…						

Table continues on the next page...

Table 509. CHI LRAM Layout (continued)

ADR	BANK5	BANK4	BANK3	BANK2	BANK1	BANK0
0x7F	MBIDXR255	MBFIDR255	MBCCFR255	MBIDXR254	MBFIDR254	MBCCFR254
0x80	MBDOR5	MBDOR4	MBDOR3	MBDOR2	MBDOR1	MBDOR0
…						
0xAB	-	-	-	-	MBDOR259	MBDOR258
0xAC	LEETR5	LEETR4	LEETR3	LEETR2	LEETR1	LEETR0

56.6.23.1 CHI LRAM Read and Write Access

The CHI LRAM is accessed by the application via regular register read and write accesses.

56.6.24 Memory Content Fault Detection

The FlexRay module provides integrated memory content error detection for both the CHI LRAM and PE DRAM, and memory content error correction for the PE DRAM. The memory error detection for the CHI LRAM uses an standard Hamming code with a Hamming distance of 3 and detects all single-bit and double-bit errors (SEDDED) .The memory error detection and correction for the PE DRAM uses an enhanced Hamming code with a Hamming distance of 4 and detects and corrects all single-bit errors and detects all double-bit errors (SECDED).

This section describes the reporting of the occurrence of memory content errors, the reaction of the module on the occurrence, and how the application can inject memory errors in order to trigger the report and response behavior.

56.6.24.1 Memory Error Types

A memory error is the distortion of one or more bits read out of the memory. The reading of the values of all zeros and all ones is considered as an special case. The FlexRay module detects and indicates the memory errors as shown in the "Detected Memory Error Types" table below. The entries on the top have higher priority.

Each memory read access reads out *all* banks of the addressed row, and runs error detection on *all* banks, even in the case that the application has triggered a read from only one bank. This may lead to the reporting of an memory error if at least one bank contains a memory error, even if an error free bank has been read.

Table 510. Detected Memory Error Types

Memory	Priority	Memory Data	Indication
CHI LRAM	0 (highest)	All Zero's	Non-Corrected Error
PE DRAM			Non-Corrected Error
CHI LRAM		All One's	Non-Corrected Error
PE DRAM			
CHI LRAM	1 (lowest)	One Bit Flipped	Non-Corrected Error

Table continues on the next page...

Table 510. Detected Memory Error Types (continued)

Memory	Priority	Memory Data	Indication
PE DRAM			Corrected Error
CHI LRAM		Two Bits Flipped	Non-Corrected Error
PE DRAM			
CHI LRAM		Three or more Bits Flipped	one out of {No error, Non-Corrected Error}, defined by coding given in CHI LRAM checkbits and CHI LRAM Syndrome
PE DRAM			one out of {No error, Corrected Error, Non-Corrected Error}, defined by coding given in PE DRAM Checkbits and PE DRAM Syndrome

56.6.24.2 Memory Error Reporting

The memory error reporting is enabled only if the ECC functionality enable bit ECCE in the Module Configuration Register (MCR) is set.

For each of the two memories exists two sets of internal registers to store the detection of one corrected and one non-corrected memory error.

If a memory error is detected, the module checks whether the related error interrupt flag in the ECC Error Interrupt Flag and Enable Register (EEIFER) is set.

- *If the error interrupt flag is set, the related internal error reporting register is not updated and the related error overflow flag is set to 1 to indicate a loss of error condition.*
- *If the error interrupt flag is not set, the internal reporting register is updated and the error interrupt flag is set to 1. If two or more memory errors of the same type are detected, the error for the bank with the lower bank number will be reported, and the error overflow flag will be set to 1.*

If a memory error is detected for at least two banks of one memory, the related error overflow flag is set to 1 to indicate a loss of error condition.

In addition to above the Error indications (corrected / non corrected) along with the failing address is sent out from the module for external error logging and corrective actions by the Software.

56.6.24.2.1 PE DRAM Checkbits

The coding of the checkbits reported in ECC Error Report Code Register (EERCR) for PE DRAM memory errors is shown in Table 510. This table shows the implemented enhanced Hamming code. If the error injection was applied to distort the checkbits, then the distorted checkbits are reported.

Table 511. PE DRAM checkbits coding

CODE	CODE				DATA							
	3	2	1	0	7	6	5	4	3	2	1	0
4[1]	X	X	X	X	X	X	X	X	X	X	X	X
3[2]	-	-	-	-	X	X	X	X	-	-	-	-

Table continues on the next page...

Table 511. PE DRAM checkbits coding (continued)

CODE	CODE				DATA							
	3	2	1	0	7	6	5	4	3	2	1	0
2	-	-	-	-	X	-	-	-	X	X	X	-
1	-	-	-	-	-	X	X	-	X	X	-	X
0	-	-	-	-	-	X	-	X	X	-	X	X

1. The checkbit CODE[4] is set to 1 if and only if there is a even number of 1's in columns with X.
2. The checkbits CODE[3]… CODE[0] are set to 1 if and only if there is a odd number of 1's in all columns with X.

This coding of the checkbit ensures that neither 0x000 nor 0xFFF are valid code words and written into the memory.

56.6.24.2.2 PE DRAM Syndrome

The coding of the syndrome reported in ECC Error Report Code Register (EERCR) for PE DRAM memory errors is shown in the following table.

Table 512. EERCR[CODE] PE DRAM Syndrome Coding

EERCR[CODE]		Description
[4]	[3:0]	
0x1	0x0	No Error (Never appears in error report registers)
0x0	0x0	If data == 0: Non Corrected Error (Dedicated Handling of All Zero Code Word) If data!= 0: Corrected Error (Parity Bit 4)
0x0	0x1	Corrected Error (Parity Bit 0)
0x0	0x2	Corrected Error (Parity Bit 1)
0x0	0x3	Corrected Error (Data Bit 0)
0x0	0x4	Corrected Error (Parity Bit 2)
0x0	0x5	Corrected Error (Data Bit 1)
0x0	0x6	Corrected Error (Data Bit 2)
0x0	0x7	Corrected Error (Data Bit 3)
0x0	0x8	Corrected Error (Parity Bit 3)
0x0	0x9	Corrected Error (Data Bit 4)
0x0	0xA	Corrected Error (Data Bit 5)
0x0	0xB	Corrected Error (Data Bit 6)

Table continues on the next page...

Table 512. EERCR[CODE] PE DRAM Syndrome Coding (continued)

EERCR[CODE]		Description
[4]	[3:0]	
0x0	0xC	Corrected Error (Data Bit 7)
0x0	0xD-0xF	Non-Corrected Error
0x1	0x1-0xF	Non-Corrected Error

56.6.24.2.3 CHI LRAM checkbits

The coding of the checkbits reported in ECC Error Report Code Register (EERCR) for CHI LRAM memory errors is shown in the table below. This table shows the implemented Hamming code. If the error injection is applied to distort the checkbits, then the distorted checkbits are reported.

Table 513. CHI LRAM checkbits coding

CODE[1]	DATA															
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
4	X	X	X	X	X	-	-	-	-	-	-	-	-	-	-	-
3	-	-	-	-	-	X	X	X	X	X	X	X	-	-	-	-
2	X	X	-	-	-	X	X	X	X	-	-	-	X	X	X	-
1	-	-	X	X	-	X	X	-	-	X	X	-	X	X	-	X
0	X	-	X	-	X	X	-	X	-	X	-	X	X	-	X	X

1. The checkbit CODE[n] is set to 1 if and only if there is a odd number of 1's in all columns with X.

56.6.24.2.4 CHI LRAM Syndrome

The coding of the syndrome reported in ECC Error Report Code Register (EERCR) for CHI LRAM memory errors is shown in the next table.

Table 514. EERCR[CODE] CHI LRAM Syndrome Coding

EERCR[CODE]	Description
0x00	No Error (Never appears in error report registers)
0x01-0x1F	Non Corrected Error

56.6.24.3 Memory Error Response

The memory error response is enabled only when the ECC functionality enable bit ECCE in the Module Configuration Register (MCR) is set.

In case of the detection of a *corrected* memory error, the FlexRay module continues its normal operation using the corrected data word. This section describes the behavior of the FlexRay module after the detection of a *non-corrected* memory error.

56.6.24.3.1 CHI LRAM Error Response after CC Read

When the CC is out of the *POC:default config* state, it reads the configuration data and the data field offsets of all utilized message buffers in every slot and in the NIT. If a non-corrected memory error is detected during this module read access the error response of the module depends from LRAM location where the error occurred.

- *If the LRAM address belongs to physical message buffer configuration data the FlexRay module will consider the affected message buffer as disabled for the current search and will exclude this buffer from the search. The configuration of the affected message buffer is not changed.*

  If the affected message buffer is a tx message buffer, no frame will be transmitted from this message buffer in the next slot. If the affected message buffer is a rx message buffer, no frame will be received to this message buffer in the next slot.
- *If the LRAM address belongs to the data field offset area and the related physical message buffer is used for Rx or Tx the first access to the system memory caused by payload read or write yields to the assertion of the CHIERFR[ILSA_EF]. No memory access occurs w.r.t. payload access is performed for the complete frame.*

56.6.24.3.2 CHI LRAM Error Response after Application Read

The application can read the content of the CHI LRAM via reading the MBCCFRn, MBFIDRn, MBIDXRn, MBDORn, and LEETRn registers. If a non-corrected memory error is detected during this kind of read access, the module indicates the detected memory error, delivers the non-corrected data read and continues its normal operation.

56.6.24.3.3 PE DRAM Error Response after CC Read

If the CC detects an non-corrected memory error during internal read of program data which is contained in PE DRAM, this is considered as an fatal protocol error and the module enters the protocol freeze state immediately.

56.6.24.3.4 PE DRAM Error Response after Application Read in *POC:default config* state

If the CC detects an non-corrected memory error during an application triggered read from any PE DRAM address and the protocol is in the *POC:default config* state, this is considered as an fatal protocol error and the module enters the protocol freeze state. This behavior allows for checking the freeze functionality in case of the detection of non-corrected errors.

56.6.24.3.5 PE DRAM Error Response after Application Read out of *POC:default config*

If the CC detects an non-corrected memory error during an application triggered read from any PE DRAM address, and the protocol is not in the *POC:default config* state, this error is not considered as an fatal error and the protocol state is not changed. This prevents any interference of the running protocol by PE DRAM error injection reads.

56.6.25 Memory Fault Injection

The error injection functionality is used by the application to inject data errors into the memories to trigger and check the memory error detection functionality.

The error injection is enabled only if the ECC functionality enable bit ECCE in the Module Configuration Register (MCR) and the error injection enable control bit EIE in the ECC Error Report and Injection Control Register (EERICR) are set.

The error injection mode is configured by the EIM configuration bit in the ECC Error Report and Injection Control Register (EERICR).

When the error injection is enabled, each write access to the configured memory location will be distorted.

The injector has the same behavior for FlexRay module memory writes and application memory writes.

56.6.25.1 CHI LRAM Error Injection

The following sequence describes an memory error injection sequence for the CHI LRAM memory. This sequence consists of the setup of the error injector followed by an application triggered write access to provoke an distortion of the memory content. The content of the CHI LRAM is described in Table 509.

When the CC is in *POC:default config*, there are no limitations for the error injection and no impacts of error injection to the application. For error injection out of *POC:default config* see Memory Fault Injection out of POC:default config.

Injector Setup:

1. MCR[ECCE]:= 1;

   // enable ECC functionality
2. EERICR[EIE]:=I_MODE;

   // configure error injection mode
3. EEIAR[MID]:= 1;

   // select CHI LRAM for error injection
4. EEIAR[BANK]:= I_BANK;

   // select bank for error injection; I_BANK = {0,1,2,3,4,5}
5. EEIAR[ADDR]:= I_ADDR;

   // select address for error injection; I_ADDR <= 0xAC
6. EEIDR[DATA]:= D_DIST;

   // define data distortion pattern
7. EEICR[CODE]:= C_DIST;

   // define checkbit distortion pattern
8. EERICR[EIE]:=1;

   // enable error injection

Application Write Access:

If (I_BANK==0) -> MBCCFR(2n) / MBDOR(6k) / LEETR0 := DATA;

If (I_BANK==1) -> MBFIDR(2n) / MBDOR(6k+1) / LEETR1 := DATA;

If (I_BANK==2) -> MBIDXR(2n) / MBDOR(6k+2) / LEETR2 := DATA;

If (I_BANK==3) -> MBCCFR(2n+1) / MBDOR(6k+3) / LEETR3 := DATA;

If (I_BANK==4) -> MBFIDR(2n+1) / MBDOR(6k+4) / LEETR4 := DATA;

If (I_BANK==5) -> MBIDXR(2n+1) / MBDOR(6k+5) / LEETR5 := DATA;

// write DATA to the defined injection bank and injection address (see Table 509).

56.6.25.2 PE DRAM Error Injection

The following sequence describes an memory error injection sequence for the PE DRAM memory. This sequence consists of the setup of the error injector followed by an application triggered write access to provoke an distortion of the memory content.

When the FlexRay module is in *POC:default config*, there are no limitations for the error injection and no impacts of error injection to the application. For error injection out of *POC:default config* see PE DRAM Error Injection out of POC:default config.

Injector Setup:

1. MCR[ECCE]:= 1;

   // enable ECC functionality
2. EERICR[EIE]:=I_MODE;

   // configure error injection mode
3. EEIAR[MID]:= 0;

// select PE DRAM for error injection

4. EEIAR[BANK]:= I_BANK;

   // define bank for error injection; I_BANK = {0,1}

5. EEIAR[ADDR]:= I_ADDR;

   // define address for error injection; I_ADDR <= 0x7F

6. EEIDR[DATA]:= D_DIST;

   // define data distortion pattern

7. EEICR[CODE]:= C_DIST;

   // define checkbit distortion pattern

8. EERICR[EIE]:=1;

   // enable error injection

Application Write Access (e.g. I_ADDR=0x70):

1. PEDRAR:= 0x30E0;

   // INST=0x3; ADDR=0x70

2. wait until PEDRAR[DAD] == 1;

   // wait for end of PE DRAM access

3. val = PEDRDR[DATA]; |

   // get read back PE DRAM data

NOTE

The write access to the PE DRAM triggers an subsequent read access from PE DRAM in the next cycle, which triggers the detection of the distorted data.

56.6.26 StopWatch function

FlexRay supports a StopWatch function to facilitate OS synchronization to the FlexRay network. This provides a capability to capture the time difference in system clocks between the operating system timer tick and the FlexRay time base. This allows the software synchronization handler to adjust the OS clock to sync with the FR macrotick and are fulfilled.

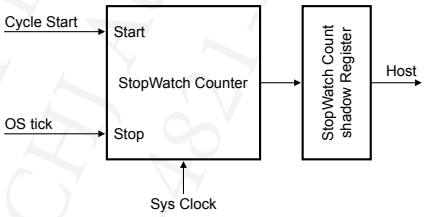


Figure 507. StopWatch counter logic

As shown below, there is a 32 bit stopwatch counter that resets and starts with each cycle start. This counter runs on the system clock and increments every cycle until an OS tick event is received, after which the counter halts and the count value is transferred to a shadow register. The shadow register (Stop Watch Count Register (STPWR)) holds the counter value till the host reads all the 32 bits, after which it automatically gets cleared. The StopWatch logic is controlled through a control bit STPW_EN in the PEOER (Protocol Event Output Enable and StopWatch Control Register (PEOER). The counter runs only when this bit is set. The following figure shows the actual timing of the stopwatch counter logic operation.

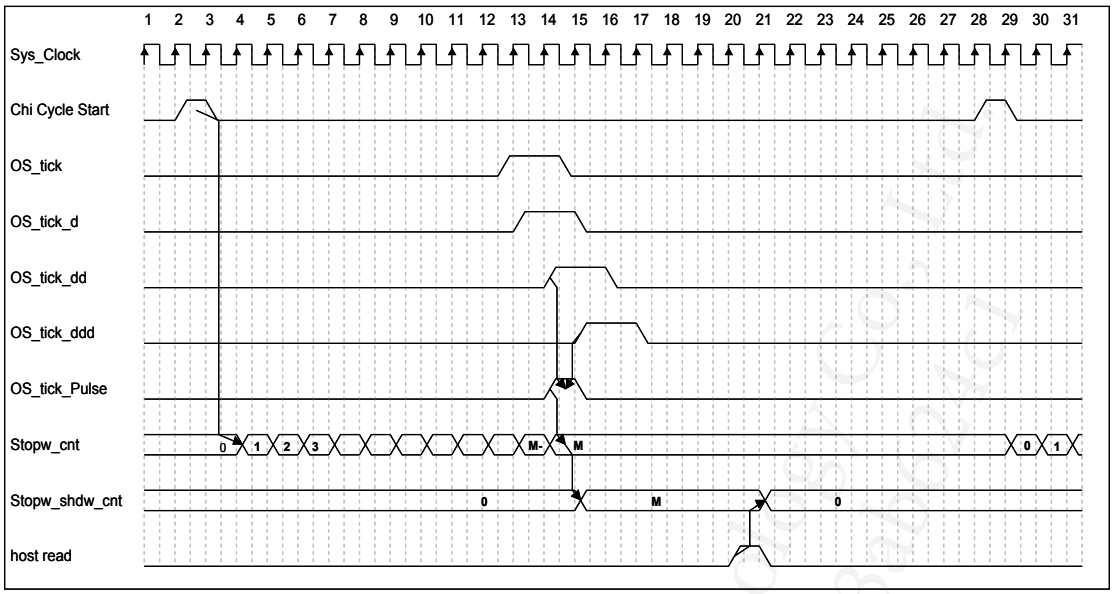


Figure 508. StopWatch function timing diagram

NOTE

'Host read' implies that all 32 bits have been read out completely. Any partial read (i.e for 8/16 bits access) will not clear the StopWatch count register.

NOTE

If cycle start and OS tick occurs simultaneously, timer will remain at 0. But OS_tick has a 2 stage synchronizer and cycle_start has a single stage synchronizer into chi clock domain. So simultaneously here means OS_tick_pulse and chi_cycle_start occurring at the same cycle.

56.7 Application Information

Module operation is discussed in this section.

56.7.1 Module Configuration

This section describes essential parts of the module configuration.

56.7.1.1 Configure System Memory Access Time-Out Register (SYMATOR)

To ensure reliable operation of the CC, the application must ensure that the TIMEOUT value in System Memory Access Time-Out Register (SYMATOR) and the CHI clock frequency f_{CHI} in MHz fulfill this equation[1]:

$$0 \le \mathrm{SYMATOR}[\mathrm{TIMEOUT}] \le \lfloor 0.45 \cdot f_{\mathrm{CHI}} - 8 \rfloor$$

For a given SYMATOR[TIMEOUT] value, f_{CHI} can be increased without causing unreliable operation of the CC. The same holds for reducing the SYMATOR[TIMEOUT] value for a given f_{CHI}.

Some examples for maximum values of the SYMATOR[TIMEOUT] for a minimum CHI frequency are given in the table below.

[1] See for all constraints of minimum CHI clock frequency.

Table 515. Maximum SYMATOR[TIMEOUT] examples

f_{CHI}	SYMATOR[TIMEOUT]	f_{CHI}	SYMATOR[TIMEOUT]
18 MHz	0	100 MHz	<= 37
23 MHz	<= 2	120 MHz	<= 46
27 MHz	<= 4	140 MHz	<= 55
32 MHz	<= 6	160 MHz	<= 64
60 MHz	<= 19	180 MHz	<= 73
80 MHz	<= 28	200 MHz	<= 82

56.7.1.1.1 System Bus Wait State Constraints

The SYMATOR[TIMEOUT] value corresponds directly to a certain acceptable number of wait states on the system bus.

For single channel configurations and if the sync frame table generation functionality is *not* used (SFTCCSR[SDVEN,SIDEN] = 0) no timeout will be detected if less than 2 × SYMATOR[TIMEOUT]+1 wait states will be seen on the system bus for each system bus access.

For dual channel configurations, or if the sync frame table generation functionality is used, no timeout will be detected if less than SYMATOR[TIMEOUT]-1 wait states will be seen on the system bus for each system bus access.

56.7.1.2 Configure Data Field Offsets

The data field offsets are located in the Message Buffer Data Field Offset Registers (MBDORn) and Receive FIFO Start Data Offset Register (RFSDOR). The application has to configure the data field offset values for all message buffers which are used.

When the module is enabled, the initilization value of the MBDORn[MBDO] and RFSDOR[SDO] is 0. This value is considered to be illegal (see System Bus Illegal Address Access).

56.7.2 Initialization Sequence

This section describes the required steps to initialize the CC. The first subsection describes the steps required after a module reset, the second section describes the steps required after preceding shutdown of the CC.

56.7.2.1 Module Initialization

This section describes the module related initialization steps after a system reset.

1. Configure CC.
    a. configure the control bits in the Module Configuration Register (MCR)
    b. configure the system memory base address in System Memory Base Address Register (SYMBADR)
2. Enable the CC.
    a. write 1 to the module enable bit MEN in the Module Configuration Register (MCR)

The CC now enters the Normal Mode. The application can commence with the protocol initialization described in Protocol Initialization.

56.7.2.2 Protocol Initialization

This section describes the protocol related initialization steps.

1. Configure the Protocol Engine.

a. issue CONFIG command via Protocol Operation Control Register (POCR)

b. wait for *POC:config* in Protocol Status Register 0 (PSR0)

c. configure the PCR0,…, PCR30 registers to set all protocol parameters

2. Configure the Message Buffers and FIFOs.

a. set the number of message buffers used and the message buffer segmentation in the Message Buffer Segment Size and Utilization Register (MBSSUTR)

b. define the message buffer data size in the Message Buffer Data Size Register (MBDSR)

c. configure each message buffer by setting the configuration values in the Message Buffer Configuration, Control, Status Registers (MBCCSRn), Message Buffer Cycle Counter Filter Registers (MBCCFRn), Message Buffer Frame ID Registers (MBFIDRn), Message Buffer Index Registers (MBIDXRn)

d. configure the FIFOs

e. issue CONFIG_COMPLETE command via Protocol Operation Control Register (POCR)

f. wait for *POC:ready* in Protocol Status Register 0 (PSR0)

After this sequence, the CC is configured as a FlexRay node and is ready to integrate into the FlexRay cluster.

56.7.2.3 CHI LRAM Initialization

The initialization of the CHI LRAM is performed by the CC when it leaves the Disabled Mode. The unitization runs for 173 CHI clock cycles. All fields in the MBCCSRn, MBCCFRn, MBFIDRn, MBDORn, and LEETRn registers are initialized to 0. All application read or write accesses to these registers are delayed until the initialization is finished.

56.7.2.4 PE DRAM Initialization

The PE DRAM initialization is performed by the CC in the *POC:default config* state. This initialization runs for 4.8 µs, and will delay the state transition from *POC:default config* into *POC:config*.

56.7.3 Memory Fault Injection out of POC:default config

This section provides information for application driven memory fault injection out if *POC:default config*. The CC provides means to inject memory faults from the application without any impacts to the internal protocol operation of the CC.

56.7.3.1 CHI LRAM Error Injection out of POC:default config

The CC will never perform any internal read access from the LRAM ECC Error Test Registers (LEETRn). Any memory errors injected into these CHI LRAM locations will never be detected by internal access, independent from the protocol state.

The application should use these registers and related CHI LRAM location to inject memory errors into the CHI LRAM. The injection sequence is described in CHI LRAM Error Injection.

56.7.3.2 PE DRAM Error Injection out of POC:default config

The CC will never perform any internal read access from the PE DRAM address 0x70. This is the only one PE DRAM address writable by the application out of the *POC:default config* state.

The application should use these PE DRAM location to inject memory errors into the PE DRAM. The injection sequence is described in PE DRAM Error Injection.

56.7.4 Shut Down Sequence

This section describes a secure shut down sequence to stop the CC gracefully. The main targets of this sequence are

- *finish all ongoing reception and transmission*
- *do not corrupt FlexRay bus and do not disturb ongoing FlexRay bus communication*

For a graceful shutdown the application shall perform the following tasks:

1. Disable all enabled message buffers.
    a. repeatedly write 1 to MBCCSRn[EDT] until MBCCSRn[EDS] == 0.
2. Stop Protocol Engine.
    a. issue HALT command via Protocol Operation Control Register (POCR)
    b. wait for *POC:halt* in Protocol Status Register 0 (PSR0)

56.7.5 Number of Usable Message Buffers

This section describes the required minimum CHI clock frequency for a specified number of utilized message buffers configured in the Message Buffer Segment Size and Utilization Register (MBSSUTR), a configured mini-slot length *gdMinislot*, and a configured nominal macrotick length *gdMacrotick*[1].

The CC uses a sequential search algorithm to determine the individual message buffer assigned or subscribed to the next slot. This search is started at the start of slot and must be finished before the start of the next slot.

The shortest FlexRay slot is a corrected empty dynamic slot. An corrected empty dynamic slot is a mini-slot and consists of gdMinislot corrected macroticks with a duration of *gdMacrotick*. The minimum duration of an corrected macrotick is $gdMacrotick_{min}$ = 39 µT. This results in a minimum length of an correct slot

$$\Delta_{slotmin} = 39 \bullet pdMicrotick \bullet gdMinislot$$

The message buffer search engine runs on the CHI clock and evaluates one individual message buffer per CHI clock cycle. For internal status update operations and to account for clock domain crossing jitter, an additional amount of 27 CHI clock cycles is required to ensure correct search engine operation.

For a given number of utilized message buffers MBSSUTR[LAST_MB_UTIL] + 1 and for a given CHI clock frequency f_{chi}, this results in a search duration of

$$\Delta_{search} = \frac{1}{f_{chi}} \bullet (MBSSUTR[LAST\_MB\_UTIL] + 27)$$

The message buffer search must be finished within one slot which requires fulfillment of this equation:

$$\Delta_{search} \leq \Delta_{slotmin}$$

This results in the formula given below which determines the required minimum CHI frequency for a given number of message buffers that are utilized.

$$f_{chi} \geq \frac{(MBSSUTR[LAST\_MB\_UTIL] + 27)}{39 \bullet pdMicrotick \bullet gdMinislot}$$

The required minimum CHI Clock frequency for a selected set of relevant protocol parameters and for the LAST_MB_UTIL field in the Message Buffer Segment Size and Utilization Register (MBSSUTR) set to 255 is given in the following table.

Table 516. Minimum fchi [MHz] examples (256 message buffers used)

pdMicrotick [ns]	*gdMinislot*					
	2	3	4	5	6	7
25.0	145.2	96.8	72.6	58.1	48.4	41.5
50.0	72.6	48.4	36.3	29.1	24.2	20.8

[1] See for all constraints of minimum CHI clock frequency.

NOTE

If the minimum CHI frequency is not met the CHIERFR[MBS_EF] flag is set. Refer to the CHI Error Flag Register (CHIERFR) for details.

56.7.6 Protocol Control Command Execution

This section considers the issues of the protocol control command execution.

The application issues any of the protocol control commands listed in the POCCMD field of the POCR Field Descriptions table, located in the Register Descriptions section, by writing the command to the POCCMD field of the Protocol Operation Control Register (POCR). As a result the CC sets the BSY bit while the command is transferred to the PE. When the PE has accepted the command, the BSY flag is cleared. All commands are accepted by the PE.

The PE maintains a protocol command vector. For each command that was accepted by the PE, the PE sets the corresponding command bit in the protocol command vector. If a command is issued while the corresponding command bit is set, the command is not queued and is lost.

If the command execution block of the PE is idle, it selects the next accepted protocol command with the highest priority from the current protocol command vector according to the protocol control command priorities given in the next table. If the current protocol state does not allow the execution of this protocol command (see POC state changes in *FlexRay Communications System Protocol Specification, Version 2.1 Rev A*) the CC asserts the illegal protocol command interrupt flag IPC_IF in the Protocol Interrupt Flag Register 1 (PIFR1). The protocol command is not executed in this case.

Some protocol commands may be interrupted by other commands or the detection of a fatal protocol error as indicated by the table below. If the application issues the FREEZE or READY command, or if the PE detects a fatal protocol error, some commands already stored in the command vector will be removed from this vector.

Table 517. Protocol Control Command Priorities

Protocol Command	Priority	Interrupted By	Cleared and Terminated By
FREEZE	(highest) 1	none	
READY	2		
CONFIG_COMPLETE	3		
ALL_SLOTS	4	FREEZE, READY, CONFIG_COMPLETE, fatal protocol error	FREEZE, READY, CONFIG_COMPLETE, fatal protocol error
ALLOW_COLDSTART	5		
RUN	6		FREEZE, fatal protocol error
WAKEUP	7		FREEZE, fatal protocol error
DEFAULT_CONFIG	8		FREEZE, fatal protocol error
CONFIG	9		
HALT	(lowest) 10		FREEZE, READY, CONFIG_COMPLETE, fatal protocol error

56.7.7 Message Buffer Search On Simple Message Buffer Configuration

This sections describes the message buffer search behavior for a simplified message buffer configuration. The FIFO behavior is not considered in this section.

56.7.7.1 Simple Message Buffer Configuration

A simple message buffer configuration is a configuration that has at most one transmit message buffer and at most one receive message buffer assigned to a slot *S*. The simple configuration used in this section utilizes two message buffers, one single buffered transmit message buffer and one receive message buffer.

The transmit message buffer has the message buffer number *t* and has following configuration

Table 518. Transmit Buffer Configuration

Register	Field	Value	Description
MBCCSR*t*	MTD	1	transmit buffer
MBCCFR*t*	MTM	0	event transition mode
	CHA	1	assigned to channel A
	CHB	0	not assigned to channel B
	CCFE	1	cycle counter filter enabled
	CCFMSK	000011	cycle set = {4n} = {0,4,8,12,…}
	CCFVAL	000000	
MBFIDR*t*	FID	*S*	assigned to slot S

The availability of data in the transmit buffer is indicated by the commit bit MBCCSRt[CMT] and the lock bit MBCCSRt[LCKS].

The receive message buffer has the message buffer number r and has following configuration.

Table 519. Receive Buffer Configuration

Register	Field	Value	Description
MBCCSR*r*	MTD	0	receive buffer
MBCCFR*r*	MTM	-	n/a
	CHA	1	assigned to channel A
	CHB	0	not assigned to channel B
	CCFE	1	cycle counter filter enabled
	CCFMSK	000001	cycle set = {2n} = {0,2,4,6,…}
	CCFVAL	000000	
MBFIDR*r*	FID	*S*	subscribed slot

Furthermore the assumption is that both message buffers are enabled (MBCCSRt[EDS] = 1 and MBCCSRr[EDS] = 1)

NOTE

The cycle set {4n+2} = {2,6,10,…} is assigned to the receive buffer only.

The cycle set {4n} = {0,4,8,12,…} is assigned to both buffers.

56.7.7.2 Behavior in static segment

In this case, both message buffers are assigned to a slot *S* in the *static* segment.

The configuration of a transmit buffer for a static slot *S* assigns this slot to the node as a transmit slot. The FlexRay protocol requires:

- When a slot occurs, if the slot is assigned to a node on a channel that node must transmit either a normal frame or a null frame on that channel. Specifically, a null frame will be sent if there is no data ready, or if there is no match on a transmit filter (cycle counter filtering, for example).

Regardless of the availability of data and the cycle counter filter, the node will transmit a frame in the static slot *S*. In any case, the result of the message buffer search will be the transmit message buffer *t*. The receive message buffer *r* will not be found, no reception is possible.

56.7.7.3 Behavior in dynamic segment

In this case, both message buffers are assigned to a slot *S* in the *dynamic* segment. The FlexRay protocol requires:

- When a slot occurs, if a slot is assigned to a node on a channel that node only transmits a frame on that channel if there is data ready and there is a match on relevant transmit filters (no null frames are sent).

The transmission of a frame in the dynamic segment is determined by the availability of data and the match of the cycle counter filter of the transmit message buffer.

56.7.7.3.1 Transmit Data Not Available

If transmit data are *not available*, i.e. the transmit buffer is not committed MBCCSR*t*[CMT]=0 and/or locked MBCCSR*t*[LCKS]=1,

1. for the cycles in the set {4n}, which is assigned to both buffers, the receive buffer will be found and the node can receive data, and
2. for the cycles in the set {4n+2}, which is assigned to the receive buffer only, the receive buffer will be found and the node can receive data.

The receive cycles are shown in the following figure.



Figure 509. Transmit Data Not Available

56.7.7.3.2 Transmit Data Available

If transmit data are *available*, i.e. the transmit buffer is committed MBCCSR*t*[CMT]=1 and not locked MBCCSR*t*[LCKS]=0,

1. for the cycles in the set {4n}, which is assigned to both buffers, the transmit buffer will be found and the node transmits data.
2. for the cycles in the set {4n+2}, which is assigned to the receive buffer only, the receive buffer will be found and the node can receive data.

The receive and transmit cycles are shown in the following figure.

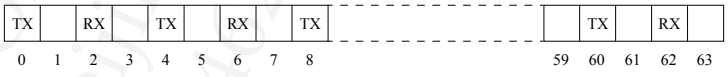


Figure 510. Transmit Data Available

56.8 Glossary

BCU	Buffer control unit—handles MB access
BMIF	Bus master interface—provides master access to the FlexRay memory area
CC	Communication controller
CDC	Clock domain crossing unit
CHI	Controller host interface
EBI	External bus interface
FlexRay memory area	Memory area to store the physical MB payload data, frame header, frame and slot status, and synchronization frame related tables
FSS	Frame start sequence
HIF	Host interface—provides host access to controller
Host	The FlexRay CC host MCU
Key slot	Transmits the startup frame, sync frame, or designated single slot frame
LUT	Lookup table—stores MB header index values
LRAM	LUT RAM—the module internal memory to store message buffer configuration data and data field offsets for individual message buffers and receive shadow buffers
MB	Message buffer
Mini-slot	An interval of time within the dynamic segment of the communication cycle that is of constant duration (in terms of macroticks) and that is used by the synchronized FTDMA media access scheme to manage media arbitration
MBIDX	MB index—the position of a header field entry within the header area; if the header area is accessed as an array, this is the same as the array index of the entry
MCU	Microcontroller unit
µT	Microtick
MT	Macrotick
MTS	Media access test symbol
Message frame	Frame for which the null frame indicator is 1
NMV	Network management vector
Normal frame	Null frame or message frame for which both the sync frame indicator and the startup frame indicator are 0
Null frame	Frame for which the null frame indicator is 0
NIT	Network idle time
PE	Protocol engine
POC	Protocol operation control; each state of the POC is denoted by POC:state
Rx	Reception
Slot	An interval of time during which access to a communication channel is granted exclusively to a specific node for the transmission of a frame with a frame ID corresponding to the slot
SEQ	Sequencer engine
SU	Status update

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca5460e5-4821-48a8-b253-39d3ab6241

SECDED	Single-bit error correction, double-bit error detection
System memory	Memory that contains the FlexRay memory area
System bus	Bus that connects the controller and system memory
Sync frame	Null frame or message frame for which the sync frame indicator is 1
Startup frame	Null frame or message frame for which both the sync frame indicator and the startup frame indicator are 1
TCU	Time control unit
Tx	Transmission

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 57
Cross-Triggering Unit (CTU)

57.1 Chip-specific CTU information

57.1.1 CTU connections

The following table shows the CTU input triggers, ADC interconnections, and TGSISR assignments in this chip.

Table 520. CTU connections

Input module	Input signal	TGSISR I/P	Source	Driver signal
CTU	PWM_REL	0	FTM_0	CH0
CTU	PWM_ODD_0	1	FTM_0	CH1
CTU	PWM_ODD_1	2	FTM_0	CH2
CTU	PWM_ODD_2	3	FTM_1	CH0
CTU	PWM_ODD_3	4	FTM_1	CH1
CTU	PWM_EVEN_0	5	FTM_1	CH2
CTU	PWM_EVEN_1	6	PIT_0	TRIGGER[6]
CTU	—	7–14 (Reserved)		
CTU	EXT_IN	15	External pin	External pin
ADC_0[1]	TRIGGER	—	CTU	TRIGGER_0
ADC_0[1]	CMD	—	CTU	CMD_0
ADC_1[2]	TRIGGER	—	CTU	TRIGGER_1
ADC_1[2]	CMD	—	CTU	CMD_1

1. Throughout the CTU chapter, ADC_0 is referred to as ADC A (ADC_A).
2. Throughout the CTU chapter, ADC_1 is referred to as ADC B (ADC_B).

57.2 Introduction

In PWM driven systems, it is important to schedule the acquisition of the state variables with respect to the PWM cycle. State variables are obtained through the following peripherals: ADC, position counter (for example, quadrature decoder, resolver, and sine-cos sensor), and PWM duty cycle decoder.

The Cross-Triggering Unit (CTU) is intended to completely avoid CPU involvement in the acquisition of state variables during a control cycle defined by the PWM cycle, the half PWM cycle, or a multiple number of PWM cycles. Several CTU registers are double-buffered, thus allowing coherent update during the transition from one control cycle to the next. Thus, double-buffered registers updated at control cycle N are updated during the transition from control cycle N to control cycle N+1.

The CTU provides four FIFO memory blocks to temporarily store ADC results for CPU access.

57.3 Block diagram

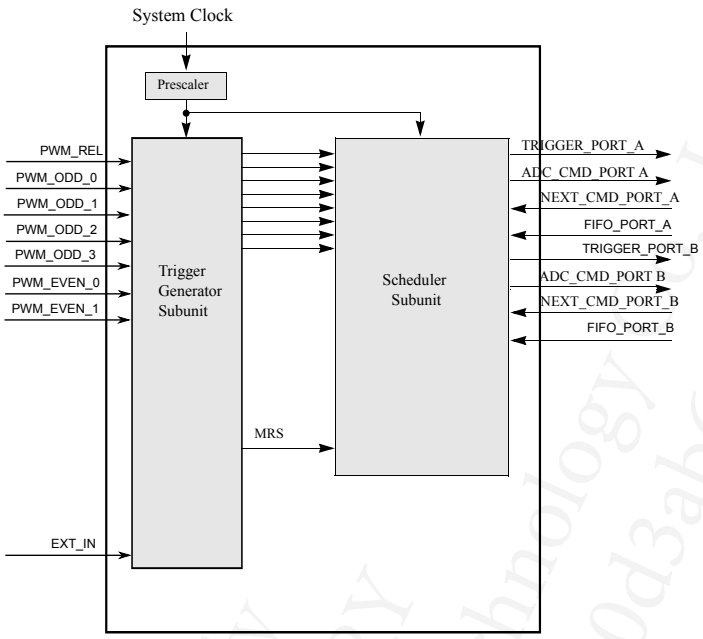


Figure 511. CTU block diagram

57.4 CTU overview

The CTU receives several signals from different sources such as FTM , timers, position decoder These signals are then processed to generate up to eight trigger events. An input event can be a rising edge, a falling edge, or both edges of the incoming signal. The output can be a pulse, an ADC command, or a stream of consecutive ADC commands for oversampling support. The outputs are targeted to one or more ADCs.

The CTU provides the following features:

- Cross triggering between ADC, FTM timers, on-chip general purpose timers
- Double-buffered trigger unit that generates up to eight triggers
- Trigger generation unit configurable in sequential or triggered modes
- Trigger delay functionality to compensate for the delay of external low-pass filters
- Double-buffered ADC command list pointers
- Double-buffered ADC conversion command list
- Each trigger has the capability to generate consecutive conversion commands
- ADC conversion commands allow two ADCs to be controlled synchronously with queue selection to store conversion results
- DMA support

In a typical application, the CTU interfaces with the following peripherals:

- FTM generators for motor control
- Timers

The inputs are 16 digital signals. The CTU detects the rising and/or falling edges for each one of them.

Figure 511 shows a high-level CTU block diagram. The CTU hardware architecture consists of two subunits:

- Trigger generator

- Scheduler

The Trigger generator subunit receives CTU input signals and selects the active edges to generate the Master Reload signal. The signal is used to load several CTU double buffered registers. See sections TGS in triggered mode and TGS in sequential mode for more details about the Master Reload signal. The Trigger generator subunit also generates up to eight trigger event signals are used by the Scheduler subunit.

This subunit generates the trigger event output according to the trigger event signals from the Trigger generator subunit. The trigger event output starts an ADC conversion or generates output signals to trigger events for IPs in the SoC. See section Scheduler subunit (SU) for more detail about the Scheduler subunit.

57.4.1 Modes of operation

The CTU has two main modes of operation:

- Triggered mode: This input event from the CTU interface is used to generate a sequence of up to eight triggers to several outputs such as the ADCs, timers, and external triggers. Internal sequencer logic is used to schedule the triggers based upon the input event occurrence.
- Sequential mode: Each input event generates only one trigger for the selected output, such as ADCs, timers, and external triggers.

57.5 Signal description

The following table shows the chip-level signals for the CTU module.

Table 521. Signal properties

Name	Function
EXT_IN	Input pin for external triggers

57.6 CTU register descriptions

CTU registers are implemented in two sizes:

- 32-bit registers: byte access for write operations and 32-bit access for read operations
- 16-bit registers: byte access for write operations and 32-bit access for read operations

If a 32-bit write operation is performed over a 16-bit register, the write operation is performed on the 4 aligned bytes. Read operation should be performed only at 32-bit boundary.

NOTE

Reserved bit registers, when available for write operations, should be written always with logic 0. Read access does not return meaningful data from reserved bit registers. This module does not issue transfer errors when trying to access its reserved locations.

The following tables describe the CTU memory mapped registers.

57.6.1 CTU memory map

CTU base address: 401F_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Trigger Generator Subunit Input Selection Register (TGSISR)	32	RW	0000_0000h
4h	Trigger Generator Subunit Control Register (TGSCR)	16	RW	0000h
6h	Trigger Compare Register (T0CR)	16	RW	0000h
8h	Trigger Compare Register (T1CR)	16	RW	0000h
Ah	Trigger Compare Register (T2CR)	16	RW	0000h
Ch	Trigger Compare Register (T3CR)	16	RW	0000h
Eh	Trigger Compare Register (T4CR)	16	RW	0000h
10h	Trigger Compare Register (T5CR)	16	RW	0000h
12h	Trigger Compare Register (T6CR)	16	RW	0000h
14h	Trigger Compare Register (T7CR)	16	RW	0000h
16h	TGS Counter Compare Register (TGSCCR)	16	RW	0000h
18h	TGS Counter Reload Register (TGSCRR)	16	RW	0000h
1Ch	Commands List Control Register 1 (CLCR1)	32	RW	0000_0000h
20h	Commands List Control Register 2 (CLCR2)	32	RW	0000_0000h
24h	Trigger Handler Control Register 1 (THCR1)	32	RW	0000_0000h
28h	Trigger Handler Control Register 2 (THCR2)	32	RW	0000_0000h
2Ch - 5Ah	Commands List Register A for ADC single-conversion mode commands (CLR_A_1 - CLR_A_24)	16	RW	0000h
2Ch - 5Ah	Command List Register B for ADC dual-conversion mode commands (CLR_B_1 - CLR_B_24)	16	RW	0000h
2Ch - 5Ah	Command List Register C for self-test commands (CLR_C_1 - CLR_C_24)	16	RW	0000h
6Ch	FIFO DMA Control Register (FDCR)	16	RW	0F00h
70h	FIFO Control Register (FCR)	32	RW	0000_0000h
74h	FIFO Threshold Register (FTH)	32	RW	0000_0000h
7Ch	FIFO Status Register (FST)	32	W1C	0000_2222h
80h - 8Ch	FIFO Right Aligned Data Register (FR0 - FR3)	32	RO	0000_0000h
A0h - ACh	FIFO Signed Left Aligned Data Register (FL0 - FL3)	32	RO	0000_0000h
C0h	Error Flag Register (EFR)	16	W1C	0000h
C2h	Interrupt Flag Register (IFR)	16	W1C	01FEh
C4h	Interrupt/DMA Register (IR)	16	RW	0000h
C6h	Control ON Time Register (COTR)	16	RW	0000h
C8h	Control Register (CR)	16	RW	0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
CAh	Digital Filter Register (DFR)	16	RW	0000h
CCh	Expected Value A Register (EXPAR)	16	RW	FFFFh
CEh	Expected Value B Register (EXPBR)	16	RW	FFFFh
D0h	Counter Range Register (CNTRNGR)	16	RW	0000h
D4h	List Control/Status Register (LISTCSR)	32	RW	0000_0000h

57.6.2 Trigger Generator Subunit Input Selection Register (TGSISR)

Offset

Register	Offset
TGSISR	0h

Function

The TGSISR register is double buffered. The load from the corresponding buffer to the register is controlled by the TGSISR_RE bit in the CTU_CR register. The TGSISR_RE bit is self negated. Once set, this bit remains set until the TGSISR register is updated from the corresponding buffer register. Reads to the TGSISR register actually return the content of the buffer and not the content of the register. Thus, a read will return the most recent data written to the register address even though the buffer data is not synchronized with the register data.

This device has of triggers available, bits corresponding to unavailable triggers, if any, have no effect.

NOTE

For the particular input connection assigned to each field in this register, see the CTU's chip-specific configuration details.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	I15_FE	I15_RE	Reserved		Reserved		Reserved		Reserved		Reserved		Reserved		Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		I6_FE	I6_RE	I5_FE	I5_RE	I4_FE	I4_RE	I3_FE	I3_RE	I2_FE	I2_RE	I1_FE	I1_RE	I0_FE	I0_RE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing C-L Automotive Technology Co., Ltd.
c8c42c5-4821-48a8-b2c1-390a3ab62f61

Fields

Field	Function
31 I15_FE	Input 15 Falling Edge Enable. 0b - Disabled 1b - Enabled
30 I15_RE	Input 15 Rising Edge Enable. 0b - Disabled 1b - Enabled
29-28 —	Reserved This field is not used, writing to this has no effect.
27-26 —	Reserved This field is not used, writing to this has no effect.
25-24 —	Reserved This field is not used, writing to this has no effect.
23-22 —	Reserved This field is not used, writing to this has no effect.
21-20 —	Reserved This field is not used, writing to this has no effect.
19-18 —	Reserved This field is not used, writing to this has no effect.
17-16 —	Reserved This field is not used, writing to this has no effect.
15-14 —	Reserved This field is not used, writing to this has no effect.
13 I6_FE	Input 6 Falling Edge Enable. 0b - Disabled 1b - Enabled
12 I6_RE	Input 6 Rising Edge Enable. 0b - Disabled 1b - Enabled
11 I5_FE	Input 5 Falling Edge Enable. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
10 I5_RE	Input 5 Rising Edge Enable. 0b - Disabled 1b - Enabled
9 I4_FE	Input 4 Falling Edge Enable. 0b - Disabled 1b - Enabled
8 I4_RE	Input 4 Rising Edge Enable. 0b - Disabled 1b - Enabled
7 I3_FE	Input 3 Falling Edge Enable. 0b - Disabled 1b - Enabled
6 I3_RE	Input 3 Rising edge Enable 0b - Disabled 1b - Enabled
5 I2_FE	Input 2 Falling Edge Enable. 0b - Disabled 1b - Enabled
4 I2_RE	Input 2 Rising Edge Enable. 0b - Disabled 1b - Enabled
3 I1_FE	Input 1 Falling Edge Enable. 0b - Disabled 1b - Enabled
2 I1_RE	Input 1 Rising edge Enable 0b - Disabled 1b - Enabled
1 I0_FE	Input 0 Falling Edge Enable. 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

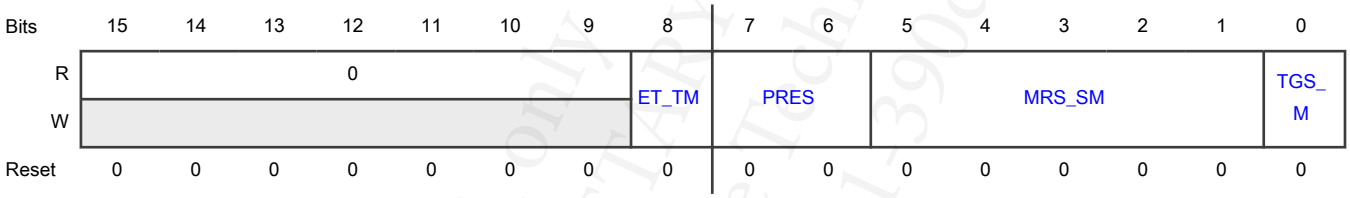
Field	Function
0 I0_RE	Input 0 Rising Edge Enable. 0b - Disabled 1b - Enabled

57.6.3 Trigger Generator Subunit Control Register (TGSCR)

Offset

Register	Offset
TGSCR	4h

Diagram



Fields

Field	Function
15-9 —	Reserved
8 ET_TM	Enable Toggle Mode for external Trigger. 0b - Pulse mode 1b - Toggle mode
7-6 PRES	Prescaler selection bits for TGS and SU. This prescaler is used by the TGS counter. 00b - Value is 1 01b - Value is 2 10b - Value is 3 11b - Value is 4
5-1 MRS_SM	MRS Selection in Sequential Mode-5 bits to select one of the 2 x 16inputs shown in the Trigger generator subunit input selection (TGSISR) register.
0	Trigger Generator Subunit Mode

Table continues on the next page...

Table continued from the previous page...

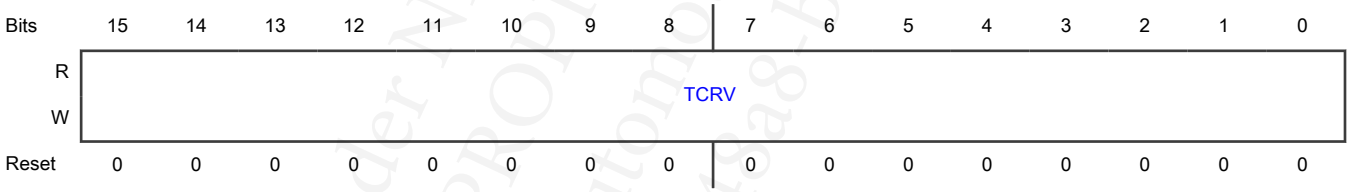
Field	Function
TGS_M	0b - Triggered Mode 1b - Sequential Mode

57.6.4 Trigger Compare Register (T0CR - T7CR)

Offset

Register	Offset
T0CR	6h
T1CR	8h
T2CR	Ah
T3CR	Ch
T4CR	Eh
T5CR	10h
T6CR	12h
T7CR	14h

Diagram



Fields

Field	Function
15-0 TCRV	Trigger Compare Register Value

57.6.5 TGS Counter Compare Register (TGSCCR)

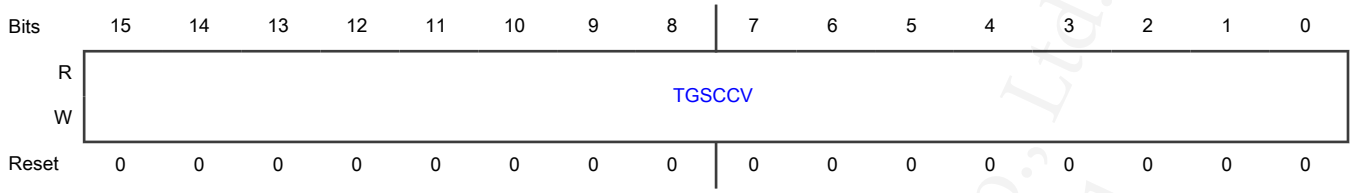
Offset

Register	Offset
TGSCCR	16h

Provided under NDA only
COMPANY PROPRIETARY
Beijing C-TY Automotive Technology Co., Ltd.
c8462e5-4821-43a8-b251-390d3ab624c1

Function

Diagram



Fields

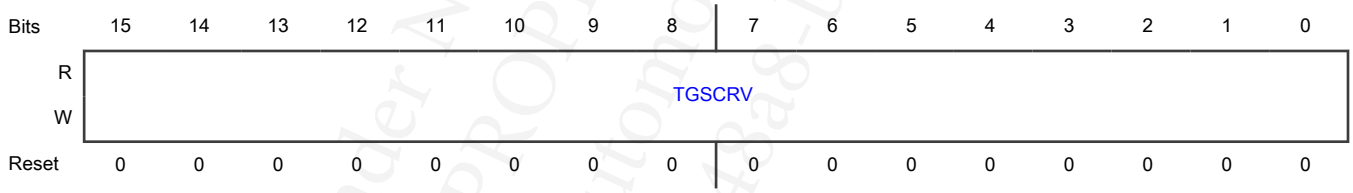
Field	Function
15-0 TGSCCV	TGS Counter Compare Value. The value in this register is compared against the TGS counter. When a match occurs, the TGS counter is stopped.

57.6.6 TGS Counter Reload Register (TGSCRR)

Offset

Register	Offset
TGSCRR	18h

Diagram



Fields

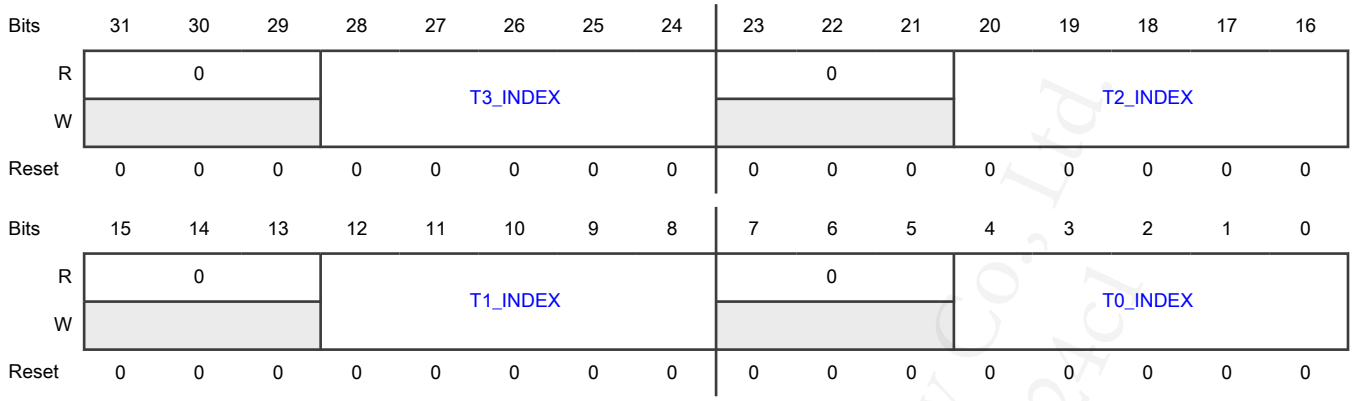
Field	Function
15-0 TGSCRV	TGSCRV TGS Counter Reload Value. When an MRS occurs, the value in this register is used to reload the TGS counter.

57.6.7 Commands List Control Register 1 (CLCR1)

Offset

Register	Offset
CLCR1	1Ch

Diagram



Fields

Field	Function
31-29 —	Reserved
28-24 T3_INDEX	Trigger 3 Commands List 1st command address. This is the address of the first command in the LIST associated with this trigger. Valid values range from 0 through 23.
23-21 —	Reserved
20-16 T2_INDEX	Trigger 2 Commands List 1st command address. This is the address of the first command in the LIST associated with this trigger. Valid values range from 0 through 23.
15-13 —	Reserved
12-8 T1_INDEX	Trigger 1 Commands List 1st command address. This is the address of the first command in the LIST associated with this trigger. Valid values range from 0 through 23.
7-5 —	Reserved
4-0 T0_INDEX	Trigger 0 Commands List 1st command address. This is the address of the first command in the LIST associated with this trigger. Valid values range from 0 through 23.

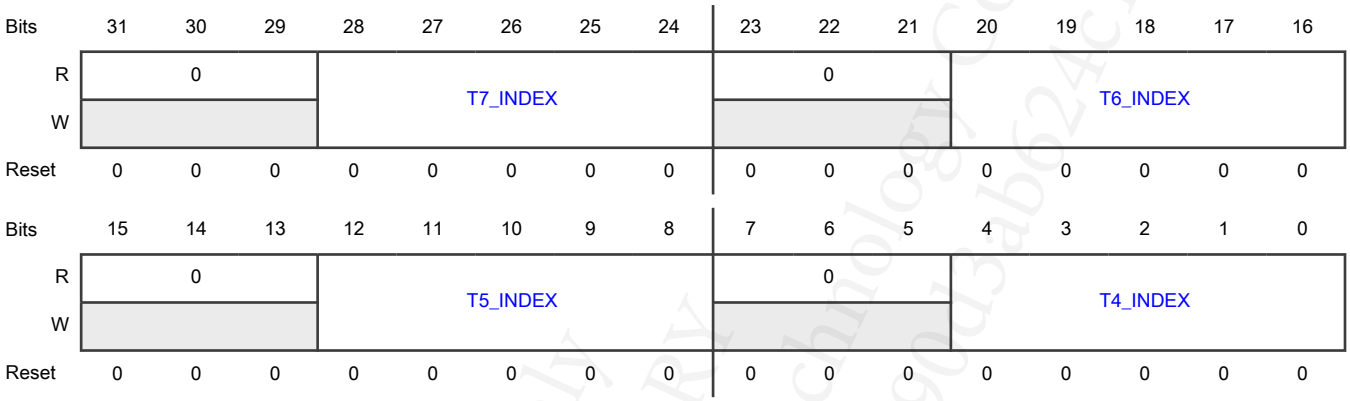
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c40e5-4821-48a8-b251-390d8ab624c1

57.6.8 Commands List Control Register 2 (CLCR2)

Offset

Register	Offset
CLCR2	20h

Diagram



Fields

Field	Function
31-29 —	Reserved
28-24 T7_INDEX	T7_INDEX Trigger 7 Commands List 1st command address. This is the address of the first command in the LIST associated with this trigger. Valid values range from 0 through 23.
23-21 —	Reserved
20-16 T6_INDEX	Trigger 6 Commands List 1st command address. This is the address of the first command in the LIST associated with this trigger. Valid values range from 0 through 23.
15-13 —	Reserved
12-8 T5_INDEX	Trigger 5 Commands List 1st command address. This is the address of the first command in the LIST associated with this trigger. Valid values range from 0 through 23.
7-5	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4-0 T4_INDEX	Trigger 4 Commands List 1st command address. This is the address of the first command in the LIST associated with this trigger. Valid values range from 0 through 23.

57.6.9 Trigger Handler Control Register 1 (THCR1)

Offset

Register	Offset
THCR1	24h

Function

The THCR1 register is double buffered. The update from the corresponding buffer register is controlled by the MRS signal. The THCR1 and THCR2 registers are together organized into 8 groups of enable bits. Each group is associated with one trigger from the Trigger subunit. Each group of enable bits enables corresponding to a master trigger enable, External Trigger outputs, and the ADC command list enable, respectively. If the master trigger enable Tn_E is cleared, the trigger is disabled and the other trigger enable bits in the group have no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	T3_E	Reserved	Reserved	Reserved	Reserved	Reserved	T3_ADCE	0	T2_E	Reserved	Reserved	Reserved	Reserved	Reserved	T2_ADCE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	T1_E	Reserved	Reserved	Reserved	Reserved	Reserved	T1_ADCE	0	T0_E	Reserved	Reserved	Reserved	Reserved	Reserved	T0_ADCE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30 T3_E	Trigger 3 Enable.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
29 —	Reserved This field is not used, writing to this has no effect.
28 —	Reserved This field is not used, writing to this has no effect.
27 —	Reserved This field is not used, writing to this has no effect.
26 —	Reserved This field is not used, writing to this has no effect.
25 —	Reserved This field is not used, writing to this has no effect.
24 T3_ADCE	Trigger 3 ADC command output Enable. 0b - Disabled 1b - Enabled
23 —	Reserved This field is not used, writing to this has no effect.
22 T2_E	Trigger 2 Enable. 0b - Disabled 1b - Enabled
21 —	Reserved This field is not used, writing to this has no effect.
20 —	Reserved This field is not used, writing to this has no effect.
19 —	Reserved This field is not used, writing to this has no effect.
18 —	Reserved This field is not used, writing to this has no effect.
17	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	This field is not used, writing to this has no effect.
16 T2_ADCE	Trigger 2 ADC command output Enable. 0b - Disabled 1b - Enabled
15 —	Reserved This field is not used, writing to this has no effect.
14 T1_E	Trigger 1 Enable. 0b - Disabled 1b - Enabled
13 —	Reserved This field is not used, writing to this has no effect.
12 —	Reserved This field is not used, writing to this has no effect.
11 —	Reserved This field is not used, writing to this has no effect.
10 —	Reserved This field is not used, writing to this has no effect.
9 —	Reserved This field is not used, writing to this has no effect.
8 T1_ADCE	Trigger 1 ADC command output Enable. 0b - Disabled 1b - Enabled
7 —	Reserved This field is not used, writing to this has no effect.
6 T0_E	Trigger 0 Enable. 0b - Disabled 1b - Enabled
5 —	Reserved This field is not used, writing to this has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 —	Reserved This field is not used, writing to this has no effect.
3 —	Reserved This field is not used, writing to this has no effect.
2 —	Reserved This field is not used, writing to this has no effect.
1 —	Reserved This field is not used, writing to this has no effect.
0 T0_ADCE	Trigger 0 ADC command output Enable. 0b - Disabled 1b - Enabled

57.6.10 Trigger Handler Control Register 2 (THCR2)

Offset

Register	Offset
THCR2	28h

Function

The THCR2 register is double buffered. The update from the corresponding buffer register is controlled by the MRS signal. The THCR1 and THCR2 registers are together organized into 8 groups of enable bits. Each group is associated with one trigger from the Trigger subunit. Each group of enable bits enables corresponding to a master trigger enable, External Trigger outputs, and the ADC command list enable, respectively. If the master trigger enable Tn_E is cleared, the trigger is disabled and the other trigger enable bits in the group have no effect.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	T7_E	Reserved	Reserved	Reserved	Reserved	Reserved	T7_ADCE	0	T6_E	Reserved	Reserved	Reserved	Reserved	Reserved	T6_ADCE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	T5_E	Reserved	Reserved	Reserved	Reserved	Reserved	T5_ADCE	0	T4_E	Reserved	Reserved	Reserved	Reserved	Reserved	T4_ADCE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
NXP CONFIDENTIAL PROPRIETARY
Copyright (c) 2016-2021 Nxp Automotive Technology Co., Ltd.
All rights reserved.
Printed by 6c4615c4821c2a8-b251-390d3ab624c1

Fields

Field	Function
31 —	Reserved
30 T7_E	Trigger 7 Enable. 0b - Disabled 1b - Enabled
29 —	Reserved This field is not used, writing to this has no effect.
28 —	Reserved This field is not used, writing to this has no effect.
27 —	Reserved This field is not used, writing to this has no effect.
26 —	Reserved This field is not used, writing to this has no effect.
25 —	Reserved This field is not used, writing to this has no effect.
24 T7_ADCE	Trigger 7 ADC command output Enable. 0b - Disabled 1b - Enabled
23 —	Reserved
22 T6_E	Trigger 6 Enable. 0b - Disabled 1b - Enabled
21 —	Reserved This field is not used, writing to this has no effect.
20 —	Reserved This field is not used, writing to this has no effect.
19 —	Reserved This field is not used, writing to this has no effect.

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 —	Reserved This field is not used, writing to this has no effect.
17 —	Reserved This field is not used, writing to this has no effect.
16 T6_ADCE	Trigger 6 ADC command output Enable. 0b - Disabled 1b - Enabled
15 —	Reserved
14 T5_E	Trigger 5 Enable. 0b - Disabled 1b - Enabled
13 —	Reserved This field is not used, writing to this has no effect.
12 —	Reserved This field is not used, writing to this has no effect.
11 —	Reserved This field is not used, writing to this has no effect.
10 —	Reserved This field is not used, writing to this has no effect.
9 —	Reserved This field is not used, writing to this has no effect.
8 T5_ADCE	Trigger 5 ADC command output Enable. 0b - Disabled 1b - Enabled
7 —	Reserved
6 T4_E	Trigger 4 Enable. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
5 —	Reserved This field is not used, writing to this has no effect.
4 —	Reserved This field is not used, writing to this has no effect.
3 —	Reserved This field is not used, writing to this has no effect.
2 —	Reserved This field is not used, writing to this has no effect.
1 —	Reserved This field is not used, writing to this has no effect.
0 T4_ADCE	Trigger 4 ADC command output Enable. 0b - Disabled 1b - Enabled

57.6.11 Commands List Register A for ADC single-conversion mode commands (CLR_A_1 - CLR_A_24)

Offset

For n = 1 to 24:

Register	Offset
CLR_A_n	2Ah + (n × 2h)

Function

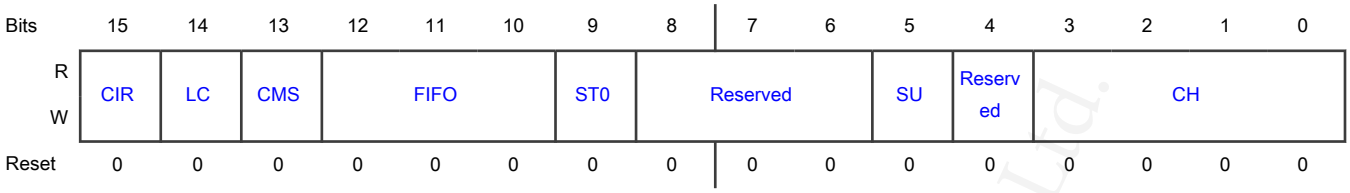
The Command List Register (CLR_x_n), where n = 1,.....,24, has one of three formats (CLR_A_n, CLR_B_n, CLR_C_n) depending on the setting of its CMS and ST0 bits. If CMS=0 and ST0=0, the single conversion mode is selected and the CLR register assumes the CLR_A_n format described in the following figure.

NOTE

Reserved fields of this register are read/write as they share same physical register and address. The value of reserved fields must be zero.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e4-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
15 CIR	Command execution Interrupt Request enable bit. 0b - Disabled 1b - Enabled
14 LC	Last Command bit 0b - Not last 1b - Last
13 CMS	Conversion Mode Selection. Must be 0 in this register format. 0b - Single conversion mode 1b - Dual conversion mode
12-10 FIFO	FIFO used for ADC Port A / Port B. See CLR_A[SU] bit for Port A versus Port B selection. 000b - FIFO_0 select 111b - FIFO_7 select
9 ST0	ST0 Self Test mode control 0. Must be 0 in this register format.
8-6 —	Reserved The value of this field must be zero.
5 SU	ADC Port A / Port B selection. 0b - ADC Port A selected 1b - ADC Port B selected
4 —	Reserved The value of this field must be zero.
3-0 CH	ADC Port channel number.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

57.6.12 Command List Register B for ADC dual-conversion mode commands (CLR_B_1 - CLR_B_24)

Offset

For n = 1 to 24:

Register	Offset
CLR_B_n	2Ah + (n × 2h)

Function

The Command List Register (CLR_x_n), where n = 1,.....,24, has one of three formats (CLR_A_n, CLR_B_n, CLR_C_n) depending on the setting of its CMS and ST0 bits. If CMS=1 and ST0=0, the dual conversion mode is selected and the CLR register assumes the CLR_B_n format described in the following figure.

NOTE

Reserved fields of this register are read/write as they share same physical register and address. The value of reserved fields must be zero.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CIR	LC	CMS	FIFO			ST0		CH_B			Reserved	CH_A			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 CIR	Command execution Interrupt Request enable bit. 0b - Disabled 1b - Enabled
14 LC	Last Command Bit 0b - Not last 1b - Last
13 CMS	Conversion Mode Selection Must be 1 in this register format. 0b - Single conversion mode 1b - Dual conversion mode
12-10 FIFO	FIFO used for ADC Port A / Port B. See CLR_A[SU] bit for Port A versus Port B selection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - FIFO_0 select 111b - FIFO_7 select
9 ST0	Sel-Test Mode Control 0 The value of this field must be kept 0 for the correct operation of CLR_B format.
8-5 CH_B	ADC Port B Channel Number
4 —	Reserved The value of this field must be zero.
3-0 CH_A	ADC Port A Channel Number

57.6.13 Command List Register C for self-test commands (CLR_C_1 - CLR_C_24)

Offset

For n = 1 to 24:

Register	Offset
CLR_C_n	2Ah + (n × 2h)

Function

The Command List Register (CLR_x_n), where n = 1,.....,24, has one of three formats (CLR_A_n, CLR_B_n, CLR_C_n) depending on the setting of its ST0 bit. You must first set ST0 to 1 to set the CLR register in the self-test mode register format. In this case, the CLR register assumes the CLR_C_n format described in the following figure.

NOTE

Reserved fields of this register are read/write as they share same physical register and address. The value of reserved fields must be zero.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c462e5-4821-3ca8-b251-390d3ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CIR	LC	ST1	ST_CMS	ST_SU	Reserved	ST0	ALG					BSIZE			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 CIR	Command Execution Interrupt Request enable bit (this bit has no influence in this mode). **NOTE** CIR bit value has no influence in self-test mode. The interrupt request is disabled and cannot be enabled by using this bit in this mode. 0b - Disabled 1b - Enabled
14 LC	Last command bit. 0b - Not last 1b - Last
13 ST1	Self Test command control 1. ST1 must be 0 for a self test command.
12 ST_CMS	Conversion Mode Selection for Self Test command. 0b - Single Conversion Mode 1b - Dual Conversion Mode
11 ST_SU	Self Test Selection Unit bit. Usable only if ST_CMS is 0. 0b - ADC Port A 1b - ADC Port B
10 —	Reserved The value of this field must be zero.
9 ST0	Self Test command control 0. ST0 must 1 in this register format.
8-7 ALG	Algorithm scheduled 00b - Algorithm S 01b - Reserved 10b - Algorithm C 11b - Full algorithm (S + C)

Table continues on the next page...

Table continued from the previous page...

Field	Function
6-0 BSIZE	Burst size of the algorithm iteration 0000000b - Single step execution 1111111b - 512 step execution with one trigger

57.6.14 FIFO DMA Control Register (FDCR)

Offset

Register	Offset
FDCR	6Ch

Function

The FDCR register enables the DMA requests issued by the CTU FIFOs. The DMA request occurs when it is enabled and the corresponding FIFO had reached a threshold defined in register FTH.

NOTE

Interrupt and DMA are controlled independently, thus FIFO DMA operation and interrupts can be enabled at the same time.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				Reserved	Reserved	Reserved	Reserved	Reserved				DE3	DE2	DE1	DE0
W	W1C				W1C	W1C	W1C	W1C	W1C							
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

Fields

Field	Function
15-12 —	Reserved
11 —	Reserved **NOTE** The reset value of this bit is 1b. If this reset value is to be preserved during a write to this bit, 0b must be written. This bit is a "write 1 to clear" bit, which means a write of 1b changes the bit's value to 0b.

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 —	Reserved **NOTE** The reset value of this bit is 1b. If this reset value is to be preserved during a write to this bit, 0b must be written. This bit is a "write 1 to clear" bit, which means a write of 1b changes the bit's value to 0b.
9 —	Reserved **NOTE** The reset value of this bit is 1b. If this reset value is to be preserved during a write to this bit, 0b must be written. This bit is a "write 1 to clear" bit, which means a write of 1b changes the bit's value to 0b.
8 —	Reserved **NOTE** The reset value of this bit is 1b. If this reset value is to be preserved during a write to this bit, 0b must be written. This bit is a "write 1 to clear" bit, which means a write of 1b changes the bit's value to 0b.
7-4 —	Reserved **NOTE** The reset value of each individual bit is 0b1. If this reset value is to be preserved during a write to these bits, and 0b0 must be written to each of these bits.
3 DE3	FIFO 3 DMA enable. 0b - Disabled 1b - Enabled
2 DE2	FIFO 2 DMA enable. 0b - Disabled 1b - Enabled
1 DE1	FIFO 1 DMA enable. 0b - Disabled 1b - Enabled
0 DE0	FIFO 0 DMA enable. 0b - Disabled 1b - Enabled

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
c8c462e5-4821-48a8-b251-39d3ab24c1

57.6.15 FIFO Control Register (FCR)

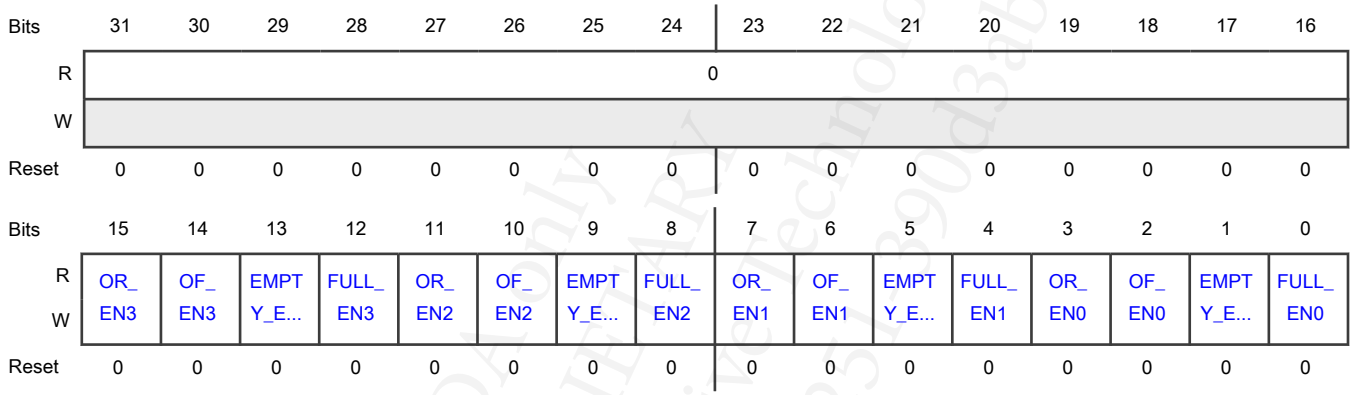
Offset

Register	Offset
FCR	70h

Function

This register implements interrupt enable bits to signal FIFO behavior. FIFO overrun, overflow, empty, and full indication from one of the FIFOs are OR-ed together to indicate through one interrupt line that one or more flags are active. After receiving an interrupt the FST register should be checked to evaluate what caused that interrupt. There is one interrupt line per FIFO module.

Diagram



Fields

Field	Function
31-16 —	Reserved
15 OR_EN3	FIFO 3 Overrun interrupt enable. 0b - Disabled 1b - Enabled
14 OF_EN3	FIFO 3 threshold Overflow interrupt enable. 0b - Disabled 1b - Enabled
13 EMPTY_EN3	FIFO 3 Empty interrupt enable. 0b - Disabled 1b - Enabled
12	FIFO 3 Full interrupt enable.

Table continues on the next page...

Table continued from the previous page...

Field	Function
FULL_EN3	0b - Disabled 1b - Enabled
11 OR_EN2	FIFO 2 Overrun interrupt enable. 0b - Disabled 1b - Enabled
10 OF_EN2	FIFO 2 threshold Overflow interrupt enable. 0b - Disabled 1b - Enabled
9 EMPTY_EN2	FIFO 2 Empty interrupt enable. 0b - Disabled 1b - Enabled
8 FULL_EN2	FIFO 2 Full interrupt enable. 0b - Disabled 1b - Enabled
7 OR_EN1	FIFO 1 Overrun interrupt enable. 0b - Disabled 1b - Enabled
6 OF_EN1	FIFO 1 threshold Overflow interrupt enable. 0b - Disabled 1b - Enabled
5 EMPTY_EN1	FIFO 1 Empty interrupt enable. 0b - Disabled 1b - Enabled
4 FULL_EN1	FIFO 1 Full interrupt enable. 0b - Disabled 1b - Enabled
3 OR_EN0	FIFO 0 Overrun interrupt enable. 0b - Disabled 1b - Enabled
2	FIFO 0 threshold Overflow interrupt enable.

Table continues on the next page...

Table continued from the previous page...

Field	Function
OF_EN0	0b - Disabled 1b - Enabled
1 EMPTY_EN0	FIFO 0 Empty interrupt enable. 0b - Disabled 1b - Enabled
0 FULL_EN0	FIFO 0 Full interrupt enable. 0b - Disabled 1b - Enabled

57.6.16 FIFO Threshold Register (FTH)

Offset

Register	Offset
FTH	74h

Function

This register defines the threshold value for four FIFO modules. The threshold is used for comparison against the FIFO pointer and to evaluate if an overflow condition occurred for that FIFO. The overflow flag indicates that a certain amount of data, defined by the threshold value, is available in the FIFO for read. The overflow condition is set if the number of elements in the FIFO is greater than the threshold value. Thus, out of reset this register is set to detect one element in the respective FIFO since the initial value is zero.

NOTE

The threshold value should not exceed the FIFO size

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	TH3								TH2							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	TH1								TH0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c84625 4821 - Aaa 251 - 390d3ab624c1

Fields

Field	Function
31-24 TH3	FIFO 3 Threshold.
23-16 TH2	FIFO 2 Threshold.
15-8 TH1	FIFO 1 Threshold.
7-0 TH0	FIFO 0 Threshold.

57.6.17 FIFO Status Register (FST)

Offset

Register	Offset
FST	7Ch

Function

This register has the flags that indicate the status of the FIFO modules. Up to four FIFOs are supported. FIFO overrun, overflow, empty, and full flags are available. If enabled by the FCR register bits, these flags can generate an interrupt to the CPU. There is one interrupt signal per FIFO which combines overrun, overflow, empty, and full flags.

NOTE

The only actual error indication corresponds to the overrun flags which stay asserted once the overrun condition occurs. They are cleared by writing 1 to the respective flag. The overflow, empty, and full flags correspond to the current state of the FIFOs.

NOTE

Bits OR3, OR2, OR1, and OR0 can be read and cleared by software

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHA Automotive Technology Co., Ltd.
c8c462e5-481-4ea8-9251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OR3	OF3	EMP3	FULL3	OR2	OF2	EMP2	FULL2	OR1	OF1	EMP1	FULL1	OR0	OF0	EMP0	FULL0
W	W1C				W1C				W1C				W1C			
Reset	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0

Fields

Field	Function
31-16 —	Reserved
15 OR3	FIFO 3 Overrun interrupt flag. It is cleared if 1 is written to this bit location. 0b - Interrupt has not occurred 1b - Interrupt has occurred
14 OF3	FIFO 3 threshold Overflow interrupt flag. 0b - Interrupt has not occurred 1b - Interrupt has occurred
13 EMP3	FIFO 3 Empty interrupt flag. 0b - Interrupt has not occurred 1b - Interrupt has occurred
12 FULL3	FIFO 3 Full interrupt flag. 0b - Interrupt has not occurred 1b - Interrupt has occurred
11 OR2	FIFO 2 Overrun interrupt flag. It is cleared if 1 is written to this bit location. 0b - Interrupt has not occurred 1b - Interrupt has occurred
10 OF2	FIFO 2 threshold Overflow interrupt flag. 0b - Interrupt has not occurred 1b - Interrupt has occurred
9	FIFO 2 Empty interrupt flag.

Table continues on the next page...

Table continued from the previous page...

Field	Function
EMP2	0b - Interrupt has not occurred 1b - Interrupt has occurred
8 FULL2	FIFO 2 Full interrupt flag. 0b - Interrupt has not occurred 1b - Interrupt has occurred
7 OR1	FIFO 1 Overrun interrupt flag. It is cleared if 1 is written to this bit location. 0b - Interrupt has not occurred 1b - Interrupt has occurred
6 OF1	FIFO 1 threshold Overflow interrupt flag. 0b - Interrupt has not occurred 1b - Interrupt has occurred
5 EMP1	FIFO 1 Empty interrupt flag. 0b - Interrupt has not occurred 1b - Interrupt has occurred
4 FULL1	FULL1 FIFO 1 Full interrupt flag. 0b - Interrupt has not occurred 1b - Interrupt has occurred
3 OR0	FIFO 0 Overrun interrupt flag. It is cleared if 1 is written to this bit location. 0b - Interrupt has not occurred 1b - Interrupt has occurred
2 OF0	FIFO 0 threshold Overflow interrupt flag. 0b - Interrupt has not occurred 1b - Interrupt has occurred
1 EMP0	FIFO 0 Empty interrupt flag. 0b - Interrupt has not occurred 1b - Interrupt has occurred
0 FULL0	FIFO 0 Full interrupt flag. 0b - Interrupt has not occurred 1b - Interrupt has occurred

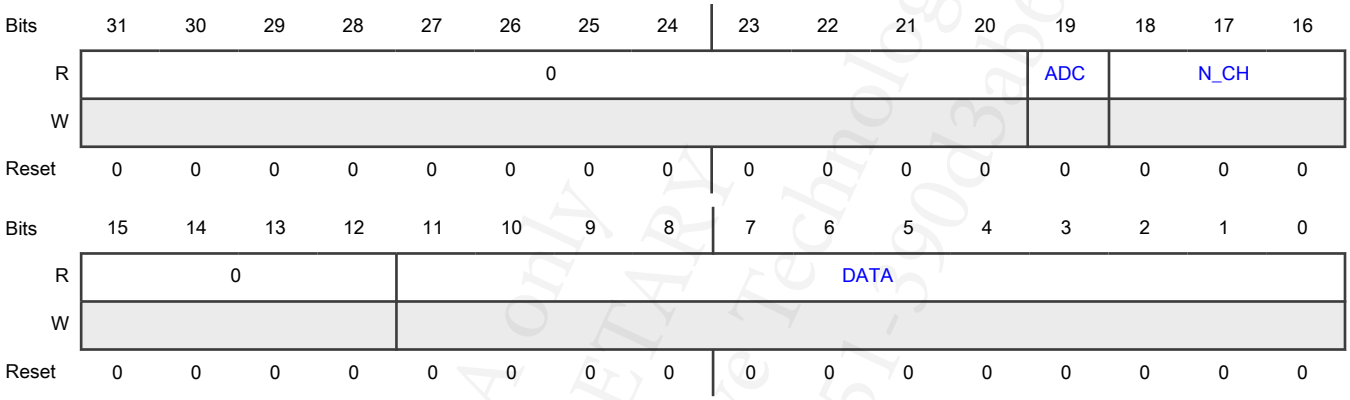
Provided by IHS DA only
CONTROLLED PROPRIETARY
Bosch CJ Automotive Technology Co., Ltd.
c0c46285-4821-4818-b251-390d3ab624c1

57.6.18 FIFO Right Aligned Data Register (FR0 - FR3)

Offset

Register	Offset
FR0	80h
FR1	84h
FR2	88h
FR3	8Ch

Diagram



Fields

Field	Function
31-20 —	Reserved
19 ADC	This bit indicates from which ADC Port the value in the DATA field corresponds to. 0b - Data coming from ADC Port B 1b - Data coming from ADC Port A
18-16 N_CH	Number of the channel that DATA field corresponds to.
15-12 —	Reserved
11-0 DATA	Data of stored channel.

Provided under NDA only
COMPANY PROPRIETARY
Beijing C.A.R. Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-59003ab624c1

57.6.19 FIFO Signed Left Aligned Data Register (FL0 - FL3)

Offset

Register	Offset
FL0	A0h
FL1	A4h
FL2	A8h
FL3	ACh

Function

NOTE

A read of this register consumes the data from the respective FIFO.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												ADC	N_CH		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	LA_DATA													0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-20 —	Reserved
19 ADC	This bit indicates from which ADC Port the value in the DATA field corresponds to. 0b - Data coming from ADC Port B 1b - Data coming from ADC Port A
18-16 N_CH	Number of the channel that DATA field corresponds to.
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-3 LA_DATA	LA_DATA Data of stored channel.
2-0 —	Reserved

57.6.20 Error Flag Register (EFR)

Offset

Register	Offset
EFR	C0h

Function

The EFR register bits are cleared if a 1 is written to the bit position. Otherwise, the bit retains the previous value.

NOTE

The ERRCMP field reads 0 at reset. When reset is released, this field remains cleared until the CTU timer clock is enabled. After the CTU timer clock is enabled, ERRCMP reads 1. This field must be cleared before enabling the interrupt requests.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		LIST_BE	CS	Reserved	ERRCMP	Reserved	Reserved	Reserved	Reserved	ADC_OE	TGS_OSM	MRS_O	ICE	SM_TO	MRS_RE
W			W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15-14 —	Reserved
13 LIST_BE	List Busy Error. Indicates that a list of commands is trying to send commands to a busy ADC or to the same ADC at the same time. It also indicates a conflict in the self-test algorithm execution by the ADCs, see Self-Test Execution Errors section. Cleared if 1 is written to this bit location. 0b - No error 1b - Error

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 CS	Self test Counter Status. This bit indicates if a self-test algorithm is in progress. If 1 is written to this bit, a reset to the self test counter will occur, thus terminating the self-test algorithm execution. As a status bit it indicates: 0b - All the counters are at reset value, thus none of the self-test algorithm are executing 1b - One of the counters used in self test mode is not in the reset state, thus self-test is in progress.
11 —	Reserved
10 ERRCMP	Error Compare. When a match occurs, the TGS Counter has reached the TGSCCR value. Cleared if 1 is written to this bit location. 0b - No match 1b - Match
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 ADC_OE	ADC command generation Overrun Error. This bit is set if an ADC command is issued when there is a command already being processed. Cleared if 1 is written to this bit location. 0b - No Error 1b - Error
4 TGS_OSM	The TGS Overrun in Sequential Mode. This bit is set when a new event occurs before the trigger event selected by the previous event had occurred. The error is generated only if the trigger from the TGS is linked to a SU output. Cleared if 1 is written to this bit location. 0b - No overrun 1b - Overrun
3 MRS_O	Master Reload Signal Overrun. If the time window defined by two consecutive MRS is not enough for the number of programmed events in the THCR registers, an error occurs and MRS_O flag is set. Cleared if 1 is written to this bit location. 0b - No overrun

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Overrun
2 ICE	Invalid Command Error. An error that occurs when the shared channel is accessed by two ADCs at the same time, or when an ADC is already using the channel. This situation can occur with a dual conversion command or with two single conversions in parallel mode. 0b - No error 1b - Error
1 SM_TO	Trigger Overrun (more than 8 ES) in TGS Sequential Mode. If the number of positive edges of ES is higher than 8, an error occurs and SM_TO flag is set. Cleared if 1 is written to this bit location. 0b - No overrun 1b - Overrun
0 MRS_RE	Master Reload Signal Reload Error. If an MRS occurs while the user is updating a double-buffered register, the MRS_RE is set indicating an error condition. Cleared if 1 is written to this bit location. 0b - No error 1b - Error

57.6.21 Interrupt Flag Register (IFR)

Offset

Register	Offset
IFR	C2h

Function

The IFR register bits are cleared if a 1 is written to the bit position. Otherwise, the bit retains the previous value.

NOTE

The Tn_I fields read 0 at reset. When reset is released, these fields remain cleared until the CTU timer clock is enabled. After the CTU timer clock is enabled, the Tn_I fields read 1. These flags must be cleared before enabling the interrupt requests.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4c1-48a8-b251-390...ab64c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				SERR_B	SERR_A	ADC_I	T7_I	T6_I	T5_I	T4_I	T3_I	T2_I	T1_I	T0_I	MRS_I
W					W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0

Fields

Field	Function
15-12 —	Reserved
11 SERR_B	If this bit is set, it means that the time between the start of a conversion and the end of that conversion is out of the expected range which is defined by the EXPBR and CNTRNGR registers. See Expected Value B (EXPBR) and Counter Range (CNTRNGR) registers. Cleared if 1 is written to this bit location.
10 SERR_A	If this bit is set, it means that the time between the start of a conversion and the end of that conversion is out of the expected range defined by the EXPAR and CNTRNGR registers. See the Expected value A (EXPAR) and Counter Range (CNTRNGR) registers. Cleared if 1 is written to this bit location.
9 ADC_I	ADC command interrupt flag is set when a new command is issued. The interrupt related to this flag is enabled by the CIR bit in the CLR_x_n registers. Cleared if 1 is written to this bit location.
8 T7_I	Trigger 7 interrupt flag is set when the corresponding trigger is issued. Cleared if 1 is written to this bit location.
7 T6_I	Trigger 6 interrupt flag is set when the corresponding trigger is issued. Cleared if 1 is written to this bit location.
6 T5_I	Trigger 5 interrupt flag is set when the corresponding trigger is issued. Cleared if 1 is written to this bit location.
5 T4_I	Trigger 4 interrupt flag is set when the corresponding trigger is issued. Cleared if 1 is written to this bit location.
4 T3_I	Trigger 3 interrupt flag is set when the corresponding trigger is issued. Cleared if 1 is written to this bit location.
3 T2_I	Trigger 2 interrupt flag is set when the corresponding trigger is issued. Cleared if 1 is written to this bit location.
2 T1_I	Trigger 1 interrupt flag is set when the corresponding trigger is issued. Cleared if 1 is written to this bit location.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 T0_I	Trigger 0 interrupt flag is set when the corresponding trigger is issued. Cleared if 1 is written to this bit location.
0 MRS_I	MRS Interrupt flag is set when the Master Reload Signal occurs. The interrupt associated with this flag is enabled by the MRS_IE bit in the IR register. Cleared if 1 is written to this bit location.

57.6.22 Interrupt/DMA Register (IR)

Offset

Register	Offset
IR	C4h

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	T7_IE	T6_IE	T5_IE	T4_IE	T3_IE	T2_IE	T1_IE	T0_IE	0		SAF_CNT...	SAF_CNT...	DMA_DE	MRS_DMAE	MRS_IE	IEE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 T7_IE	Trigger 7 Interrupt Enable 0b - Disabled 1b - Enabled
14 T6_IE	Trigger 6 Interrupt Enable 0b - Disabled 1b - Enabled
13 T5_IE	Trigger 5 Interrupt Enable 0b - Disabled 1b - Enabled
12 T4_IE	Trigger 4 Interrupt Enable 0b - Disabled 1b - Enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
11 T3_IE	Trigger 3 Interrupt Enable 0b - Disabled 1b - Enabled
10 T2_IE	Trigger 2 Interrupt Enable 0b - Disabled 1b - Enabled
9 T1_IE	Trigger 1 Interrupt Enable 0b - Disabled 1b - Enabled
8 T0_IE	Trigger 0 Interrupt Enable 0b - Disabled 1b - Enabled
7-6 —	Reserved
5 SAF_CNT_B_EN	Enable the ADC Port B counter to check the conversion time. 0b - Disabled 1b - Enabled
4 SAF_CNT_A_EN	Enable the ADC Port A counter to check the conversion time. 0b - Disabled 1b - Enabled
3 DMA_DE	If this bit is set, a dma done acts as a write '1' in the GRE bit.
2 MRS_DMAE	If GRE bit is set, DMA request is issued on MRS occurrence.
1 MRS_IE	MRS Interrupt Enable. 0b - Disabled 1b - Enabled
0 IEE	Interrupt Error Enable. 0b - Disabled 1b - Enabled

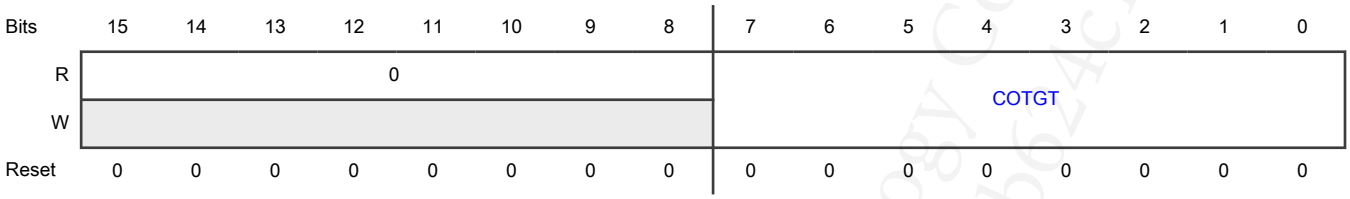
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

57.6.23 Control ON Time Register (COTR)

Offset

Register	Offset
COTR	C6h

Diagram



Fields

Field	Function
15-8 —	Reserved
7-0 COTGT	COTGT Control ON-Time and Guard Time for external trigger. See Scheduler subunit (SU) for a more detailed description.

57.6.24 Control Register (CR)

Offset

Register	Offset
CR	C8h

Function

NOTE

The CTU_ADC_R bit resets all lists in execution. If CTU list execution is in parallel mode, this bit resets both lists even if they issued commands to different ADCs. Also, the self-test algorithm counters go to reset state.

NOTE

Bits Tn_SG, CTU_ADC_R, CGRE, and MRS_SG can be set by software. Bits CTU_ODIS, DFE, GRE, and TGSISR_RE can be read and set by software.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e3-4821-48a8-b251-390d3ab624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									CTU_ADC...	CTU_ODIS	DFE	CGRE	FGRE	MRS_SG	GRE	TGSISR_...
W	T7_SG	T6_SG	T5_SG	T4_SG	T3_SG	T2_SG	T1_SG	T0_SG								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
15 T7_SG	Trigger 7 Software Generated. 0b - S/W not generated 1b - S/W generated
14 T6_SG	Trigger 6 Software Generated. 0b - S/W not generated 1b - S/W generated
13 T5_SG	Trigger 5 Software Generated. 0b - S/W not generated 1b - S/W generated
12 T4_SG	Trigger 4 Software Generated. 0b - S/W not generated 1b - S/W generated
11 T3_SG	Trigger 3 Software Generated. 0b - S/W not generated 1b - S/W generated
10 T2_SG	Trigger 2 Software Generated. 0b - S/W not generated 1b - S/W generated
9 T1_SG	Trigger 1 Software Generated. 0b - S/W not generated 1b - S/W generated
8 T0_SG	Trigger 0 Software Generated. 0b - S/W not generated 1b - S/W generated

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 CTU_ADC_R	CTU command list control state machine Reset. Self negated bit. This bit resets the command list in the CTU in case the ADC does not respond to a conversion command within the expected time window. After a reset, the command list stops execution and waits for a new trigger event in order to initiate a command list execution.
6 CTU_ODIS	CTU Output Disable. 0b - Enabled 1b - Disabled
5 DFE	Writing a 1b to this bit creates a trigger in CTU timer clock domain which is used to register the value of Digital Filter in the register CTU_DFR into CTU timer clock domain. This bit is self-clearing. It is cleared by CTU when its value it registered into CTU timer clock domain. Digital Filter Enable. 0b - Disabled 1b - Enabled
4 CGRE	Clear GRE to 0.
3 FGRE	Flag GRE. See Reload mechanism for more details.
2 MRS_SG	MRS Software Generated. 0b - S/W not generated 1b - S/W generated
1 GRE	General Reload Enable. See Reload mechanism for more details. 0b - Disabled 1b - Enabled
0 TGSISR_RE	TGS Input Selection Register Reload Enable. Controls the re-load of the TGSISR register. This bit is self negated, thus it always returns zero. See Trigger Generator Subunit Input Selection Register (TGSISR) for more details.

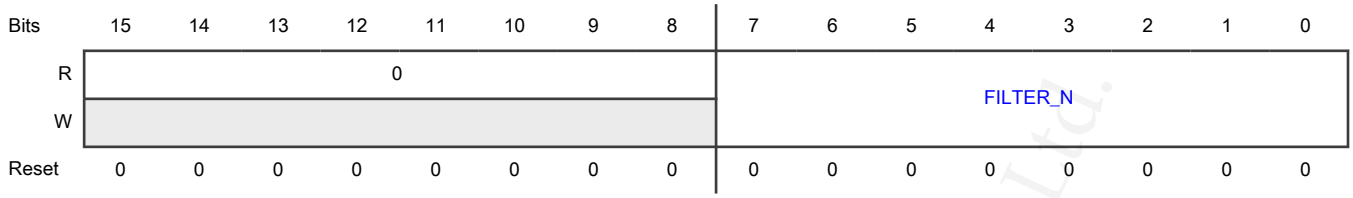
57.6.25 Digital Filter Register (DFR)

Offset

Register	Offset
DFR	CAh

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4a21-4a8-b251-390d3ab24c1

Diagram



Fields

Field	Function
15-8 —	Reserved
7-0 FILTER_N	Digital Filter value. The external signal (EXT_IN) is considered at 1 if it is latched FILTER_N times at 1 and is considered at 0 if it is latched FILTER_N times at 0. If FILTER_N = 0, the filter is in bypass. 0d - Filter is bypassed (any trigger on EXT_IN if captured by synchronizer is passed) 1d - External signal must be stable at least 1 cycle. 2d - External signal must be stable at least 2 cycles. Nd - External signal must be stable at least N cycles.

57.6.26 Expected Value A Register (EXPAR)

Offset

Register	Offset
EXPAR	CCh

Function

The EXPAR register has the expected number of system clock cycles (the slave bus clock) for an AD conversion to be completed by the ADC connected to Port A. If SAF_CNT_A_EN is set in the IR register and a conversion takes a number of cycles out of the interval defined by EXPA and CNTRNG to complete, then the SERR_A error bit is set in the IFR register. Please see the CNTRNGR for more details.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4824-48a8-b251-390a3b624c1

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EXPA															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
15-0 EXPA	This value is the expected number of system clock cycles needed by ADC Port A to complete a conversion.

57.6.27 Expected Value B Register (EXPBR)

Offset

Register	Offset
EXPBR	CEh

Function

The EXPBR register has the expected number of system clock cycles (the slave bus clock) for an AD conversion to be completed by the ADC connected to Port B. If SAF_CNT_B_EN is set in the IR register and a conversion takes a number of cycles out of the interval defined by EXPB and CNTRNG to complete, then the SERR_B error bit is set in the IFR register. Please see the CNTRNGR for more details.

Diagram

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EXPB															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
15-0 EXPB	This value is the expected number of system clock cycles needed by ADC Port B to complete a conversion.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-b51-390d3ab624c1

57.6.28 Counter Range Register (CNTRNGR)

Offset

Register	Offset
CNTRNGR	D0h

Function

The CNTRNGR register provides a way to mask out the less significant bits in the EXP_A and EXP_B fields of EXPAR and EXPBR registers, respectively. If set, bits in CNTRNG mask out related bits in EXP_A and EXP_B fields, thus providing a range in which a conversion should occur. The expression used for the conversion range check is: set SERR_A bit if (conversion counter & ~CNTRNG > EXP_A & ~CNTRNG) or (conversion counter & ~CNTRNG < EXPP_A & ~CNTRNG), where the conversion counter increments at system clock rate (slave bus clock rate). This equation also applies to SERR_B and EXP_B.

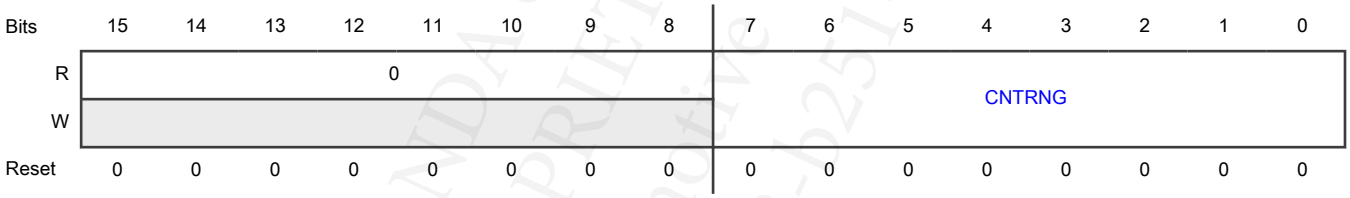
NOTE

The 1s in the CNTRNG must be set from the right to the left (less significant bits) without gap with 0.

NOTE

The safety counter allowed range is obtained by taking the EXPA/B value and exchanging to "X" the corresponding positions where the CNTRNG bit is 1.

Diagram



Fields

Field	Function
15-8 —	Reserved
7-0 CNTRNG	CNTRNG If set, the bits in this field mask out the equivalent bits of the expected values EXP_A and EXP_B. In this way, a range of system clock (slave bus clock) cycles is defined for a conversion to occur. This means the bit positions with 1s in the CNTRNG change the corresponding bits on EXPA/EXPB and safety counters A/B to "X". For example, with EXPA = 1b0010_1101 and CNTRNG = 0b0111, the safety counter can end with values of the format 0b0010_1XXX to avoid SERR_A error flag.

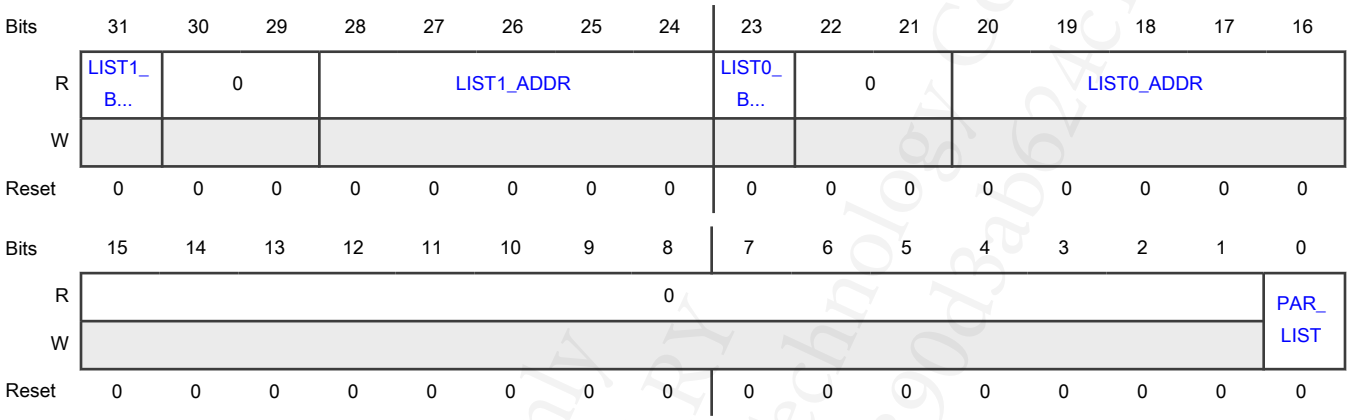
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46f2c-4821-48a8-b251-390d3bbc0c01

57.6.29 List Control/Status Register (LISTCSR)

Offset

Register	Offset
LISTCSR	D4h

Diagram



Fields

Field	Function
31 LIST1_BLK	List 1 Blocked flag. When set, this bit indicates that the address indicated by LIST1_ADDR had failed when trying to issue a command to the corresponding ADC. This bit is only meaningful if LIST_BE flag in EFR register is set. This bit is cleared when LIST_BE flag is cleared.
30-29 —	Reserved
28-24 LIST1_ADDR	List Address 1. This field shows the command address that was being executed when the LIST_BE flag in the EFR register was set.
23 LIST0_BLK	List 0 Blocked flag. This bit indicates that the address indicated by LIST0_ADDR had failed when trying to issue a command to the corresponding ADC. This bit is only meaningful if LIST_BE flag in EFR register is set. This bit is cleared when LIST_BE flag is cleared.
22-21 —	Reserved
20-16 LIST0_ADDR	List Address 0. Indicates the command address being executed when LIST_BE flag in EFR register was set.
15-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 PAR_LIST	Parallel mode List. This bit selects the parallel mode for the list execution. 0b - List is executed in streaming mode 1b - List is executed in parallel mode

57.7 Functional description

The CTU is consists of two subunits: the Trigger Generator subunit, or TGS, and the Scheduler subunit, or SU. The TGS is responsible for generating triggers based on the module input signals connected to timer modules. The trigger signal from TGS is then connected to the scheduler subunit, which is responsible for the generation of ADC commands or triggers to on-chip logic such as timers. The following sections describe the TGS and SU in more detail.

57.7.1 Trigger Generator Subunit (TGS)

The Trigger Generator Subunit, TGS, generates up to eight trigger events. The main sub-blocks in the TGS are:

- Counter: generates sequential trigger events
- Trigger list: with 8 x 16-bit double-buffered registers for timing comparison

The TGS has two modes of operation:

- **Triggered mode**: each event source generates up to eight trigger event outputs.
- **Sequential mode**: each event source generates only one trigger event output.

NOTE

An *event* means a pulse generated from the selected CTU trigger inputs. It is represented in Figure 514 by the ES signal (event signal).

Trigger events can be initiated by hardware or by software. A software trigger is generated by writing to the CTU Control Register, CR. The CR register Tx_SG bits are automatically cleared by the CTU hardware when the respective trigger event is generated.

57.7.2 TGS in triggered mode

See Figure 512 that shows the TGS configuration in triggered mode.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a6-b251-390d1ab6...

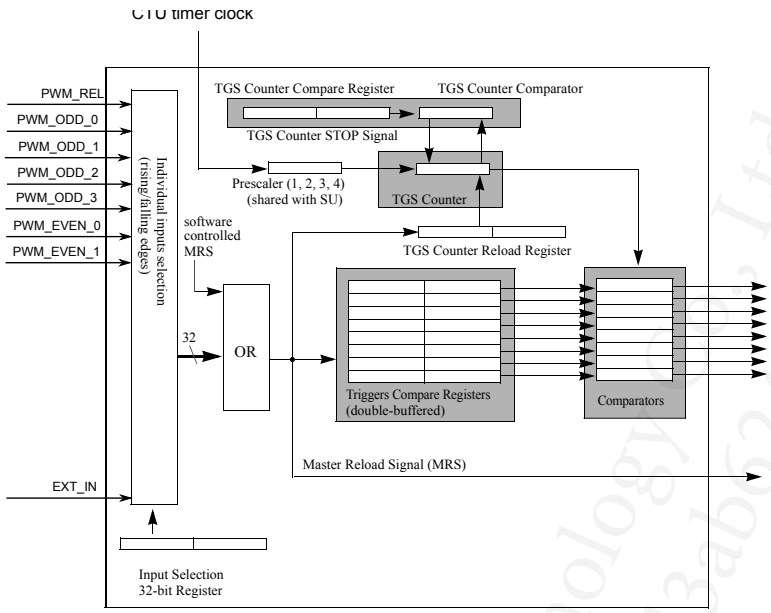


Figure 512. Trigger Generator subunit in the triggered mode

NOTE

The CTU timer clock should be connected to the same clock source used by the CTU inputs clock. See Figure 512 that shows these inputs are from the FTM generator, and EXT_IN.

The TGS has 16 input signals selected from the input selection register (TGSISR). The available selections are these:

- Inactive, no selection
- Rising edge
- Falling edge
- Both edges

Up to 2 x 16 input events corresponding to 16 input signals are selected through the TGSISR register and OR-ed in order to generate the Master Reload Signal (MRS), which defines a *control cycle*.

The MRS can also be generated by software. The software operation is done by writing to the MRS_SG bit in the CTU Control Register. The MRS_SG bit is self cleared; thus, it reads always as 0. The MRS is used to re-load the TGS Counter Register with the values from the double-buffered TGSCRR register. The MRS is also used to re-load all double-buffered registers, such as the trigger compare registers, TGSCR and TGSCRR, with the corresponding buffered values.

The triggers list registers consist of eight compare registers (T0CR through T7CR). Each register is associated with a comparator, where a match generates an output trigger. On the occurrence of a re-load triggered by MRS, the comparison for trigger events generation starts from the first cycle of counting. However, the comparison with the TGS Counter Compare Register (TGSCCR) is always disabled during one TGS clock cycle and then re-enabled. A match between the TGSCCR and TGS counter stops the TGS counter. Therefore, this one cycle disable allows the user to configure the TGSCCR and TGSCRR with the same value.

The prescaler for TGS and SU is defined by the PRES bits in the TGS Control Register and divides the CTU timer clock. The available selections for the prescaler values are 1, 2, 3, and 4.

Figure 513 shows a timing diagram for the TGS operation in the triggered mode. The MRS and the trigger event occurrences are indicated along with the delay relative to the MRS occurrence.

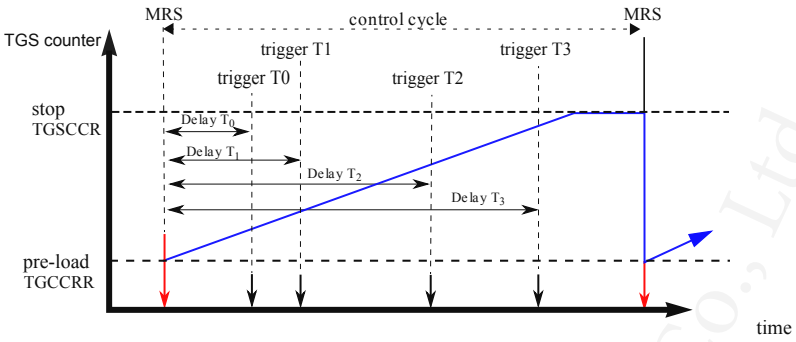


Figure 513. Example timing diagram for TGS in triggered mode

57.7.3 TGS in sequential mode

Figure 514 shows the blocks and signals related to TGS in sequential mode.

In sequential mode, only one of the 2 x 16 input signals is selected by the 5-bit MRS_SM (master reload selection). The selected signal re-loads the trigger list and resets the 3-bit ES counter which selects the trigger event. Sequences of up to eight trigger events are generated in one control cycle.

The 2 x 16 input events corresponding to 1'd0 input signals with 2 edges detection per signal are individually enabled through the register TGSISR and OR-ed in order to generate the event signal ES. This signal is used to re-load the TGS counter register and to increment the 3-bit ES counter.

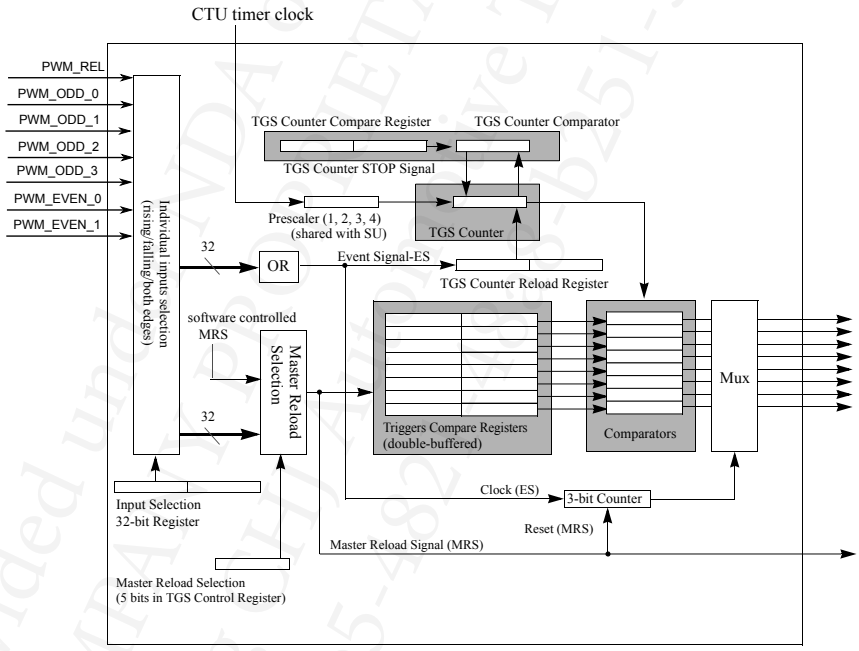


Figure 514. TGS in sequential mode

Figure 515 is a timing diagram for TGS in sequential mode showing the MRS and ES. The trigger events are indicated by a delay with respect to ES. Note that the ES and MRS are aligned in the beginning. The TGS counter is re-loaded on each ES and starts counting up until the next ES or until it matches the value in the TGSCCR register and stops.

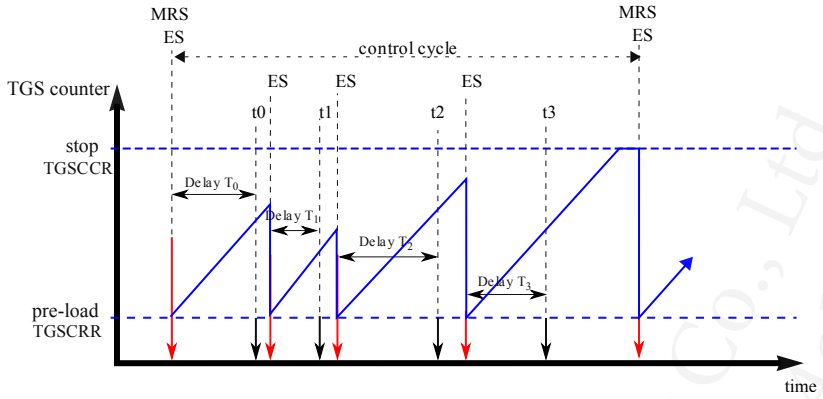


Figure 515. Example timing diagram for TGS in sequential mode

57.7.4 TGS counter

The TGS counter counts from negative to positive values as defined by the re-load and stop values defined in register TGSCRR and TGSCCR, respectively. The maximum counter value is 0x7FFF; after that, a counter wrap occurs and the counter value transitions from 0x7FFF to 0x8000. The following figure shows TGS counting examples.

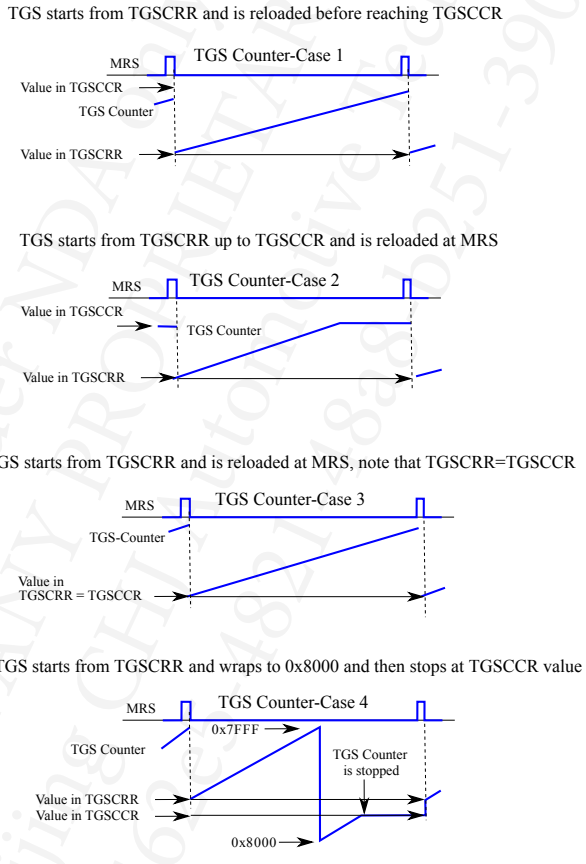


Figure 516. TGS counter behavior

NOTE

The TGS counter starts counting after an MRS and only stops if a counter wrap occurs and the TGSCCR value is reached before the next MRS occurs.

Loading the initial value of TGS counter in Triggered mode

TGSCRR is a double-buffered register. It has a CPU copy (TGSCRR) and a buffered copy (B_TGSCRR) for internal operation. The TGS counter and B_TGSCRR are loaded on the MRS pulse at the same clock cycle. So, the TGS counter is loaded with the existing value of B_TGSCRR before it is updated from TGSCRR. For this reason, the modified value of TGSCRR takes effect after one control cycle.

Following is a timing diagram of the TGSCRR and TGSCCR register updates. The CPU updates the TGSCRR and TGSCCR during the Nth control cycle (1), and then their buffer copies are updated with new values on the next MRS pulse (2). Because the TGS counter is loaded on the same MRS (3), it is reloaded with B_TGSCRR on the second MRS pulse (4) after the CPU writes the new values into the TGSCRR. So, the effect of TGSCRR takes place at N+2 control cycle (4) whereas the effect of TGSCCR takes place in N+1 control cycle (5).

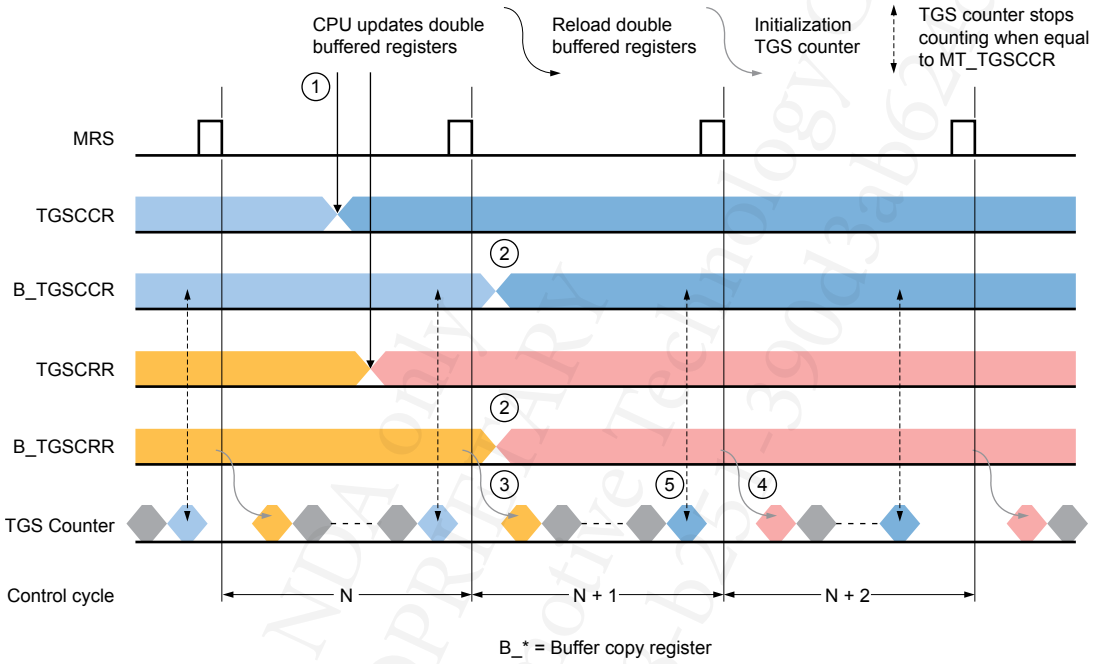


Figure 517. Update TGSCRR in the Triggered mode

NOTE

So, to conclude, the initial value of TGS counter, TGSCRR, must be set one control period earlier than the rest of the double-buffered registers.

Loading the initial value of TGS counter in Sequential mode:

In this mode, the TGS counter is loaded with B_ TGSCRR on every ES pulse, unlike in the Triggered mode. The MRS pulse always coincides with any one of the ES pulses, so they both occur on same clock cycle. For this reason, the TGS counter is loaded with the actual value of B_TGSCRR before it is updated from TGSCRR. So, the modified value of TGSCRR takes effect on the second ES pulse of the next control cycle with respect to the write to TGSCRR by CPU.

Following is the timing diagram of the TGSCRR and TGSCCR register updates. The CPU updates the TGSCRR and TGSCCR during the Nth control cycle (1), then their buffer copies are updated with new values on the next MRS pulse (2). Because the TGS counter is loaded on the same MRS (3), it is reloaded with updated B_TGSCRR on the second ES pulse (4) at N+1 cycle after the CPU writes the new values into the TGSCRR. So, the effect of TGSCRR takes place at N+1 control cycle (second ES pulse onwards (4)) whereas the effect of TGSCCR takes place in N+1 control cycle (first ES pulse onwards (5)).

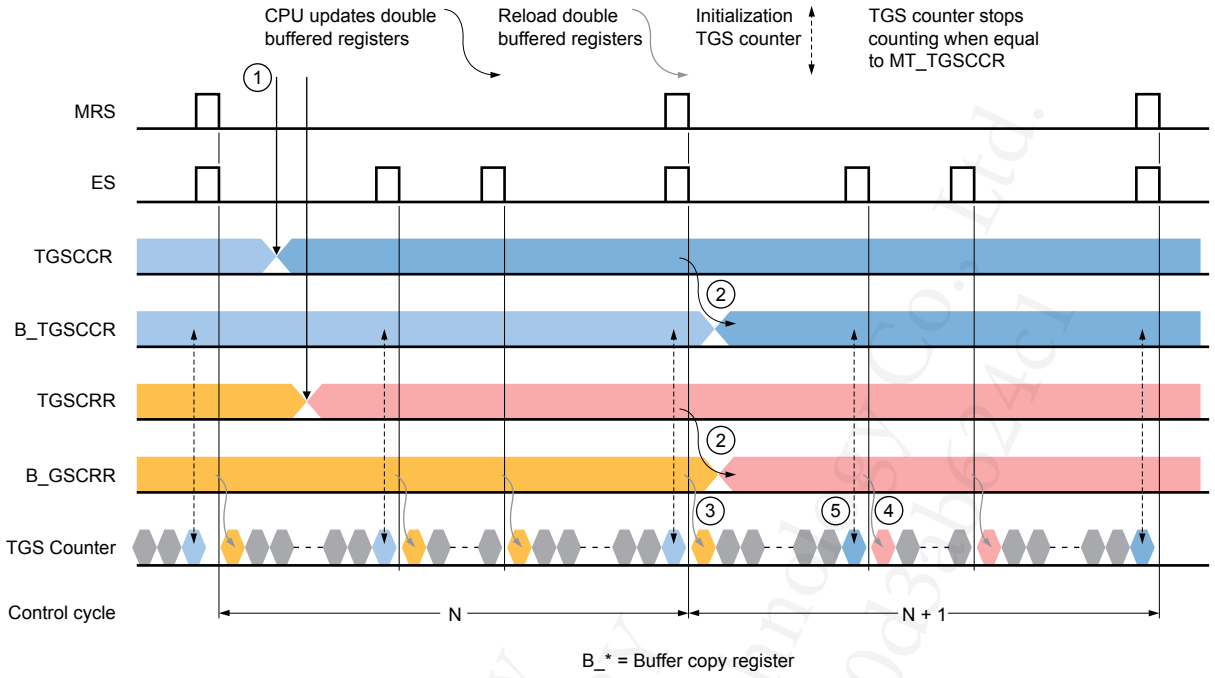


Figure 518. Update TGSCRR in the Sequential mode

57.7.5 Scheduler subunit (SU)

The following figure shows the block diagram of the Scheduler subunit. The SU trigger event outputs are these:

- ADC command or ADC stream of commands

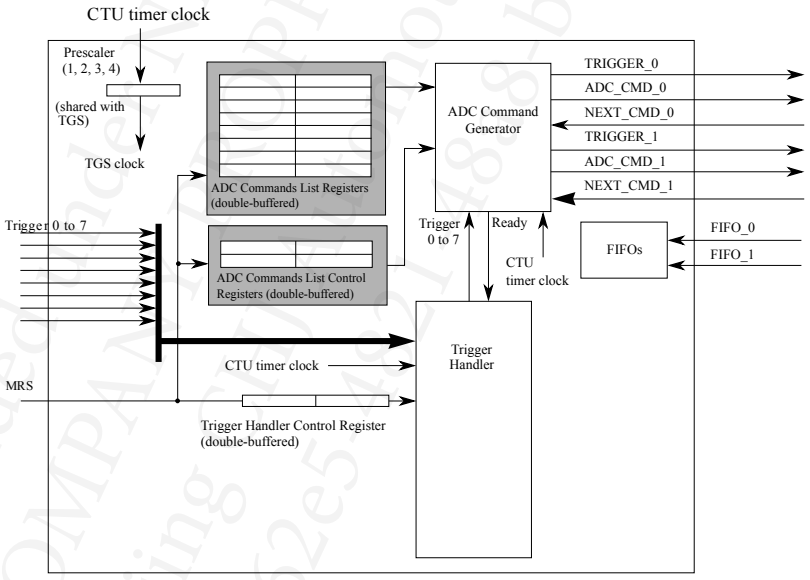


Figure 519. Scheduler subunit

The SU receives eight trigger signals from the Trigger Generator subunit(TGS) and starts a command list to the selected ADC or generates the trigger event outputs. This module has the same functionality in both TGS modes: triggered and sequential. Each of the 2 SU outputs can be linked to any of the eight trigger events from TGS. This is implemented by the Trigger Handler block. Each trigger event can be linked to one or more SU outputs.

If two events are linked at the same time to the same output (either ADC commands or output triggers), an error is generated and flagged in the EFR register's TnOE bits. In this case, only the trigger with the lowest index is used. For example, if trigger 0 and trigger 1 are linked to the same ADC output and they occur at the same time, an error is generated and the ADC command corresponding to trigger 0 is generated.

When a trigger is linked to an ADC, an associated ADC stream of commands is generated. The ADC Commands List Control Register CLCR1/2 defines the list entry point that is the first command to be executed. Subsequent list pointers are generated in an incremental way as the commands in the list are executed.

57.7.6 ADC commands list

The SU implements a command list that stores the ADC commands. The command list stores up to 24 x 16-bit commands (see ADC command list format) in a double-buffered implementation. The command list buffer registers may be updated at any time between two consecutive MRs, but the changes are transferred from the buffers to the actual registers only after an MRS occurs, so a correct reload procedure is performed (see Reload mechanism). The first command in a list is pointed by the 5-bit CLCRx register. After a command list is triggered, it executes until the last command is found. The last command (n) is defined by the LC bit of the next command (n+1) in the list. Note that the command with LC=1 does not belong to the list and thus it is not executed. Also note that the LC bit is ignored in the first command of a list. See the command list registers, CLR_A_n, CLR_B_n, or CLR_C_n, where n = 1,...,24. The figure below describes the state diagram for the Commands List execution. A sequence of commands on the list is executed until the last command is found. See ADC command list format for more details.

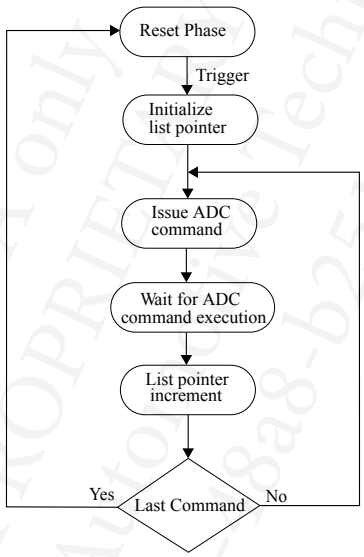


Figure 520. ADC command list scanning state diagram

NOTE

The CTU reads the next command line (the LC bit) to determine if the present command is the last one to be executed.

57.7.7 ADC command list format

The ADC command can be configured for single or dual conversion. In single conversion mode, the command targets one ADC only. In dual conversion mode, the conversion command is sent to both ADCs at the same time. Due to the clock domain crossing between CTU and ADCs, there could be a misalignment of up to one ADC clock cycle between the two commands.

The result of each conversion, in both modes, is stored in one of the four available FIFOs. In dual conversion mode, both ADCs store the result of their conversion in the same FIFO. If both ADCs access the FIFO in the same clock cycle, ADC unit A has higher priority, otherwise the first available conversion result is stored first.

The CTU supports up to 2 x 16 channels for both ADCs. The same physical channel cannot be assigned for both ADCs in a dual conversion mode. If there is an attempt to do so, an error is flagged in the EFR register's ICE bit, but the CTU sends the command to both ADCs.

In general, an ADC command consist of the following fields depending upon the conversion format used (see Commands List Register A for ADC single-conversion mode commands (CLR_A_1 - CLR_A_24)):

- Channel A: ADC A channel number (4 bits)
- Channel B: ADC B channel number (4 bits)
- Target ADC selection: ADC A or ADC B used in single conversion mode only (1 bit)
- FIFO selection bits for the selected ADC unit: up to four FIFOs (2 bits)
- Conversion mode selection: single or dual conversion mode (1 bit)
- Last command (LC): defines the last command in a list (1 bit)
- Interrupt request: enable interrupt request on command execution (1 bit)

The first command of a list is indicated by the CLCR1/2 registers' Tx_INDEX field. The last command is indicated by the last command bit(LC) of the following command in the command list. The first command is always executed, and the execution of the following commands is controlled by the LC bit.

The figure below describes several scenarios for the command sequencing. If the first command indicated by the CLCR1/2 register is one of the lines identified as "Last command," then the list is a single command list. Also note that the list pointer wraps back to the first command in the list after command 24 is executed and if it is not the last command. If no command is defined as the last command, then the list is executed continuously in a loop of 24 commands.

- Command List A: two command queues are available with elements 4 and 24 being last commands
- Command List B: sub-list 1 with one single command at address 1, and sub-list 2 with a queue of commands with last command at address 24
- Command List C: sub-list 2 with one single command at address 24, and sub-list 1 with a command queue with last command at address 23
- Command List D: two command queues with a wrap at address 24

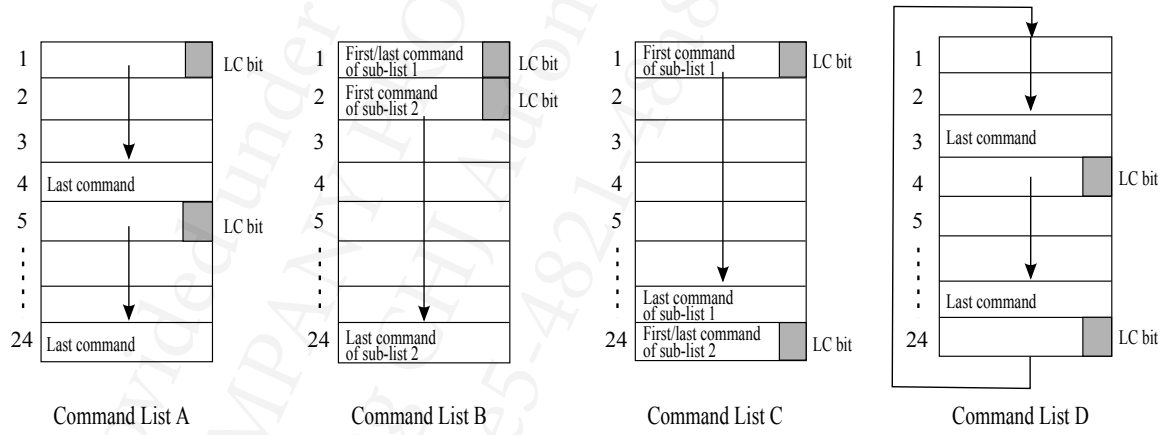


Figure 521. Command list sequencing

NOTE

The addresses shown in in the above figure do not represent addresses in the memory map. Please refer to the Memory Map section to check for the actual addresses.

57.7.8 ADC command list operation modes

There are two modes of operation in the ADC command list execution: streaming mode and parallel mode. In the streaming mode, the command lists should behave as a stream of commands, indicating that two or more lists cannot be executed at the same time, but only in a sequence. Thus, if list 1 is triggered, it should finish the execution of its last command before list 2 can be triggered. If there is an attempt to trigger more than one list at the same time, then an error occurs.

In the parallel mode, up to two lists can be executed at the same time. To avoid errors during the execution of two parallel lists, they should not have commands for both ADCs. Therefore, if list 1 has commands for ADC A, then list 2 should only have commands for ADC B. Also note that if a shared analog input is used by one of the lists, it should not be used by the other list to avoid a contention and therefore an error. ADC dual conversion commands should also not be used in the parallel mode.

Figure 522 shows the execution of two lists in streaming mode. While a list is being executed, any attempt to trigger the execution of another list while in this mode will cause an error. Note that both lists may have commands for ADC A or B with no restriction. Also, dual commands may be used.

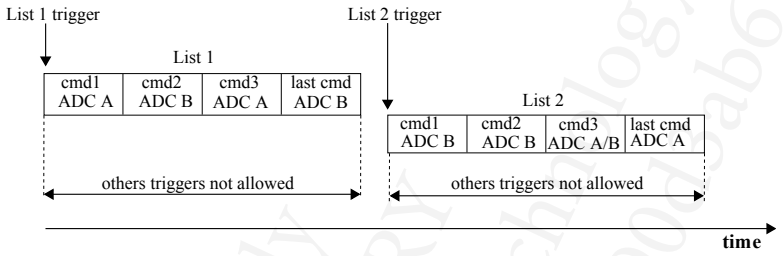


Figure 522. List execution in Streaming mode

Figure 523 shows the execution of two lists in the parallel mode. There are no restrictions for the triggers of both lists. After the two lists are triggered, no other triggers are allowed. If there is an attempt to trigger a third list while lists 1 and 2 are executing, then an error is issued by setting the ADC_OE bit in the EFR register. Note that list 1 only has commands for ADC A and list 2 only has commands for ADC B. Having ADC A and ADC B commands in the same list may cause an error because two commands for the same ADC cannot be executed from different lists at the same time.

NOTE

The mixing of commands for both ADCs in the same list in the parallel mode must not be used because the correct execution of commands in this case is not guaranteed by the CTU hardware.

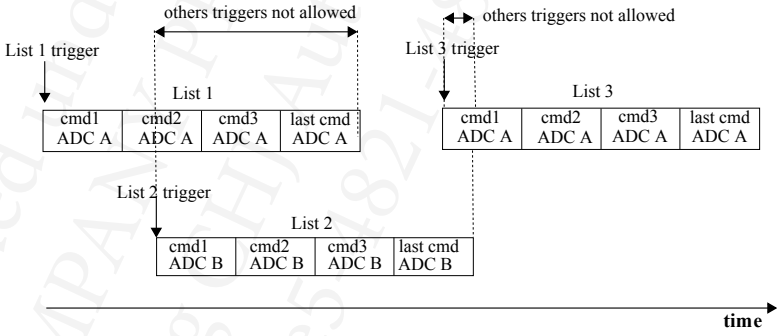


Figure 523. List execution in parallel mode

The list mode is controlled by the PAR_LIST bit in the LISTCSR register.

57.7.9 ADC results FIFOs

ADC results are stored in one of the four FIFOs. These FIFOs allow the storage of ADC results according to the type of data acquisition (phase currents, rotor position, ground-noise, and so on) so the software can distinguish data from several ADC channels. Each FIFO has its own interrupt line, DMA request signal, and a status register. The target FIFO for a conversion result

is specified in the ADC command. Each FIFO element is 32-bit wide. The following figure shows the FIFO selection for ADC A and ADC B results.

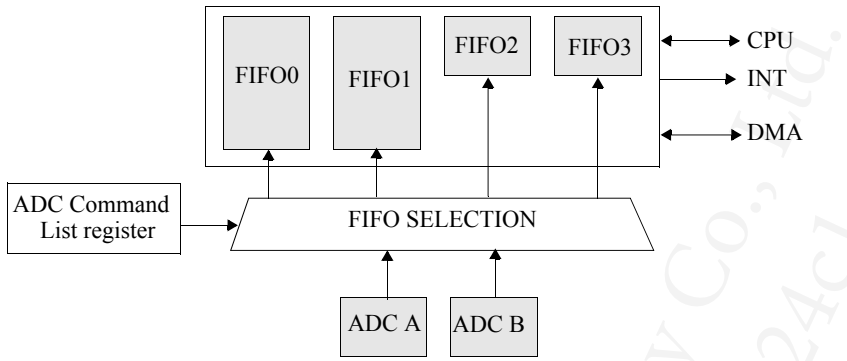


Figure 524. FIFO selection for ADC results storage

CTU has 4 FIFOs. The depth of each FIFO is fixed as given below:

- FIFO 0 : depth is 8.
- FIFO 1 : depth is 8.
- FIFO 2 : depth is 8.
- FIFO 3 : depth is 8.

FIFO access can be performed by 16-bit or 32-bit read transactions. If a 32-bit reads the conversion result, the target ADC and the channel number are accessed. An 8-bit access must not be used; otherwise, the FIFO pointer increments and the complete data is not retrieved. The FIFO result register can be read in a right- or left-aligned format using two different addresses:

- Unsigned right-justified, read from register FLx (for FIFO x)

  Conversion result is unsigned right-justified data, that is bits [11:0] are used for a 12-bit resolution and bits [15:12] always return zero

- Signed left-justified, read from register FLx (for FIFO x)

  Conversion result is signed left-justified data, i.e. bit [15] is reserved for sign and is always read as zero, bits [14:3] are used for 12-bit resolution and bits [2:0] always return zero

The FIFO access should be based on the status bits in the FCR register. Empty and full indications are provided. If data is read from an empty FIFO, then the read data should not be considered as valid data. Note that there is no indication of an underflow condition. If the FIFO is full and there is an attempt to write more data, then the new data is not written to the FIFO and an overrun flag is set.

57.7.10 Reload mechanism

To assure coherent programming, several CTU registers are double-buffered, thus allowing the programming of new data on the buffers while not affecting the data being used. All the registers are uploaded with buffer data at the same time. For the majority of the CTU registers, the re-load is controlled by the Master Reload Signal(MRS).

As the Master Reload Signal is generated by the hardware, it may occur while the software is still updating the buffer registers. In this case, non-coherent values are written to the registers because the CPU did not finish the programing of all registers. To avoid this situation, a General Reload Enable(GRE), control bit is provided. If GRE is cleared, then no reload occurs. If this bit is set, then the reload is performed when MRS occurs. The GRE bit is automatically cleared by MRS or is cleared by software using the CGRE bit in the same register. The CGRE bit is self-cleared and thus always read as zero.

The software should be able to program all registers buffers within a control cycle. To help the software to evaluate that a flawless register programing was executed, the MRS_RE flag is provided in the EFR register. If cleared, this flag indicates that all register buffers were programed before MRS occurred, as follows:

- If any of the double buffered registers is written, the FGRE flag is set indicating that a programing cycle has initiated.
- Set GRE to indicate that all updates to the buffer registers were performed.
- The reload of all registers is executed when the MRS occurs. Because GRE is set, MRS_RE is not be set.
- GRE and FGRE are cleared when MRS occurs.

See Figure 525 for more details.

An error is flagged by MRS_RE when MRS occurs, GRE is cleared, and FGRE is set. This scenario indicates that the software has initiated a register programming but has not finished it before the MRS, as follows:

- If any of the double buffered registers is written, the FGRE flag is set indicating that a programing cycle has initiated.
- MRS occurs and GRE is not set. In this case, MRS_RE is set and, if enabled, an interrupt is generated.
- FGRE is not cleared when MRS occurs and no register is updated, because GRE is cleared.

Note that the software does not need to update all double-buffered registers during a *control cycle*. To avoid the error indication, it is sufficient to set GRE before an MRS occurs.

A race condition occurs when GRE is being cleared by CGRE at the same time that an MRS occurs. In this case, GRE is considered to be cleared and no reload occurs. Another race condition may occur between CGRE and GRE being set. Because these two bits are in the same register, they could eventually be set at the same time. In this case, CGRE has precedence; thus, GRE is cleared.

NOTE

MRS has priority over the TGSISR_RE bit in the CR register, which is the re-synchronization bit of the TGSISR.

The following figure describes two scenarios for the register re-load. Scenario (a) is a normal re-load where the buffers of the double-buffered registers are programmed and the GRE bit is set before the next MRS occurs. The scenario presented in (b) describes an error condition where the software did not signal the hardware that all registers are ready for re-load before the next MRS. This is indicated by GRE being at zero when MRS occurs. In this case, the MRS_RE flag is set indicating the fault condition.

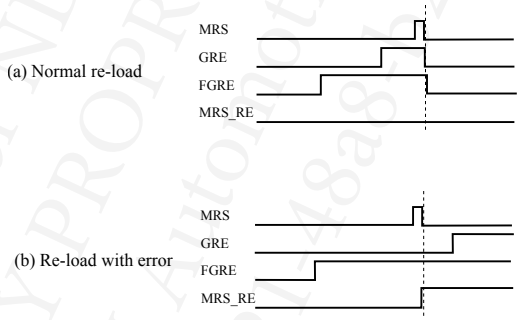


Figure 525. Reload scenarios

NOTE

If MRS is generated by software (by writing 1 b to CTU_CR [MRS_SG]) after GRE is set then due to longer processing path of GRE bit compared to MRS_SG bit, the effect of GRE may reach later occasionally in MT clock domain than MRS_SG (based on relative positioning of clock edges), consequently the original write sequence (GRE set -> MR pulse) may get reversed (MR pulse -> GRE set) in MT clock domain. This leads to loading error and MRS_RE is set with GRE and FGRE is not cleared. To avoid this situation user must maintain a minimum delay between the writes of GRE bit and MRS_SG bit as defined in the equation below.

The delay between writes of GRE and MRS_SG need to meet the below equation:

D min = 2 (system clock period + CTU timer clock period)

57.8 Interrupts and DMA requests

57.8.1 DMA support

The DMA interface can be used to configure the CTU registers. A DMA request is issued if MRS occurs and the GRE bit is set. In this case, the previously transferred data is loaded from the buffers into the registers and a new transfer cycle is initiated with the DMA request. The DMA done signal from the DMA controller is used to set the GRE bit that allows the reload to be performed. The following figure describes this operation.

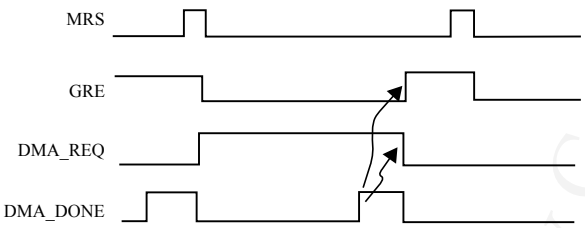


Figure 526. Register reload using DMA

DMA support can also be provided for reading FIFO stored data. Each FIFO can be configured to perform a DMA request when the number of stored words reaches a threshold value defined in the FTH register. See the FIFO Threshold(FTH), register for more information. After a DMA_DONE and the remaining data in the FIFO is below the watermark, the DMA request is removed. For an efficient operation, the DMA should be configured to execute a loop reading all the data in the FIFO, considering the amount of data the same as defined by the watermark.

57.8.2 CTU faults and errors

The CTU detects and signals the following error conditions:

- MRS reload error: MRS occurs while a user is updates the double-buffered registers.
- Sequential mode trigger overrun SM_TO: More than eight event signals occur before the next MRS in the TGS sequential mode.
- Trigger event overrun indicated by interrupt signals ADC_OE .
- The CTU allows the user to pre-set a trigger to eTimer1 in the middle of an ADC conversion. In parallel mode, even two ADC commands can be issued at the same time. The error only occurs if the second trigger is targeting the same ADC.
- Invalid (unrecognized) ADC command occurs and the ICE bit is set.
- Master reload signal overrun MRS_O: MRS occurs before all triggers selected by Tn_E and TnETE or TnTmE or Tn_ADCE bits in THCR1/2 registers have occurred.
- TGS_OSM TGS overrun in sequential mode: new event occurs before the trigger event selected by the previous event had occurred.

The faults/errors flags in the CTU error flag register and in the CTU interrupt flag register can be cleared by writing 1 to the respective bit. Writing 0 to these bits has no effect.

NOTE

The CTU does not support write-protection mechanism for any register, indicating that all registers are accessible at any time.

The CTU does not generate a transfer error when its reserved address space is accessed.

57.8.2.1 CTU parallel list errors

The command list operation in parallel mode may generate errors because two lists may be in execution at the same time. Following are some of the error conditions:

Commands for the same ADC are issued at the same time by two lists. In this case:

- One of the commands is disregarded and an error is issued.

- The LIST_BE bit in the EFR register is set.
- The address of the list commands is shown in the LISTCSR register.
- The address of the command that was not executed is indicated in the LISTCSR register.

A third list is triggered when two lists are already being executed. In this case:

- The third list is not executed and an error is indicated by the ADC_OE bit in the EFR register.

One list is being executed and a second list is triggered; the first command of the second list is for the same ADC being used by the first list. In this case:

- The command of the second list is not executed.
- The address of the second list command is shown in the LISTCSR register.
- The LIST_BE bit in the EFR register is set.

57.8.3 CTU interrupt/DMA requests

The CTU generates the following interrupt/DMA requests which are controlled by the IR register:

- Error interrupt request, see CTU faults and errors (1 interrupt line)
- ADC command interrupt request (1 interrupt line).
- MRS interrupt request (1 interrupt line)
- Trigger event interrupt request (1 interrupt line for each trigger event)
- FIFOs interrupt and/or DMA transfer request (1 interrupt line for each FIFO). See FIFO control register, FCR.
- DMA transfer request on the MRS occurrence if GRE bit is set

57.9 Self-test mode

The CTU is capable of issuing a self-test command to the ADC.

The CLRx register is used to set up a self-test command. Table 522 describes the ST0 and ST1 bits that control self-test functionality.

Table 522. Self-test command

ST_CMS bit	ST1 bit	ST0 bit	Command description
0	Not Used	0	no self test single conversion
1	Not Used	0	no self test dual conversion
Not Used	0	1	self test command
0	1	1	no self test command single conversion
1	1	1	no self test command dual conversion

Two algorithms are available in self-test mode, see Table 523.

Table 523. Self-test algorithm selection

ALG[1]	Number of steps	Executed Algorithm
00	3	Algorithm S
01[2]	-	Reserved

Table continues on the next page...

Table 523. Self-test algorithm selection (continued)

ALG[1]	Number of steps	Executed Algorithm
10	12	Algorithm C
11	15	Algorithm FULL (S + C)

1. ALG refers to the CLRx register field.
2. The 01 selection causes a command execution error and the LIST_BE flag is set in the Error Flag Register (EFR).

The self-test S algorithm is executed in an atomic burst of conversions independent of the BSIZE value in the CLRx register. The steps for this algorithm execution are these:

- Step 0: VREF test
- Step 1: VDD/VREF test
- Step 2: VREF/VREF test

When the FULL algorithm is selected, the CTU executes the S algorithm followed by the C algorithm.

The self-test execution follows the trigger scheme selected for the CTU, and either the triggered mode or sequential mode can be selected.

To completely execute one of the algorithms described in Table 523, a certain number of steps need to be performed. Each step corresponds to the time for one ADC conversion to complete. These steps may be executed at once, by just one command, or they can be interleaved with other ADC commands. To executed self-test commands interleaved with other conversion commands it is sufficient to set the BSIZE field in the CLRx register for a value lower than the number of steps defined for the selected algorithm. The following figure shows the execution of the C algorithm interleaved with regular ADC commands.

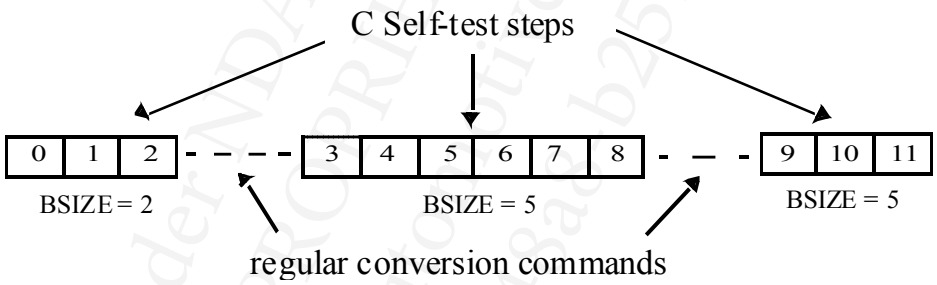


Figure 527. C Algorithm execution interleaved with others ADC commands

In the figure above, the last C self-test sequence intentionally shows BSIZE set to a larger value than the remaining steps needed to complete the C algorithm. In this case, the number of executed self-test steps do not follow BSIZE but instead follow the number of remaining steps for the selected self-test algorithm, which in this case is three steps.

NOTE

The S algorithm steps do not interleave with other ADC commands even if BSIZE is smaller than 3. Independent of the BSIZE value, all the three steps of the S algorithm are executed in a contiguous sequence.

The number of steps needed to completely execute the selected self-test algorithm are as follows:

- S algorithm requires three steps
- C algorithm requires 12 steps
- FULL runs all algorithms and requires 15 steps

In the figure below, the FULL algorithm is executed, interleaved with other ADC commands. The BSIZE = 3 in the first and second self-test commands produces different number of executed self-test steps.

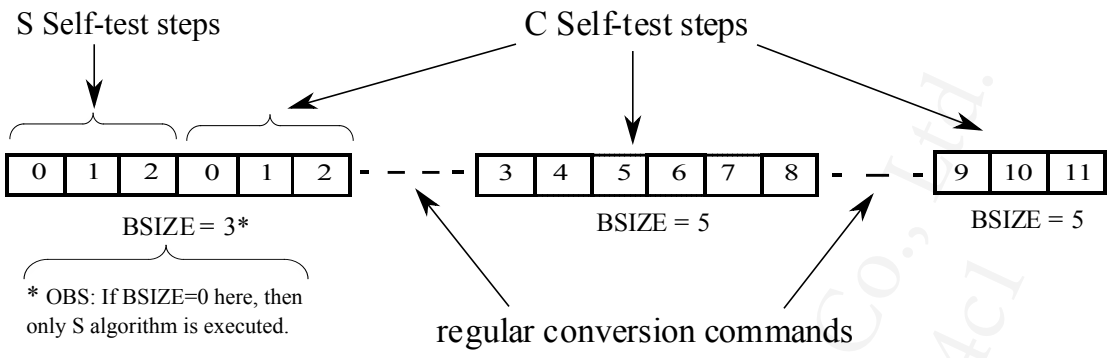


Figure 528. FULL algorithm execution interleaved with other ADC commands

57.9.1 Self-test execution errors

When a self-test algorithm is initiated for an ADC, it should complete before any other self-test algorithm is started in the same ADC. In other words, one self-test algorithm should not be interleaved with another self-test algorithm. Any attempt to do so causes the ongoing list execution to be halted and then cause the LIST_BE error flag to be set. See Error Flag Register (EFR)) for more information. To evaluate if a self-test algorithm is running, the CS bit can be checked. See Error Flag Register (EFR) for more information.. If set, this bit indicates a self-test has initiated in one of the ADCs. If a self-test algorithm conflict occurs, as described, the recovery procedure is to issue a software reset through the CTU_ADC_R bit in the CR register.

An error condition is also caused when the reserved value 01 is selected for the self-test algorithm see Error Flag Register (EFR).

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-3

Chapter 58
Analog-to-Digital Converter (SAR_ADC)

58.1 Chip-specific SAR_ADC information

58.1.1 SAR_ADC instances

The chip has two SAR_ADC instances, SAR_ADC_0 and SAR_ADC_1.

58.1.2 SAR_ADC resolution

Each SAR_ADC instance has 12-bit resolution.

58.1.3 Analog watchdogs

Each SAR_ADC instance has eight analog watchdogs.

58.1.4 Normal and injection trigger

Normal and injection triggers are supported through package pins and additional hardware triggers via CTU. CTU can trigger conversion only on external channels (mapped from pads). For details, see the IOMUX file attached to this document. This chip does not support CTU-initiated self-test on SAR_ADCs.

58.1.5 SAR_ADC channel assignments

This chip does not support any external analog muxes.

The following table identifies channel assignments for the SAR_ADC module.

Channel number	Group	Channel name	Channel type
SAR_ADC_0			
0	0	ADC_CH_00	Precision
1	0	ADC_CH_01	Precision
2	0	ADC_CH_02	Precision
3	0	ADC_CH_03	Precision
4	0	ADC_CH_04	Precision
5	0	ADC_CH_05	Precision
6	0	ADC_CH_06	Precision
7	0	ADC_CH_07	Precision
8–31	Reserved	Reserved	Reserved
32	32	Scaled Bandgap reference	Internal
33–34	Reserved	Reserved	Reserved
35	32	DVDD/2	Internal

Table continues on the next page...

Table continued from the previous page...

Channel number	Group	Channel name	Channel type
36	32	AVDD[1]	Internal
37	32	VREFL	Internal
38	32	VREFH	Internal
SAR_ADC_1			
0	0	ADC_CH_08	Precision
1	0	ADC_CH_09	Precision
2	0	ADC_CH_10	Precision
3	0	ADC_CH_11	Precision
4	0	ADC_CH_04	Precision
5	0	ADC_CH_05	Precision
6	0	ADC_CH_06	Precision
7	0	ADC_CH_07	Precision
8-31	Reserved	Reserved	Reserved
32	32	Scaled Bandgap reference	Internal
33-34	Reserved	Reserved	Reserved
35	32	DVDD/2	Internal
36	32	AVDD[1]	Internal
37	32	VREFL	Internal
38	32	VREFH	Internal
39	Reserved	Reserved	Reserved

1. AVDD corresponds to VDD_ADC.

58.2 Introduction

The ADC has 12-bit resolution and uses the Successive Approximation Register (SAR) method. It provides fast and accurate conversion, and is highly configurable for a wide range of applications.

58.3 Overview

The following figure shows the high-level interface diagram of the SAR_ADC.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4e21-48a8-b0f3-590d3ab624c1

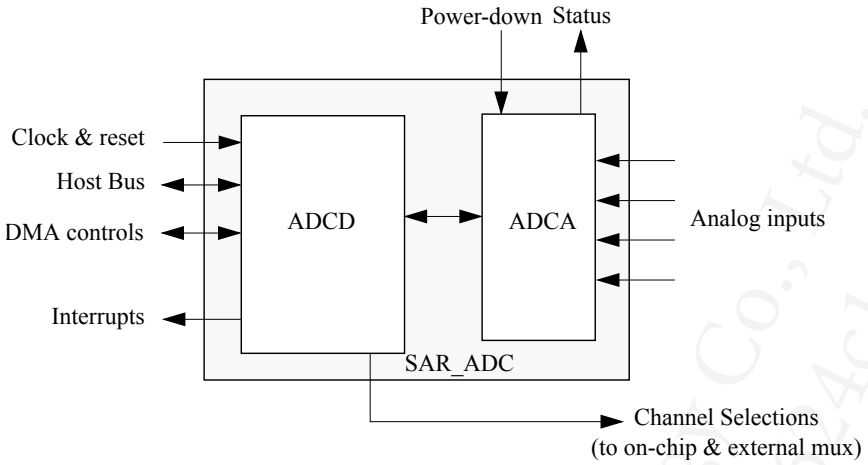


Figure 529. SAR_ADC high-level interface diagram

58.3.1 Interfaces

The SAR_ADC interfaces as shown in Figure 529 are as follows:

- Analog inputs to sample the channels to be converted.
- Channel selection controls are provided to use internal or external analog multiplexers.
- Host Bus allows access to configuration, control, and status registers which are used to start/stop conversions, monitor the conversion results, errors, violations, and so on.
- DMA interface transfers conversion results by the DMA controller.
- Interrupts can be generated for various events depending on configuration (data availability, threshold violation, error, and so on).
- Hardware triggers are used to start conversion from sources external to the SAR_ADC.
- Cross Triggering Unit (CTU) interface allows triggers from the CTU on the chip to start conversions.
- Power-down controls are used to place the analog block into a deactivated state.
- Fault indications from runtime self-test are generated for supply and analog faults.

58.3.2 Submodules

The SAR_ADC consists of two major functional parts:

- ADC Digital (ADCD)
- ADC Analog hard macro cell (ADCA)

The ADCD controls the ADCA. The ADCD holds the configuration, control, and status registers that are accessible through software using the host interface. The ADCD executes the SAR Control algorithm. The selection of channel multiplexers and analog gates are driven by the ADCD based on user configuration. Conversion results are stored in registers, and the results can be transferred to memory by DMA or read by host access.

The ADCA holds all analog circuitry which consists of a comparator, capacitor bank, bias generation, and so on. Only one channel is processed at a time. Multiple channels are processed sequentially by multiplexing the analog inputs. The duration time that each channel is sampled is configurable by the user. The hold value is used to calculate a channels digital code by successively comparing it with an analog voltage generated by a Digital to Analog Converter (DAC) inside the ADCA.

58.3.3 Clock and reset

The SAR_ADC has only one clock input (bus clock).

This clock is used for accessing registers through the Host Bus and operation of all interfaces.

The conversion clock (AD_CLK), is derived from the bus clock input. The AD_CLK can be equal to, or half of the bus clock frequency based on configuration (see register bit descriptions of MCR[ADCLKSE]).

If MCR[ADCLKSE] = 1, then the AD_CLK is equal to the bus clock. If MCR[ADCLKSE] = 0, then the AD_CLK is half of the bus clock frequency. However, depending on configuration and performance requirements, MCR[ADCLKSE] may always need to be 1.

The SAR_ADC has two resets, Functional (normal) and Calibration. Calibration reset is applied only when calibration is needed along, and is asserted with a functional reset. A functional reset can happen any time without a calibration reset.

Calibration reset is provided to hold internal calibration and status values if a functional reset occurs. The following register and field values are maintained after a functional reset:

- Main Status Register (MSR)
  - CALIBRTD
  - CALFAIL
- User OFFSET and Gain Register (USROFSGN) (Full register)
  - GAINUSER
  - OFFSUSER
- Calibration Status register (CALSTAT) (Full register)
  - TEST_RESULT
  - STAT[1:14]

58.3.4 Special channels

Reference voltages and supply voltages are available at predefined SAR_ADC channels.

58.4 Features

The general feature list of the SAR_ADC is shown below. Feature availability is chip-specific and is defined during module integration. See chip-specific SAR_ADC information for details.

- 4'd12-bit resolution
- Multiple modes of starting conversion (Normal, Injected, CTU)
  - Normal mode supports One-Shot and Scan (continuous) conversions
  - Injected mode supports One-Shot conversions only
  - CTU mode runs a single conversion per trigger
- Software-initiated conversions in Normal and Injected modes, or external hardware trigger
- Two different abort features for either a single or chain conversion in Normal and Injected modes
- Independent data registers for each channel contain information about mode of conversion, data validity, overwrite status, and conversion data
- Alternate analog watchdog thresholds (threshold selected through input ports)
- Programmable DMA enables for each channel
- Individual interrupt flags for the following conditions:
  - End of conversion of a single channel for Normal and Injected modes
  - End of chain conversion for both Normal and Injected modes
  - End of conversion in CTU mode

  - Watchdog threshold violations
- Programmable presampling for channels
- Auto-Clock-Off feature for improved power performance
- Power-Down mode to place the SAR_ADC in power-down state
- Programmable clock prescaler for SAR_ADC (bus clock, or bus clock divided by two or four)
- Software-initiated calibration
- Self-test feature

58.5 Functional description

The ADCD interfaces are shown in Figure 529. The available functionality in the ADCD is described in the following sections (see chip-specific SAR_ADC information for details).

58.5.1 Conversion

Conversions are software or hardware initiated.

The following list shows the methods of initiating conversions in order of priority (highest to lowest):

1. CTU trigger (hardware only)
2. Injection (see Injected Conversion mode)
3. Normal (see Normal Conversion mode)

Each type of conversion method has different processing features.

CTU trigger starts a single conversion for a single channel.

Injection can start a single chain of conversions one at a time, where a chain may have from one up to the maximum number of channels available (each channel appears once in the chain).

Normal conversion can start a continuous or single chain of conversion one at a time. In continuous mode, the chain of conversions is repeated unless you stop them.

When a normal conversion is in process, the starting injected conversions are done by software or hardware. Which allows a chain of channels that cannot wait to be converted (as selected by JCMR) to be handled immediately.

Conversions initiated by the CTU (one channel per trigger) are processed immediately by suspending normal conversions and discarding any ongoing injected conversion chain.

Analog watchdogs allow continuous hardware monitoring of input channel conversion results.

A built-in self-test function enables periodic checking of ADCA health. If a fault is detected, the FCCU can be signaled. The severity of the fault is programmable (critical/non-critical).

NOTE

ADC configurations must be changed only when ADC is in idle or power-down state (MSR[NSTART]=0 and MSR[ADCSTATUS] = idle/power-down).

58.5.2 Normal Conversion mode

The conversion method which is generally used to convert channels is Normal Conversion mode. To select channels for conversion, you program the Normal Conversion Mask Register (NCMR). Each channel can be individually enabled by setting the corresponding field in NCMR:

- NCMR0 selects channels in Group0.
- NCMR1 selects channels in Group32.

The mask registers must be programmed before starting the conversion and cannot be modified until all selected channel conversions have completed. A conversion always starts from the lowest channel number and continues sequentially to the highest channel number.

58.5.2.1 Starting conversions in Normal Conversion mode

After programming the required fields of the Main Configuration Register (MCR) and NCMR, conversion starts as follows:

- By software:
  - Conversion chain starts when one is written to MCR[NSTART].
- By hardware trigger:
  - If external trigger is enabled (MCR[TRGEN] = 1), and an external trigger enable is detected, the conversion will start. The enable is checked only during the start of conversion.
    - Detection of an active edge is defined by MCR[EDGE] (rising = 1, falling = 0). When an external trigger is recognized, MSR[NSTART] is set and normal conversion starts.

NOTE

A software-initiated conversion in Normal Conversion mode is ignored during any ongoing conversion chain.

NOTE

There must be a gap of three cycles between configuration of the NCMR and the start of conversion (setting of MCR[NSTART] or arrival of hardware start trigger) for proper SAR_ADC operation. The SAR_ADC calculates the total number of channels to be converted during this period.

58.5.2.2 Operation modes in Normal Conversion mode

Two operation modes are available for normal conversions:

- One-Shot Operation mode
- Scan Operation mode

The MCR[MODE] field determines the operating mode in normal conversion. The first phase of any conversion process involves sampling the analog channel. Which is followed by the conversion phase, when the sampled analog value is converted to digital as shown in Figure 530.



Figure 530. Normal conversion flow

In **One-Shot Operation mode** (MCR[MODE] = 0), a sequential conversion specified in the NCMR mask registers is performed once. At the end of each conversion, the digital result of the conversion is stored in the corresponding data register (CDR*n*; *n* = channel number).

For example: Channels A-B-C-D-E-F-G-H are available on the chip, where channels B-D-E are to be converted in One-Shot Operation mode. Conversion starts from channel B, followed by conversion of channels D-E. At the end of conversion of channel E the channel scanning stops.

MSR[NSTART] is automatically set when the normal conversion starts. At that time MCR[NSTART] is reset by hardware, allowing the software to program a new start of conversion in advance. In this case, the new requested conversion starts after completion of the currently running conversion. However, for correct device function, it is required to program MCR[NSTART] for the new conversion only after the current conversion is complete. Software can wait for the end of conversion (ISR[ECH] = 1) prior to setting MCR[NSTART] again.

If the conversion chain starts by external trigger (if MCR[TRGEN] = 1), the ADC cannot monitor the trigger input until the conversion chain finished. After the chain is finished, the next chain starts by subsequent hardware trigger edge.

In **Scan Operation mode** (MCR[MODE] = 1), a sequential conversion of *n* channels specified in the NCMR registers is continuously performed. At the end of each conversion the converted result is stored into the corresponding channel data register, as it is in One-Shot Operation mode.

MSR[NSTART] is automatically set when normal conversion starts. Unlike One-Shot Operation mode, MCR[NSTART] is not reset in Scan Operation mode. It can be reset by software when you need to stop the Scan Operation mode. In that case, the ADC will complete the current chain, and after the last conversion, it clears MSR[NSTART].

For example: Channels A-B-C-D-E-F-G-H are available on the chip, where channels B-D-E are to be converted in Scan Operation mode. Conversion starts from channel B followed by conversion of channels D-E. At the end of conversion of channel E the scanning of channel B starts again, followed by conversion of channels D-E. This sequence repeats itself until MCR[NSTART] is reset by software.

This scan chain can be stopped by writing zero to MCR[NSTART]. The conversion stops when the ongoing chain is finished.

If the conversion is started by an external trigger, MCR[NSTART] is not set. As a consequence, after it is started, the only way to stop a Scan Operation mode conversion started by external trigger is to write MCR[MODE] = 0.

End of conversion:

In both modes, at the end of each conversion an End of Conversion (EOC) interrupt is issued (if enabled by the corresponding mask bit in CIMR and IMR).

After conversion of all the selected channels in NCMR has completed, the conversion operation is considered finished. ISR[ECH] is then set and End of Chain (ECH) interrupt is issued (if enabled in IMR[MSKECH]).

The corresponding channel bit of CEOCFR is updated to indicate data is available in data register CDR*n* of the corresponding channel.

If there is no channel selected in NCMR, and there is a start of conversion trigger, ISR[ECH] is set and an ECH interrupt is issued immediately, if enabled.

NOTE

In scan (continuous) mode, ADC conversions happen very regularly as per timing derived by sample time and resolution. The host reads the result registers after checking ECH flag. The reading time of the host after setting ECH could vary based on the load on it. So, it may occasionally get invalid data (valid bit 0b in data CDR register) in channels. To avoid such scenarios, check the ECH flag twice before reading the conversion results, to ensure that the host gets valid data always for all channels. The host needs to clear the ECH flag each time it finds it set. The sequence will be as below:

1. Host waits for ECH. On ECH, it clears it.
2. Host waits for second ECH. On ECH, it clears it.
3. Host reads data for all channels.
4. Repeat steps 1–3.

58.5.3 Injected Conversion mode

An injected conversion is a conversion of a programmed chain that can be injected anytime into an ongoing conversion chain processed in Normal Conversion mode. You require to configure Injected Mask Register(s) (JCMR*n*). In Normal Conversion mode each internal or external channel can be individually selected:

- JCMR0 for channels in Group0
- JCMR1 for channels in Group32

An injected conversion chain can only be run in One-Shot Operation mode and suspends the ongoing conversion chain processed in Normal Conversion mode. When an injected conversion chain is inserted into an ongoing chain processed in Normal Conversion mode, the ongoing chain is suspended and the injected conversion request is processed. After conversion completion of the last channel of the injected chain, conversion resumes from the channel at the chain which was suspended in Normal Conversion mode, as shown in Figure 531.

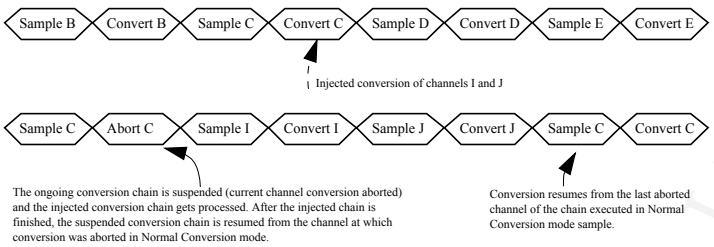


Figure 531. Injected sample/conversion sequence

58.5.3.1 Starting conversions in Injected Conversion mode

The injected conversion chain can be started by using one of the following methods:

- By software: When external triggering is disabled, MCR[JSTART] can be set. Setting MCR[JSTART] causes any Normal Conversion mode chain to be suspended and the injected chain to start processing.

MSR[JSTART] is automatically set when the injected conversion chain starts, and MCR[JSTART] is cleared. This allows the software to program a new start of conversion in advance. If a new injected conversion request occurs while an injected conversion chain is in progress (for example, setting MCR[JSTART] while MSR[JSTART] is set), then conversion will start after the current conversion has completed.

At the end of each conversion, an End Of Injected Conversion (JEOC) (ISR[JEOC] = 1) interrupt is issued (that is, IMR[MSKJEOC] = 1), and at the end of the chain an End of Injected Chain (JECH) (ISR[JECH] = 1) interrupt is issued (that is, IMR[MSKJECH] = 1).

The corresponding channel bit of CEOCFR*n* is set, indicating that data is available in the CDR of the respective channel.

After it is started, an injected chain conversion cannot be interrupted.

NOTE

If there is no channel selected in JCMR*n*, and an injected conversion is started, then ISR[JECH] is immediately set and the corresponding interrupt is asserted (if enabled). This behavior is independent of normal conversion status.

For correct operation of SAR_ADC, there must be a gap of at least three cycles between configuration of JCMR*n* and the start of a conversion (setting MCR[JSTART], or receiving a hardware start trigger (external/internal). The SAR_ADC calculates the total number of channels to be converted during this period.

58.5.4 Abort of conversion

Two different abort functions are provided:

- Abort of a single channel
- Abort of chain

You can abort the ongoing conversion by setting MCR[ABORT]. SAR_ADC then aborts the current conversion and immediately starts the conversion of the next channel of the chain. The abort action may take 1–4 cycles of bus clock to abort the current conversion, depending on the state of the conversion. At the start of any conversion and at the end of a particular conversion, when internal state counters are changing, then the abort get delay by a maximum of three cycles. This allows all states to be placed in stable condition before aborting.

So, you are advised not to change the abort field status for the next three cycles to achieve this effect.

Abort also not to be programmed along with and within three cycles of programming MCR[NSTART] or MCR[JSTART].

In the case of an abort operation, MSR[NSTART/JSTART] remains set if not in the last channel of the chain, and MCR[ABORT] is reset as soon as the channel is aborted. The EOC corresponding to the aborted channel is not generated. This behavior is true for Normal or Injected Conversion modes. If the last channel of a chain is aborted, the end of chain is reported, generating an ECH interrupt.

It is also possible to abort the current chain of conversions by setting MCR[ABORTCHAIN]. In that case, the behavior of the ADC depends on MCR[MODE] (One-Shot/Scan Operation modes). If Scan Operation mode is disabled, MSR[NSTART] is automatically reset together with MCR[ABORTCHAIN]. Otherwise, in Scan Operation mode, a new chain is started. The EOC of the current aborted conversion is not generated but an ECH interrupt is generated to signal the end of the chain.

NOTE

For a single channel chain, ECH is not generated in the case of an abort or abort chain.

When an ABORTCHAIN is requested while an injected chain is running over a suspended normal chain, both injected and normal chains are aborted. Both ISR[ECH] and ISR[JECH] are set. MSR[JSTART] is reset to mark the end of the injected chain. MSR[NSTART] is reset if normal conversion is running in One-Shot mode. Otherwise, in Scan mode it remains set and a new chain is started.

58.5.5 CTU

The CTU interface enhances the injected conversion capability of the ADC. They contain multiple control inputs that can be used to select the channels to be converted from the appropriate event configuration register. Figure 532 shows the interface.

The CTU generates a trigger ctu_trigger and a channel number ctu_numchannel to be converted. A single channel is converted for each request. After performing the conversion, the ADC returns the result on the ctu_dataout bus together with two output signals named ctu_nextcmd and ctu_push. The assertion of signal ctu_nextcmd means that ADC is ready to accept the next trigger from CTU. The ctu_push signal is asserted at the end of conversion, meaning that conversion is finished and the conversion result available at output ctu_dataout is valid.

The conversion result is also saved in the corresponding channel's data register. It is compared with watchdog thresholds when requested.

The CTU and ADC are always connected to the same clock domain and therefore are synchronous (same frequency and domain).

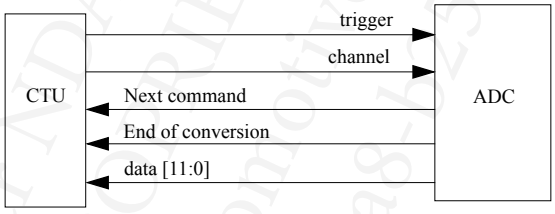


Figure 532. CTU interface

Program MCR[CTUEN] to enable the CTU interface. The operating mode (Trigger/Control) can be fixed or programmable. If available, in Power-Down mode MCR[CTU_MODE] program the CTU interface operating mode.

58.5.6 ADC clock prescaler and sample time settings

The AD_CLK frequency can be scaled by programming MCR[ADCLKSE]. If MCR[ADCLKSE] is set, then the AD_CLK frequency is the same as the bus clock. Otherwise, AD_CLK frequency is half of bus clock, MCR[ADCLKSE] bit can be written in power-down only when MCR[PWDN] = 1.

In order to support different sampling times for different type of channels the different conversion timing registers are used (CTR0, CTR1). CTR0 is associated to channels (0 to 31), CTR1 is associated to channels (32 to 63). See the register description for details.

58.5.7 Presampling

The presampling feature allows precharging or discharging the SAR_ADC internal sample capacitor node to a defined level before sampling of the selected analog input channel starts. This is useful for resetting information (history effect/offset) regarding the last converted data. During presampling, ADC samples the internally generated voltage, while in the sampling SAR_ADC samples analog input coming from pads.

Presampling can be enabled/disabled on a per-channel basis by setting the corresponding bits in the PSR0, PSR1 registers. PSR0 serves channels 0–31 , PSR1 serves channels 32–63 .

After enabling the presampling for a channel, the normal sequence of operation will be Presampling → Sampling → Evaluation for that channel. Sampling of all channels can be bypassed by setting PSCR[PRECONV]. When sampling of a channel is bypassed, the sampled and stored internal voltage applied during the presampling phase get converted (see Figure 533). See Conversion time for the timing equation for conversion.

Presampling does not apply to self-test channels.

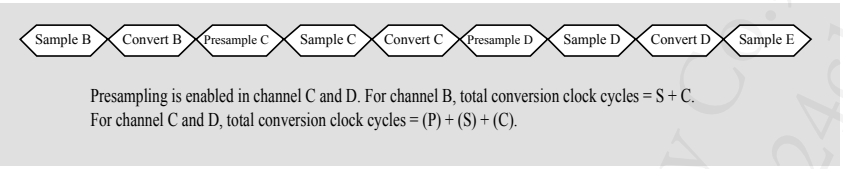


Figure 533. Presampling sequence

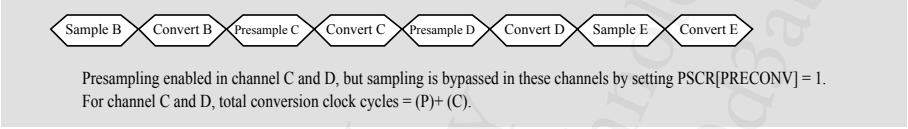


Figure 534. Presampling sequence with PRECONV=1

58.5.7.1 Presampling channel enable

The presampling channels are selected by programming the PSCR. It is possible to select between four internally generated voltages depending on the value of PSCR[PREVAL*n*], as shown in the following table.

Table 524. Presampling voltage selection

PSCR[PREVAL*n*]	Presampling voltage
00	Presample voltage - 1: (DVDD0P8/2)
01	Presample voltage - 2: (AVDD1p8/4)
10	Presample voltage - 3: VREFL_1p8
11	Presample voltage - 4: VREFH_1p8

NOTE

The voltage for presampling is (AVDD1p8/4) when PSCR[PREVAL]=01b. However, when this value is measured in sample bypass mode (PSCR[PRECONV]=1), the raw conversion result is shifted by two bits to the left (multiplied by 4) before error compensation. So the expected value of the final conversion result is AVDD1p8 (deviation may be in the range 0–3 LSB, due to left shift).

The PSCR[PREVAL*n*] fields are associated with SAR_ADC channels as follows:

- PREVAL0 for channels 0–31
- PREVAL1 for channels 32–63

58.5.8 Programmable analog watchdog

The analog watchdogs are used to monitor the conversion result to see if it is within defined limits (as shown in the figure below), specified by an upper and a lower threshold value named THRH and THRL, respectively.

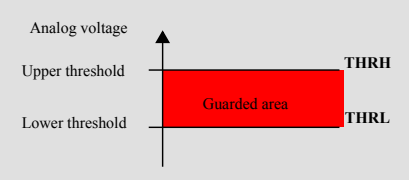


Figure 535. Guarded area

After the conversion of the selected channel, a comparison is performed between the converted value and the threshold values. If the converted value is outside the threshold values, then a corresponding threshold violation interrupt is generated.

The comparison result is stored as WDG*n*H and WDG*n*L fields in WTISR as explained in the following table. Depending on the values of WTIMR[MSKWDG*n*L] and WTIMR[MSKWDG*n*H] mask bits, an interrupt is generated on a threshold violation.

Table 525. Values of WDGxH and WDGxL fields

WDG*n*H	WDG*n*L	Converted data
1	0	Converted data > THRH
0	1	Converted data < THRL
0	0	THRH >= converted data >= THRL

Depending on the device settings, the number of analog watchdogs can vary for each instance of the SAR_ADC (see chip-specific SAR_ADC information for details).

Analog watchdog setting

For each watchdog there is one threshold value (low and high) register (THRHLR*n* [*n*=0..15]).

The analog watchdog for each channel can be enabled independently by programming the corresponding CWENR fields.

- CWENR0 for Group0 channels
- CWENR1 for Group32 channels

Watchdog selection of each channel is done by configuring CWSELR*n*[WSEL_CH*m*] (where *n* = CWSELR instance and *m* = channel number).

Each CWSELR*n* holds a selection for a maximum of eight consecutive channels (for example, CWSELR0 holds eight WSEL_CH fields for channels 0–7, CWSELR1 is for channels 8–15, and so on).

If the conversion result of a selected channel is outside the range specified by threshold values, then the corresponding bit is set in the Analog Watchdog Out of Range Register (AWORR) (for example, if channel number 12 is to be monitored with the threshold values in register THRHLR3, then WSEL_CH4 of CWSELR1 is to be programmed with 3). Enabling the watchdog is done by setting the corresponding bit for channel 12 in register CWENR0.

In this setting, a set of threshold values (THRHLR*n*) can be linked to several SAR_ADC channels. The threshold values to be selected for a channel must be programmed only once in CWSELR*n* register.

NOTE

If the higher threshold of the analog watchdog is programmed to be below the lower threshold and the conversion result is below the lower threshold, then the WTISR[WDG*n*L] interrupt for the low threshold violation is set. If the conversion result is greater than the lower threshold (consequently also greater than the higher threshold), then the interrupt WTISR[WDG*n*H] for high threshold violation is set. Thus, you should take care to avoid that situation as it could lead to misinterpretation of the watchdog interrupts.

58.5.9 DMA functionality

Conversion data of any channel can be transferred from a register to system memory through Direct Memory Access (DMA). The DMA transfers can be enabled by setting DMAE[DMAEN]. When enabled, the on-chip DMA controller can receive a DMA request after the conversion of every channel by setting the respective masking bit in the DMAR*n* registers. The DMA masking registers must be programmed before starting any conversion.

The DMA request to DMA controller can be cleared at different times in two modes:

MODE 1: Clearing of DMA request on acknowledgment from DMA controller (setting DMAE [DCLR] = 0).

MODE 2: Clearing of DMA request on read to data registers (DMAE[DCLR] = 1).

The figures below show the operation of DMA in two modes (cycle counts are for a typical case).

NOTE

The DMA should complete the read of the SAR_ADC data within the throughput rate of SAR_ADC. For example, SAR_ADC can support a maximum throughput rate of 1 MSPS. Thus, DMA should complete a data read within 1 µs after assertion of dma requests (ipd_req).

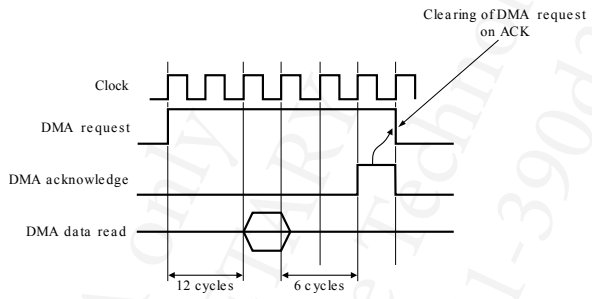


Figure 536. DMA operation MODE 1 (DMAE [DCLR] = 0)

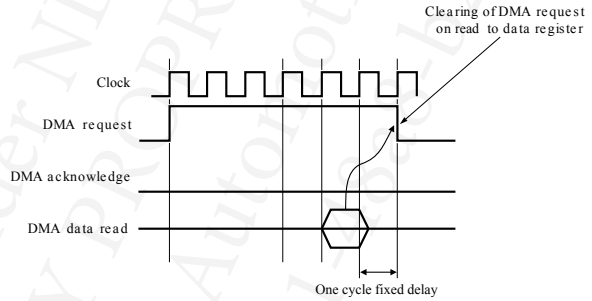


Figure 537. DMA operation MODE 2 (DMAE [DCLR] = 1)

58.5.10 Interrupts

SAR_ADC generates the following maskable interrupt signals:

- EOC (End of Conversion) interrupt request
- ECH (End of Chain) interrupt request
- JEOC (End of Injected Conversion mode) interrupt request
- JECH (End of injected Chain) interrupt request
- EOCTU (End of CTU conversion) interrupt request
- WDG*n*L and WDG*n*H (watchdog threshold) interrupt requests
- Self-testing Interrupts

Interrupts are generated during the conversion process to signal events such as EOC and so on, as explained in register description for ISR. Two registers named ISR (Interrupt Status Register) and IMR (Interrupt Mask Request Register) are provided in order to check and enable the interrupt request to external interrupt controller.

Interrupts can be individually enabled on a channel-by-channel basis by programming CIMR (Channel Interrupt Mask).

For pending EOC interrupts, registers with one bit per channel are provided to indicate completed conversions on a channel basis (CEOCFR*n*).

Interrupts generated due to the analog watchdogs are handled by two 32-bit registers named WTISR (Watchdog Threshold Interrupt Status Register) and WTIMR (Interrupt Mask for the Watchdog Threshold Register). These registers are used to check and enable an interrupt request . A watchdog interrupt causes two corresponding bits to be set for each channel monitored. One bit is set in the WTISR[WDG*n*H] register and the other bit in the WTISR[WDG*n*L] register.

Generated events are logged into status registers and used to generate an interrupt if enabled in corresponding mask registers.

In the below list, there are different interrupt outputs of ADC. The interrupt structure is shown in Figure 538.

Each interrupt is generated by combining a group of events.

- Interrupts for EOC, ECH, EOCTU, JEOC, JECH , and End of self-test algorithm (WDG_EOA_S and WDG_EOA_C) in STSR1 are combined (logic OR) into one interrupt output. .
- Interrupts of WDG*n*H and WDG*n*L are combined on the second interrupt output.
- Interrupts of all self-test errors (watchdog threshold, sequence, and timer, that is, WDSERR, WDTERR, ERR_S0, ERR_S1, ERR_S2, and ERR_C in STSR1) are combined on the third interrupt output.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
c8c462e5-4821-48a8-b251-39013a6624c1

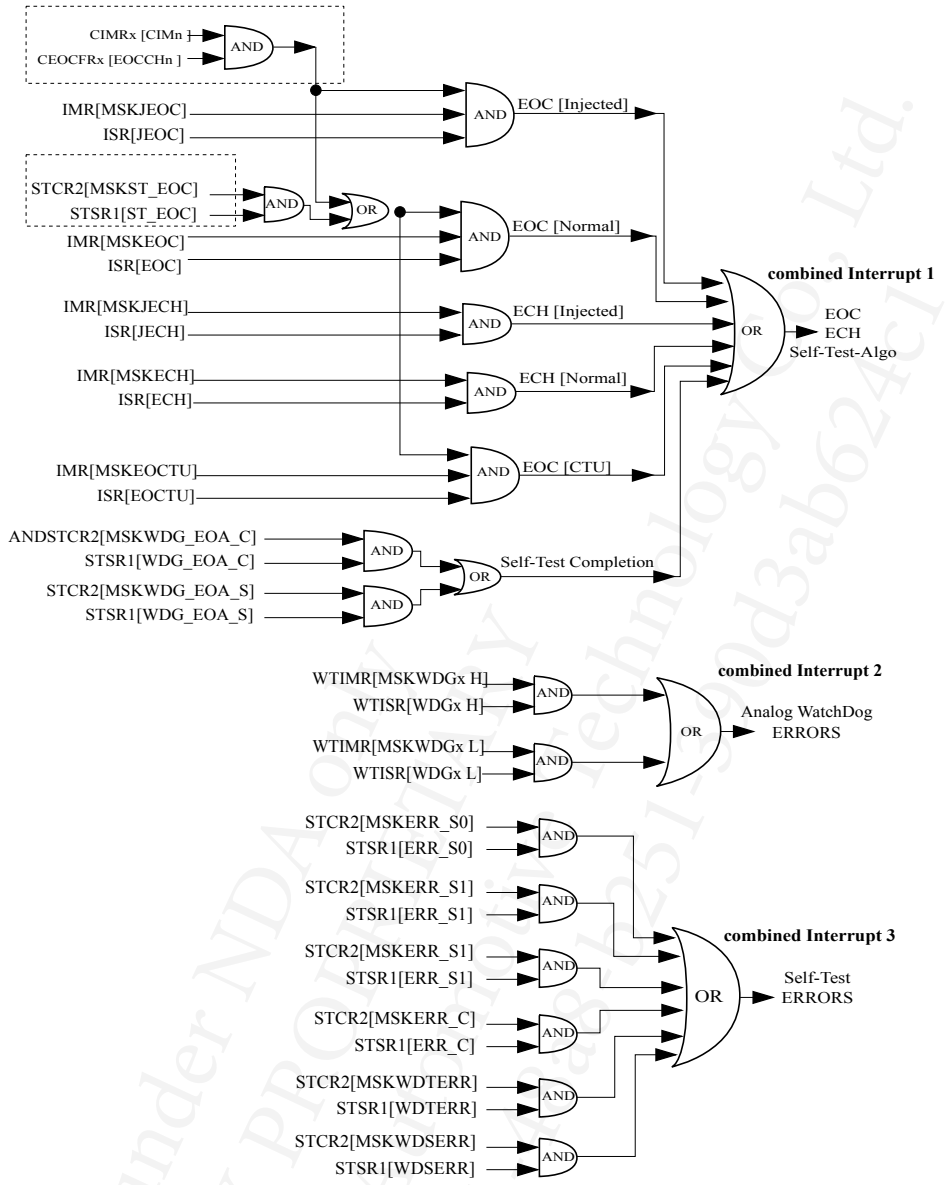


Figure 538. Interrupt structure of SAR_ADC

The ISR register contains the interrupt pending request status. If you want to clear a particular interrupt event status, then writing one to the corresponding status bit clears the pending interrupt flag (during this write operation all other bits of the ISR register must be maintained at zero).

The ISR must check the following for three different types of interrupts.

For interrupt 1:

- Read ISR to determine if the type of event falls in:
  - EOC
  - JEOC
  - ECH
  - JECH
  - EOCTU

- Read STSR1 to determine the end of self-test algorithm events (WDG_EOA_C / S) (if present).
- If the interrupt is EOC or JEOC:
  - Clear the channel pending bit of respective channel in CEOCFR*n* for EOC or JEOC in ISR.
  - Clear ISR[EOC] or ISR[JEOC].
- If the interrupt is ECH or JECH:
  - Clear ISR[ECH] or ISR[JECH].
- If the interrupt is EOCTU:
  - Clear ISR[EOCTU].
- If the interrupt is for self-test end of algorithm:
  - Clear STSR1[WDG_EOA_C/S] bit.

For interrupt 2:

- Read WTISR to determine which watchdog is giving the interrupt.
- Read AWORR*n* to determine which channel's data is creating the interrupt (if present).
- Clear the respective bit of WTISR [WDG*n*H / WDG*n*L] (*n* = watchdog number).
- Clear the respective bit of AWORR*n*[AWOR_CH*m*] (*n*=register, *m*=channel).

For Interrupt 3:

- Read STSR1 to determine the type of error that has occurred in self-test (ERR_S0/S1/S2/C/S, WDTERR, WDSERR, and so on) (if present).
- Clear the respective bit in STSR1.

The connectivity of interrupt lines at the chip level (external to SAR_ADC) determines whether software needs to read all the registers for all the interrupts or any one or a combination of any two.

58.5.11 Power-Down mode

The analog part along with SAR controller of the ADC can be put in Power-Down mode by setting MCR[PWDN] to shut down ADC hard macro and to stop the clock to SAR controller. After release of reset MCR[PWDN] remains one, so ADC analog module is kept in Power-Down mode by default. This state must be exited before starting any operation by resetting MCR[PWEN].

In Power-Down mode, no conversion can be started. If a CTU trigger pulse is received during power-down, it is discarded. You must ensure that CTU trigger is not initiated to ADC before resetting MCR[PWDN], as it may cause the CTU module to be stuck forever in a wait state.

MCR[PWDN] can be set anytime. If a conversion is ongoing, the ADC cannot move into Power-down mode immediately. The SAR_ADC enters Power-Down mode only after completion of the ongoing conversion. In Scan Operation mode the ongoing operation should be aborted manually by resetting MCR[NSTART] before or after setting MCR[PWDN].

The ADC Power-Down mode status is indicated by MSR[ADCSTATUS] when ADC enters into this mode.

If CTU is enabled, and MSR[CTUSTART] is one, then MCR[PWDN] cannot be set. When CTU Trigger mode is enabled, the application needs to wait for the end of conversion (MSR[CTUSTART] is automatically reset). When CTU Control mode is enabled, before entering Power-Down mode, the application needs to reset MCR[CTUEN] also.

If Power-Down mode is entered by setting MCR[PWDN], the process running before entry into the Power-Down mode must be restarted manually (by setting the appropriate START bit in MCR) after exiting Power-Down mode.

NOTE

Resetting MCR[PWDN] and setting MCR[NSTART] or MCR[JSTART] during the same cycle is forbidden.

58.5.12 Low-power mode support

For low-power operation, the SAR-ADC must be in an Idle state and then put into Power Down state to ensure proper signal condition at the analog boundary.

From any of the states listed below, follow the sequence of steps shown to put the SAR-ADC into a Power Down state.

1. Idle condition:
    a. Verify that the SAR_ADC is in an Idle state (MSR[ADCSTATUS[2:0]] = 000b).
    b. Set MCR[PWDN] to shut off the clock to the hard macro with the proper state of signals required for Low-power mode.
    c. Verify that the ADC is in a Power-down state (MSR[ADCSTATUS[2:0]] = 001b).
2. During calibration:

    NOTE

    Setting MCR[PWDN] during calibration is prohibited. Once calibration is started, it should be allowed to finish normally. Alternatively, it can be terminated by starting a normal conversion before setting MCR[PWDN].

    a. Check the current SAR-ADC state through MSR[ADCSTATUS[2:0]] (000b = Idle, 011b = Calibration), or through MSR[CALBUSY] (1 = calibration is in progress).
    b. When the ADC is in an Idle state, set MCR[PWDN] to shut off the clock to the hard macro with the proper state of signals required for Low-power mode.
    c. Verify that the ADC is in a Power-down state (MSR[ADCSTATUS[2:0]] = 001b).
3. During conversion (normal [one-shot]/Injected/CTU):
    a. Set MCR[PWDN].
    b. The SAR-ADC enters into Power-down mode after completing the current conversion chain (for normal and injected conversions) or the conversion for CTU-triggered conversion.
    c. Check the ADC state through MSR[ADCSTATUS[2:0]] (000b = Idle, 001b = power down).
4. During Scan mode operation of normal conversion:

    NOTE

    Setting MCR[PWDN] during Scan mode operation has no effect other than to prevent software from starting a new Normal or Injection conversion. It does not stop a conversion in progress and cannot put the ADC into a Power Down state.

    a. Software must write zero to MCR[NSTART] to stop the ongoing scan conversion before (or after) setting MCR[PWDN]. Which stops conversion at the current chain boundary.
    b. The ADC enters Power-down mode after coming to an Idle state after the MCR[PWDN] bit is set. Which shuts off the clock to the hard macro and puts analog inputs in the proper state required for Low-power mode.
    c. Verify that the ADC is in a Power-down state (MSR[ADCSTATUS[2:0]] = 001b).

58.5.13 External start feature

This feature allows synchronous conversion of two independent ADC instances operating in parallel. If the ADC is in Idle state (that is, no conversion phase ongoing and MCR[PWDN] and MCR[ACKO] are zero), and MCR[XSTRTEN] is set, an event on the external start signal causes the ADC to start a normal conversion operation. MSR[NSTART] is automatically set.

58.5.14 Auto-Clock-Off mode

To reduce the power consumption during Idle state (without going into Power-down mode), an 'Auto-Clock-Off' feature can be enabled by setting MCR[ACKO]. When enabled, the internal SAR operating clock (AD_CLK) is automatically switched off during idle periods (for example, you cannot programmed conversion).

58.5.15 Calibration

The accuracy of ADC depends on the manufacturing precision of the devices used to build the module (for instance, capacitor size matching, amplifier gain, and so on). Due to variations during manufacturing and various runtime environmental influences (like temperature), the conversion result contains errors.

To reduce or eliminate the various error contribution effects, the ADC provides self-test and calibration features.

In the calibration process, a known reference voltage is sampled and converted under controlled conditions to determine the correction values (calibration values) for offset, gain, and capacitor mismatch.

These calibration values (except gain calibration) are used in a result postprocessing step to reduce or eliminate the various error contribution effects. The gain calibration is used during sample phase to define the additional charge to be loaded in order to compensate for gain failure.

Calibration must be performed after every power-up reset (check chip-specific reset connectivity for appropriate reset applied) and whenever required in runtime operation. It is also recommended to run calibration if the operating conditions (particularly VrefH) change. Never apply functional reset during the calibration process. If applied, calibration must be rerun after exiting a reset condition; otherwise, the calibration generated values and conversion results may be unspecified.

Configuring and starting of calibration is done by programming the calibration attributes in MCR register. During calibration, the internal hardware averaging function should be enabled. Maximum averaging is recommended, with a minimum of 16.

Calibration time

Clock 40 MHz. (AD_Clk == (bus_clock / 2)) (MCR[ADCLKSE] = 0)

Average samples (MCR[NRSMPL])

- 16–327 µs
- 32–654 µs
- 128–2.612 ms
- 512–10.446 ms

The calibration status is captured by status flag (MSR[CALFAIL]) to indicate if any fault is detected during calibration execution, or if the calibration has aborted at a premature state (calibration is aborted if any normal conversion is initiated during the execution of calibration, which is called premature termination). MSR[CALFAIL] = 1 means a calibration fail and a determined calibration value is out of internal defined limits. It also means that the ADC health may not be good and may not meet specifications.

The results of individual steps are also updated in CALSTAT register (CALSTAT[STAT_*n*]). The result of the last failed step is dynamically updated in the same register.

To execute the calibration algorithm, perform the following steps:

1. Wait for deassertion of both calibration and functional reset.
2. Configure desired parameters for calibration in MCR:
   - MCR[TSAMP]
   - MCR[NRSMPL]
   - MCR[AVGEN]
   - MCR[ADCLKSE]
3. Set the ADC internal operating clock to 40 MHz (MCR[ADCLKSE] = 0). (This setting assumes that ADC input clock is set to 80 MHz.)
4. Exit power-down state (MCR[PWDN] = 0).
5. Start calibration (MCR[CALSTART] = 1). Calibration starts immediately.
6. Check the status of MSR[CALBUSY] periodically. Wait until it becomes zero (alternatively, MSR[ADCSTATUS] can be monitored until it indicates an idle condition).

7. Check the fail flag (MSR[CALFAIL]) to learn the final status. If it is set to one, then calibration failed.
8. Check the pass/fail status of each calibration step in the CALSTAT register to look for failures.
9. At the end, check the status of MSR[CALIBRTD]. If calibration is successful this bit sets to one.

58.5.16 Self-test

For safety applications, it is important to check at regular intervals whether the SAR_ADC is operating correctly. For this purpose, a self-test feature is provided. When self-test is enabled, SAR_ADC automatically checks its components and flags any errors it finds.

The test can be enabled to check the supply voltage (V_{DD}), reference voltage (V_{refH}), and calibrated values.

The following test algorithms have been implemented:

- Supply self-test (algorithm S): It includes the conversion of the band gap, supply, and VREF voltages. It includes a sequence of 'num_supply_steps' test conversions (steps). The supply test conversions must be an atomic operation (that is, all supply algorithm conversions must be performed one after another with no functional conversions in between).
- Capacitive self-test (algorithm C): It includes a sequence of test steps per definition of algorithm C (See Table 526) which executes the capacitive matrix of the CDAC used for sampling and conversion.

Individual steps can take up to 1 µs at 80 MHz SAR_ADC clock frequency.

Table 526. Self-test steps

STCR3[ALG]	STCR3 [Step] (per algorithm definition)	Description	Outcome	Comment
'00'(algorithm S)	0	Supply self-test (band gap voltage)	Measures the internal band gap voltage	Band gap is considered as a constant with very little variation across temperature and process.
	1	Supply self-test (analog supply)	Measures the analog supply for the SAR_ADC	
	2	Supply self-test (reference voltage high)	Measures the high reference voltage of the SAR_ADC	
'01' (Reserved)	Reserved			
'10'(algorithm C)	0–11	Capacitive self-test: One of the calibration steps is being run	The difference/error of individual offset value from the previously calibrated value is being returned as SAR_ADC result that is compared with programmed value in analog watch dog register(s) to detect any fault. The difference or error may be caused by any runtime fault, runtime accuracy shift, or by device noise.	The result value should be compared with analog watchdog values. See Self-test analog watchdog for more detail.

Table continues on the next page...

Table 526. Self-test steps (continued)

STCR3[ALG]	STCR3 [Step] (per algorithm definition)	Description	Outcome	Comment
'11'(algorithm S+C)	0–2 (S) 0–11 (C)	Supply for One-Shot and (Supply + Capacitive) for Scan mode only	Same as above	

In case some failure occurs in any step of the algorithm, ADC: Schedules self-testing algorithms using configuration registers Monitors the conversion result using analog watchdog registers Flags the error to a fault control unit (FCCU), if it exists.

Self-test steps can be activated from software (CPU) or CTU.

ADC mode	TEST algorithm (CPU)	TEST algorithm / step(CTU)
CPU mode (MCR.CTUEN = 0)	Yes • One-Shot mode • Scan mode	No
CTU mode	No	Yes

NOTE

ADC self-test should be run with MCR[ADCLKSE] bit set to 1. Self-test with ADCLKSE bit set to 0 can give erroneous results.

58.5.16.1 CPU mode

In this case, test conversion works similar to normal conversion. The self-test is enabled by setting STCR2[EN].

The self-test conversions are run with the functional conversions. The sequencing of steps of the selected algorithm depends on the operating mode of normal conversions as selected by MCR[MODE].

In **One-Shot mode**, if self-test is enabled, only one step of the selected self-testing algorithm is executed at the end of the chain. The step number and algorithm to be executed are programmed in STCR3. So, in One-Shot mode the sequence is as follows:

1. Program NCMR*n* to select channels to be converted for normal conversion.
2. Program MCR[MODE] = 0 to select One-Shot mode.
3. Program sampling duration values in STCR1[INPSAMP*n*] field. (x = S,C).
4. Select the self-testing algorithm in STCR3[ALG] and step STCR3[MSTEP]. Default is Algorithm S step 0.
5. Enable self-testing by setting STCR2[EN].
6. Start the normal conversion by setting MCR[NSTART].
7. All normal conversions are executed as usual.
8. At the end of all the normal conversions in the chain, step number programmed in STCR3[MSTEP] field of self-testing algorithm selected by STCR3[ALG] is executed, similar to a normal functional channel.
9. On the receiving end of the conversion for test channel, the result is written in STDR1[TCDATA], and STDR1[VALID] is set. ISR[EOC], ISR[ECH] are set, and so is STSR1[ST_EOC].
10. ADC returns to Idle state.

For example: Channels A-B-C-D-E-F-G-H are present in the device where channels B-D-E are to be converted in One-Shot mode. At the end of conversion for channels B-D-E, self-test conversion is done and ISR[ECH] and ISR[EOC] are set. The sequence is B→D→E→self-test step.

MSR[NSTART] is automatically set when the normal conversion starts and is reset at the end of conversion for test channel.

In **Scan mode**, consecutive steps of the selected self-test algorithm are converted continuously at the end of each chain of normal conversions. The number of channels converted at the end of each chain is one (except for Algorithm S, in which all the steps are performed at once without any functional conversion interleaved). So, in Scan mode the sequence is as follows.

1. Program NCMRx to select channels to be converted for normal conversion.
2. Program MCR[MODE] = 1 to select Scan mode.
3. Select the self-testing algorithm in STCR3[ALG]. By default, all algorithms (supply and capacitive) are selected that is, all algorithms will be executed step-by-step one after the other.
4. Enable self-testing by setting STCR2[EN].
5. Start the normal conversion by setting MCR[NSTART].
6. All normal conversions are executed as usual.
7. At the end of chain of the normal conversions (assuming default value of STCR3[ALG]) all steps of Algorithm S are performed (as Algorithm S is always atomic). MSR[SELF_TEST_S] is set.
8. On receiving end of conversion of test channel for last step of Algorithm S:
   - The result is written in STDR1[TCDATA] and STDR1[VALID] is set.
   - At the same time, MSR[SELF_TEST_S] is reset.
   - For Step 1, the integral part and fractional part are written in STDR2[IDATA] and STDR2[FDATA].
   - Also, EOC and ECH bits are set in ISR register and ST_EOC is set in STSR2.
9. The next chain of normal conversion then starts.
10. At end of the normal conversion chain, Step0 of algorithm C is executed.
11. On the receiving end of conversion of test channel for Step0 of algorithm C:
    - The result is stored in STDR1[TCDATA] and STDR1[VALID] is set (if MCR[OVERWR] is set).
    - Also, EOC and ECH bits are set in ISR register and ST_EOC is set in STSR2.
12. Then the next chain of normal conversion starts.
13. At end of the normal conversion chain, Step1 of algorithm C is executed.
14. This process continues for all the steps of all three algorithms.
15. ADC returns to Idle state when MCR[NSTART] is written to 0.

For example: channels A-B-C-D-E-F-G-H are present in the device where channels B-D-E are to be converted in Scan mode. At the end of every conversion chain for channels B-D-E, one step of self-test conversion is done and ISR[ECH] and ISR[EOC] are set.

The sequence is:

1. B→D→E→ST-S0→ST-S1→ST-S2→B→D→E→ST-C0→B→D→E→ST-C1→B→D→E→ST-C2→ . . . B→D→E→ST-C11→B→D→E→ST-S0 . . .
2. ST-Sx - SelfTest Supply step x (x = 0,1,2)
3. ST-Cx - SelfTest Capacitive step x (x = 0 to 11)

MCR[NSTART] is automatically set when the normal conversion starts.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca4625c5-189c-4825-9251-390d3ab624c1

NOTE

In case of Injected Conversion modes, test channel conversion is not performed. It is performed only during normal conversions.

NOTE

If during a test channel conversion, injection conversion begins, then the test conversion is aborted (as with a normal functional channel) and Injected Conversion modes are completed. After Injected Conversion modes are completed, the test conversion resumes from the step at which it was aborted. In this case, MSR[SELF_TEST_S] remains high during Injected Conversion mode.

NOTE

For self-testing, the mode bit must be programmed at least one cycle before setting MCR[NSTART], and should not be changed thereafter until the conversion is finished or terminated.

58.5.16.2 CTU mode

The CTU mode is enabled by setting MCR[CTUEN]. The CTU operating mode selection if available, can be done through MCR[CTU_MODE]. *(Please check device configuration for availability.)*

The test conversion behaves in the same way for all CTU operating modes (Trigger and Control). In both cases, if MCR[CTUEN] is set, the test conversion can be started only by CTU interface–software cannot start it. Do not change MCR[CTUEN] during normal conversion. The interface between CTU and ADC (for Self-test) is shown in the following figure.

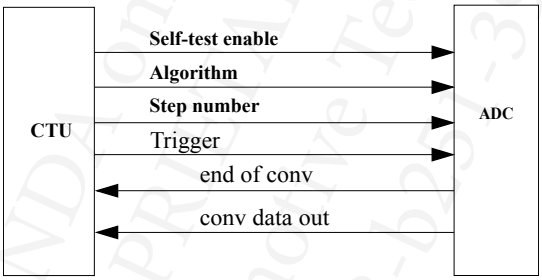


Figure 539. Self-test interface between ADC and CTU

For self-testing conversions in CTU mode, CTU asserts self-test enable along with trigger. The algorithm and the step number to be executed are put on Algorithm and Step number, respectively. The other command triggers, end of conversion and conversion data out has the same meaning as in normal CTU functional conversions.

As already mentioned for algorithm S, the three steps must be atomic. In CTU mode, CTU managed this itself, that is CTU has to send three triggers (one for each step) asserting self-test enable and updating test_step for each step.

58.5.16.3 Abort and abort chain for self-testing channel

Setting MCR[ABORT] during self-test channel conversion has no effect.

In One-Shot Operation mode, if MCR[ABORTCHAIN] is set when test channel conversion is ongoing, the test channel conversion is aborted and ISR[ECH] is set. In this case, ISR[EOC] for test channel is not generated.

In Scan Operation mode, if MCR[ABORTCHAIN] is set when test channel Step n is ongoing, the test channel Step n is aborted and next the chain conversion starts. At the end of this chain, Step n conversion is performed again. (In the case of Algorithm S, the full algorithm is executed again).

58.5.16.4 Self-test analog watchdog

The ADC digital interface also provides monitors for the conversion data for self-test algorithms. The analog watchdogs are used to determine whether the result of conversion for self-test algorithms is in the range of a particular guard area. For this purpose, separate self-test analog watchdog registers have been provided for each algorithm.

After the conversion of each step of an algorithm, a comparison is performed between the converted value and the threshold values if the analog watchdog feature is enabled by setting STAW*n*R[AWDE]. If the converted value does not lie between the upper and lower threshold values specified by Analog Watchdog Register of the particular algorithm, corresponding error bit STSR1[ERR_S*n*] is set and the step number in which the error occurred is updated in STSR1[STEP_C] (in case of algorithm C). Also, erroneous data is written in the STSR4[DATA*n*] field. The STSR1[ERR_S*n*] fields generate an interrupt if enabled by the corresponding mask bit in STCR2. The fault indication is also given to the fault control unit, if present, through critical and non-critical fault lines, so that necessary action can be taken at the chip level. Configuration in STCR2 register decides the choice between critical and noncritical type.

The analog watchdog feature works differently in algorithm S. As already mentioned, algorithm S is always an atomic operation. So, separate error bits are provided in STSR1 for each Step of algorithm S to avoid overwriting if error occurs in more than one Step. Hence, there are separate mask bits for each step in STCR1. For the same reason, separate fields exist in status registers (STSR2 and STSR3) to store erroneous data for each step.

In step0 of the supply algorithm, ADC measures the band gap voltage (+1.15 V), which is assumed to be stable. The conversion result of Step0 is measured with high (THRH) and low (THRL) thresholds defined in STAW0R if enabled (STAW0R[AWDE] = 1). STSR1[ERR_S0] is set to one if either threshold is violated. After set, it is cleared by writing a one to it.

For algorithm S Step1, a fixed-point division is performed, which takes about 26 cycles after EOC for Step1 to receive the divided value. It is used for analog watchdog checks. So, the value to be compared for Step1 contains the integer as well as fractional part—thus, two registers (STAW1AR and STAW1BR) are provided.

The comparison is done first for the integer part using the threshold values programmed in STAW1AR. If the integer part lies in the range, the fractional part comparison is skipped; otherwise it is compared to the values programmed in STAW1BR. The following table summarizes this feature for Step1.

Table 527. Algorithm S (Step1) threshold comparison

STDR2[IDATA] (integer part)	STDR2[FDATA] (fractional part)	STSR1[ERR_S1]
> STAW1AR[THRH]	Any value	Set
< STAW1AR[THRL]	Any value	Set
== STAW1AR[THRH]	> STAW1BR[THRH]	Set
== STAW1AR[THRL]	< STAW1BR[THRL]	Set

For algorithm S Step2, VREF/VREF is measured in order to check the integrity of the sampling signal. For this particular conversion, no higher threshold value is required, as the ideal value is FFFh. Only the lower threshold value is programmed in STAW2R.

For algorithm C, a separate register is provided for Step0. In Step0, an offset for other steps is measured. The converted data is compared with the threshold values provided by STAW5R if STAW4R[AWDE] is set. For other steps, this offset is subtracted from converted data before performing watchdog checks.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CSI Automotive Technology Co., Ltd.
c8c462e5-4815-48a8-b251-3900320...

NOTE

The recommended analog watchdog threshold values are set according to the expected and tested results with a clean environment (noise introduction from external environment minimized) for ADC operation. In a noisier setup, running the self-test with these analog watchdog values may cause failures.

To overcome the failures, you can relax the threshold values if the needs of the application can tolerate the existing noise level. The more the threshold values are relaxed, the more the magnitude of true error that can pass. For example, looking at the values for the STAW4R and STAW5R registers, THRH of 010h (16d) and THRL of FF0h (–16d), means that the algorithm C fails for an error of magnitude greater than (16/8 = 2) 2LSB@12b. This is the recommended value for a clean environment.

Similarly, THRH of 020h (32d) and THRL of FE0h (–32d) means that the algorithm C fails for an error of a magnitude greater than 4LSB@12b. Thus to pass the self test in a noisy environment where the application can live with the noise level, the thresholds can be relaxed from ±40 to ±64 or as needed. However, it is preferable that the environment is cleaned up to match the expectation for better operation instead of relaxing the thresholds. Relaxing the thresholds too much (more than ±64) leads to a false pass, and is not recommended for safety applications.

Relaxation of the self-test thresholds due to noise does not directly relate to normal use conversions, and it is not possible to tell what the noise impact to those measurements is with respect to the threshold relaxation. The noise is application-specific and not under control/judgment of the chip. In this scenario, not passing with the default threshold values is caused by external noise and not by the constituent capacitors. Limits are relaxed knowing that there is noise in the system that rides on top of the actual capacitor error, and the magnitude of the noise component is so high that limits should be relaxed as a result. Thus in reality whenever tests are run environment noise will be there, and if there is any true shift of the capacitor error noise it will be added to this environment noise, and should result in a violation of the relaxed threshold limits. The self-test is designed to catch and flag runtime and catastrophic errors. It is not able to, and not meant to, test for specification limits.

58.5.16.5 S1 algorithm: calculated voltage comparison method

The software can implement a method of calculating S1 algorithm STDR2[IDATA], STDR2[FDATA], THRHLRn[THRH] and THRHLRn[THRL] voltage values, then perform a comparison to the calculated THRH and THRL voltages.

The steps involved in this method are as follows:

1. Calculate the high threshold voltage:

   (STAW1AR[THRH] + (STAW1BR[THRH] / 4096)) × $V_{BANDGAP}$

2. Calculate the low threshold voltage:

   (STAW1AR[THRL] + (STAW1BR[THRL] / 4096)) × $V_{BANDGAP}$

3. Calculate the converted results to voltage:

   (STDR2[IDATA] + (STDR2[FDATA] / 4096)) × $V_{BANDGAP}$

4. Compare result from step 3 to THRH and THRL. S1 algorithm passes if low threshold voltage ≤ converted voltage ≤ high threshold voltage.

58.5.16.6 Watchdog timer

The watchdog timer is an additional check which monitors the sequence of the self-testing algorithm implemented. It also verifies that the algorithm is completed within a safe time period. The watchdog timers can be enabled for CPU as well as CTU conversions.

Each algorithm has a different watchdog timer which runs independently of the other. The watchdog timer for a particular algorithm can be enabled by setting STAW*n*R[WDTE]. The safe time value can be programmed in STBRR[WDT] (default value is 10 ms assuming an 80 MHz clock).

The safe time is measured starting from Step0 of the algorithm (including all normal chain conversions in between) to the point where Step0 of the same algorithm starts again.

The programming sequence is as follows:

1. Program NCMR*n* to select the channels on which conversion needs to be performed for normal conversion in Scan Operation mode (MCR[MODE] = 1).
2. Select the self-testing algorithm in STCR3[ALG]. By default, all algorithms (supply and capacitive) are selected, that is, all algorithms will be executed step-by-step one after the other.
3. Enable self-testing channel by setting STCR2[EN].
4. Program safe period value in STBRR[WDT].
5. Enable watchdog timer by setting STAW*n*R[WDTE]. Assume setting of STAW*n*R[WDTE] to be 't0'. It is important to do all the programming first and then enable STAWnR[WDTE], as the safe time check is also performed between setting STAWnR[WDTE] and the start of Step0. This step checks that the algorithm has started within the safe time.
6. Start the normal conversion by setting MCR[NSTART].
7. At the end of the first conversion chain, three steps (Step 0/1/2) of algorithm S are executed in sequence. Let us assume the start of Step0 to be 't1'.
8. After completion of algorithm S, conversion of the next chain is performed.
9. At the end of the conversion chain, the first Step (Step0) of algorithm C is performed.
10. In this way, one Step of algorithm C is performed after completion of one conversion chain until all the steps have been completed. Steps are executed in sequence (Step 0, 1, 2, . . 11).
11. After the last step of algorithm C, another chain conversion is executed. At the end of this chain conversion, Step0 of algorithm S is started again, thus repeating the whole sequence. Let's assume this time (starting of Step0 of Supply algorithm) to be 't2'.
12. For S algorithm, if (t1–t0) > Safe Period or (t2–t1) > Safe Period, the watchdog timer flags an error and STSR1[WDTERR] is set. A fault is asserted on the specified module port as per criticality configuration (Critical or Non_critical), and an interrupt is also generated if enabled by STCR2[MSKWDTERR]. Otherwise, the watchdog timer counter is reset and starts again to monitor the same for the next sequence.
13. A similar sequence is followed for watchdog timers for algorithm C.

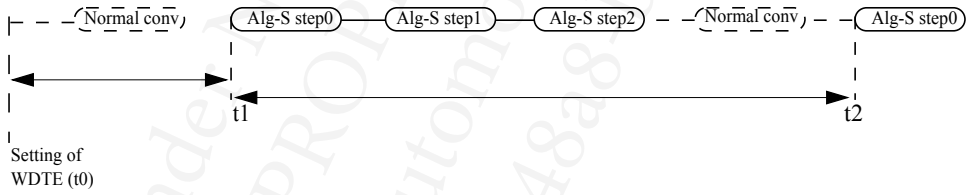


Figure 540. Watchdog timer monitor for Alg-S

NOTE

As CTU may not incorporate any safe period checking mechanism, the watchdog timers can be enabled for CTU conversions also.

NOTE

You must ensure that watchdog timers are disabled for self-test algorithms which are not executed.

58.5.16.6.1 Watchdog sequence checking

The watchdog timer also incorporates sequence checking features that check that the steps of a particular algorithm are in correct order. If steps are not in correct order, an error is flagged by setting STSR1[WDSERR]. A fault is indicated by an interrupt flag if enabled by STCR2[MSKWDSERR]. A fault is also asserted on the specified module port as per criticality configuration (Critical or Non_critical) and an interrupt generated if enabled by STCR2[MSKWDSERR].

A watchdog sequence error is flagged in the following cases:

- If steps of any algorithm are not executed in proper order.

- If abort chain occurs during test channel conversion. That step has to be repeated at the end of the next chain. This gives a sequence error when test channel conversion starts again.
  - Exception: If abort chain occurs during the last step of Algorithm S, then a sequence error is not flagged as the whole algorithm has to be repeated again.
- If, for CTU conversions, step numbers provided by CTU are not in order. Watchdog sequence checking is significant for CTU Burst mode only.

If injected conversion occurs during the test channel conversion, a watchdog sequence error is NOT flagged although the ongoing step number is aborted and is repeated again.

NOTE

Watchdog timer feature is applicable only for Scan Operation mode and not for One-Shot Operation mode.

58.5.16.7 Baud rate control for test channel

Baud rate control defines the scheduling of the test channel between the normal conversion chain. STBRR[BR] specify the scheduling rate.

By default, if a test channel is enabled, one step of the selected algorithm is executed after every chain of normal conversion. The bandwidth consumed by a test channel depends on the number of channels in a normal chain. For example, if we have 50 normal conversions in a chain, then a test channel consumes only 2% of total bandwidth. But if the number decreases to just 3 channels, then the bandwidth consumed by a test channel is 25%, which is significant (and may not be desirable as it slows down the normal conversion rate).

STBRR[BR] provides flexibility by scheduling the test channel conversion to be performed not at the end of every chain, but at the end of BR+1 number of chains. For example, if BR = 5, a single step of the selected algorithm for a test channel is performed after six chain conversions, then the next step is performed at the end of the next six chain conversions, and so on. By default, the value of BR is zero.

NOTE

This feature is applicable only for Scan Operation mode of operation and not for One-Shot Operation mode. STBRR[BR] should be set to zero for One-Shot Operation mode.

To use the baud rate feature in Scan Operation mode, NCMR should have a non-zero value.

58.5.16.7.1 Abort chain when baud rate is non-zero

As already described, for a non-zero value of STBRR[BR], the test channel conversion is performed at the end of (BR+1) number of chains. If an abort chain occurs during the chain in which test conversion is scheduled to be converted, then test conversion is done after the next (BR+1) number of chains.

For example, if STBRR[BR] is programmed to two, the sequence will be two normal chains (without any test channel conversion) followed by a chain with test channel converted at the end. Now, if an abort chain occurs during the first two chains, it is treated as a normal chain abort and test channel is converted at the end of the third chain only (as is the case without any abort chain). But if an abort chain occurs during the third chain in which the test channel is scheduled to be converted, then the test channel is converted after the next three chains, that is at the end of sixth chain (counting from the beginning).

58.5.17 Conversion time

Total conversion time depends on the SAR_ADC clock (AD_CLK). This clock can be prescaled to half or one-fourth of bus clock. The SAR_ADC operating clock (AD_CLK) is always synchronous with the bus clock. The frequency of AD_CLK depends on MCR[ADCLKSE] settings.

- When MCR[ADCLKSE] = 1, AD_CLK = bus clock.
- When MCR[ADCLKSE] = 0, AD_CLK = (bus clock / 2).

Conversion timing detail is shown in the figure below. The description of each time segment follows the figure.

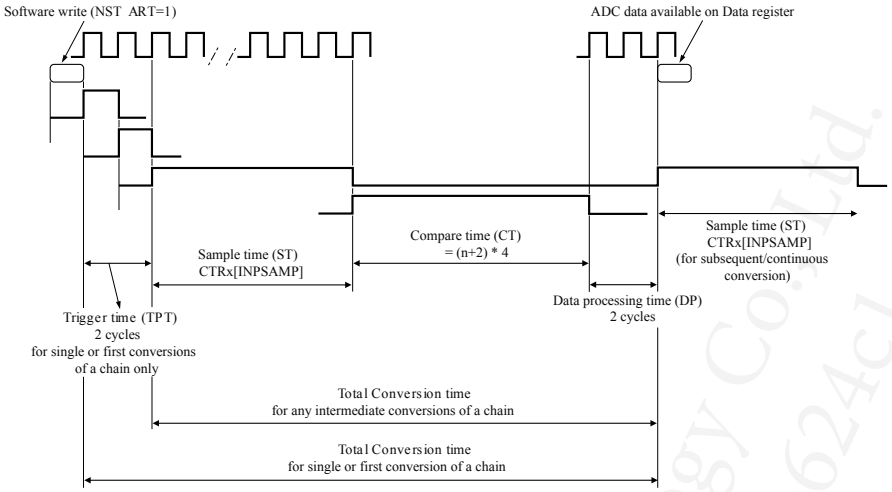


Figure 541. Conversion time

Trigger time (TPT):

To prepare a channel for conversion, two bus clock cycles are required by the SAR_ADC for the single conversion and for the first conversion in a chain. For continuous conversions and subsequent conversions in a chain, this time is not required because it is hidden in the pipeline operation.

Triggers from the synchronous CTU interface require two cycles of bus clock for the first conversion, but for back-to-back CTU conversions it takes only one cycle for trigger processing, because the other cycle is hidden in the pipeline operation.

Sample phase time (ST):

ST duration is controlled by the CTR*n*[INPSAMP[7:0]] configuration. The value in the register field represents units of AD_CLK cycles. The minimum value of sample time is eight AD_CLK cycles. If the value programmed in CTR*n*[INPSAMP[7:0]] is less than eight, it has no effect on sample time duration and will default to eight AD_CLK cycles.

Compare phase time (CT):

CT depends on the resolution of the SAR_ADC. The CT is ((x + 2) × 4) cycles of AD_CLK, where x is the resolution of SAR_ADC.

Data processing time (DP):

DP takes two AD_CLK cycles to post and load data into the data register.

Total conversion time:

The total time required for a single or first conversion is (TPT + ST + CT + DP) AD_CLK cycles. The time for subsequent conversions in a chain or continuous conversions is (ST + CT + DP) AD_CLK cycles.

Examples:

- CTR*n*[INPSAMP] = 20d (software triggered, MCR[NSTART]):
  - Single or first conversion: 2 + 20 + ((12 + 2) × 4) + 2 = 2 + 20 + 56 +2 = 80 AD_CLK cycles
  - Subsequent or continuous conversions: 20 + ((12 + 2)× 4) + 2 = 20 + 56 + 2 = 78 AD_CLK cycles
- CTR*n*[INPSAMP] = 20d (CTU triggered):
  - Single or first conversion: 2 + 20 + ((12 + 2)× 4) + 2 = 2 + 20 + 56 + 2 = 80 AD_CLK cycles
  - Subsequent back-to-back conversions: 1 + 20 +((12 + 2)× 4) + 2 = 1 + 20 + 56 + 2 = 79 AD_CLK cycles

Presample phase time (PST):

When presampling is enabled, the SAR_ADC presamples prior to starting the actual sampling, as described in Presampling. This phase takes the number of AD_CLK cycles defined in CTR*n*[INPSAMP]. Additionally, to switch from presample phase to actual channel sampling phase, it takes another two AD_CLK cycles. So, the total presampling AD_CLK cycle is CTR*n*[INPSAMP] + 2

additional cycles that are required for a conversion to complete. If presampling is implemented, all equations and examples above are valid only with the additional (CTRN*n*[INPSAMP] + 2) AD_CLK cycles.

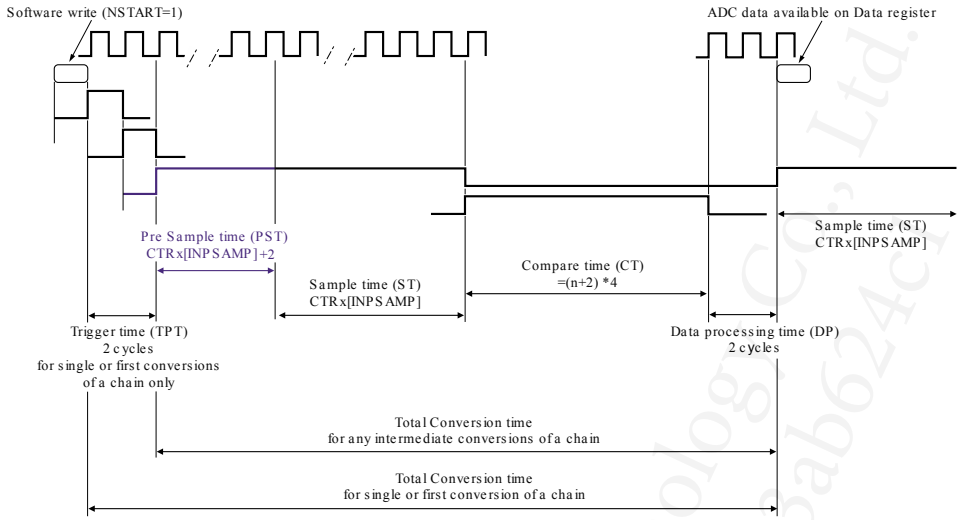


Figure 542. Conversion timing with presampling

Conversion time for calibration:

- Single conversion time = Sample time (MCR[TSAMP]) + CT (11 × 4) + DP (2)
- Interconversion gap = 1 cycle
- Total number of tests = 12
- Averaging = 32

One test duration = (single conversion time + interconversion gap) × averaging samples

Total test duration = (one test duration × total number of tests)

Example: Sample time = 22

Total test time = ((((22 + (11× 4) + 2) + 1) × 32) × 12) = 26496 cycles × 25 ns (40 MHz) = 662.4 µs

Conversion time for self-test conversion:

The time for a self-test conversion is one test period multiplied by the number of consecutive (atomic) steps. For supply self-tests, the atomic step is three and for other tests it is one.

One test period for self-test is:

- For supply tests: (ST + (14 × 4) + 2)
- For other algorithms: (ST + (11 × 4) + 2)

NOTE

Steps (conversions) of the supply algorithm always run in atomic operation, required for algorithmic procedure. So, in Scan Operation mode when all algorithms are selected, three steps (conversions) of supply algorithm run together after end of the chain. Steps (conversions) of algorithm C are not required to be run together, so after every chain one step (conversion) is executed serially. Conversion time in self-test also needs to be calculated accordingly.

Example:

- Channels in chain A→B→C
- Scan Operation mode

- All self-test algorithms selected

The resulting conversion order will be:

[A→B→C→AS_step0→AS_step1→AS_step2→A→B→C→AC_step0→A→B→C→AC_step1→A→B→C→AC_step2→ .....→A→B→C→AC_step11] →[same sequence repeats]

NOTE

AS = Algorithm S (supply self-test)

AC = Algorithm C (capacitive self-test)

58.5.18 Conversion data processing

The raw converted data of the SAR_ADC contains many types of errors like offset, gain, and so on. In order to reduce or eliminate errors, the raw conversion data is processed inside the SAR_ADC controller before sending to result register/output. The process of error correction happens in parallel during bit-by-bit evaluation, with the correction values generated during offset determination and calibration process.

The error correction value is subtracted from the raw conversion result whenever the comparator output for one of the calibrated capacitors evaluates high during the conversion period.

In order to reduce data processing time, the error correction is done in parallel during each bit evaluation of a conversion. All lower-weighted capacitors are not calibrated and no error correction is performed for these capacitors, but they are self-tested to guarantee error contribution is below specified accuracy limits.

The final result is checked for any overflow or underflow. If the processed data is above the maximum value that can be represented by ADC resolution, then the output data is forced to all ones (FFFh for 12-bit resolution). Similarly, if the processed data is negative, then output data is forced to all zeros (000h for 12-bit resolution).

58.5.19 User defined offset and gain values

Offset and gain errors in ADC are compensated by its calibration mechanism. However, there could be a source of errors outside the ADC module (for example, at system and/or board level). These errors can potentially contribute to ADC static errors and lower the overall TUE observed when an application runs. To compensate for these errors, there is a set of user-programmable offset and gain values in the User OFFSET and Gain Register (USROFSGN). By programming this register appropriately, the complete system's TUE can be enhanced (calibrated), when impacted by errors other than ADC itself.

These values (GAIN_USER and OFFSET_USER) must be stored in 2's complement format as described in the register section (see User OFFSET and Gain Register (USROFSGN)). The values stored here are being considered by the SAR_ADC engine and being computed/adjusted for every output. Using this register to compensate offset and gain is described below.

58.5.19.1 Writing for 12-bit ADC

58.5.19.1.1 User gain compensation (GAIN_USER)

The ADC can compensate a maximum of ±63 LSB @12b gain by programming GAIN_USER field in USROFSGN register. The field is in 2's complement format and stored with an 8x multiplication. Thus, if a value of ± Y is stored in this field, it leads to a final result adjusted with ±(Y/8) LSB from its unprogrammed version. It is required to keep the 3-LSBs zero to avoid any fractional subtraction/multiplication related errors.

ADC_OUTPUT_NEW = ADC_OUTPUT_OLD × (1 + (GAIN_USER / (8 × 4096))).

Example:

For GAIN_USER = 0, result is ADC_OUTPUT_NEW = ADC_OUTPUT_OLD = P

For GAIN _USER = +Y, result is ADC_OUTPUT_NEW = P × (1 + (Y / 32768))

For GAIN _USER = −Y, result is ADC_OUTPUT_NEW = P × (1 − (Y / 32768))

58.5.19.1.2 User offset compensation (OFFSET_USER)

The ADC can compensate a maximum of ±15LSB @12b offset by programming OFFSET_USER field in USROFSGN register. The field is in 2's complement format and stored with an 8x multiplication. Thus, if a value of ±X is stored in this field, it leads to a final result adjusted with ±(X/8) from its unprogrammed version. It is required to keep the 3-LSBs zero to avoid any fractional subtraction-related errors.

ADC_OUTPUT_NEW = ADC_OUTPUT_OLD − (OFFSET_USER/8)

Example:

For OFFSET_USER = 0, result is ADC_OUTPUT_NEW = ADC_OUTPUT_OLD = K

For OFFSET_USER = +X, result is ADC_OUTPUT_NEW = K − (X / 8)

For OFFSET_USER = −X, result is ADC_OUTPUT_NEW = K − (−X / 8) = K + (X / 8)

58.5.19.1.3 Using OFFSET_USER and GAIN_USER together

When both the fields are programmed together, gain is computed first and then offset is computed.

ADC_OUTPUT_NEW = (ADC_OUTPUT_OLD × (1 + (GAIN_USER / (8 × 4096)))) − ((OFFSET_USER / 8)).

Example:

For OFFSET_USER = 0 and GAIN_USER = 0, result is ADC_OUTPUT_NEW = ADC_OUTPUT_OLD = M

For OFFSET_USER = X and GAIN_USER = Y, result is ADC_OUTPUT_NEW = (M x (1 + (Y / 32768))) − (X / 8)

NOTE

Negative values are stored in 2's complement in these fields. Example:

GAIN_USER = 1F0h means +496d, and 210h means −496d

OFFSET _USER = 70h means +112d, and 90h means −112d

58.6 Programming sequences

ADC must be programmed properly to function correctly. This section describes the basic programming sequences for applications to use. Application software may use them in combination by sequencing them carefully.

58.6.1 Running calibration

Calibration is the first step that should be run after every power-on reset. ADC has two types of reset:

- Calibration
- Functional

Calibration reset is used for registers that store internal calibration settings and values. This reset is done to store them during functional reset. Functional reset may be applied many times during an application for various reasons and calibration may not required each time, so storing the settings and values in a higher level of reset avoids recalibration with every functional reset.

1. Wait for deassertion of functional reset.
2. Configure SAR controller operating clock (MCR[ADCLKSE] = 0).
3. Bring ADC out of Power-down state (MCR[PWDN] = 0).
4. Configure desired calibration settings (default values kept for highest accuracy maximum time).
   - MCR[TSAMP]: Sample time for calibration conversion
   - MCR[NRSMPL]: Number of samples in averaging
   - MCR[AVGEN]: Averaging function enable in calibration
5. Run calibration by writing a one to MCR[CALSTART].

6. Check calibration run status in MSR[CALBUSY]—wait until MSR[CALBUSY] = 0; alternatively, MSR[ADCSTAT] can be used to check status.
7. Check calibration pass/fail status in MSR[CALFAIL] field. If MSR[CALFAIL] = 1 then calibration failed. Detailed status can be checked in CALSTAT.

58.6.2 Running normal conversion

Normal conversion mode should usually be used to convert analog samples most of the time in an application, unless there is a special need. There are other modes available for special requirements as described in later sections of this document. Normal conversions can be initiated by software trigger. Calibration should be done before running any conversion to ensure an accurate result.

1. Configure channel mask registers NCMR*n*, for different channel groups.
2. Configure desired sample time in CTR*n*, for different channel groups.
3. Configure interrupt mask registers, IMR for global and CIMR*n*, for different channel groups.
4. Configure DMA registers, DMAE, and DMAR*n*, for different channel groups.
5. Configure pre-sampling registers, if needed, PSCR and PSR*n*, for different channel groups (if needed).
6. Configure watchdogs in WTIMR
7. Configure One-Shot or Scan mode in MCR[MODE] and other settings (MCR[WLSIDE], MCR[OWREN], and so on).
8. Start conversion by writing one to MCR[NSTART].
9. Check EOC status in ISR or wait for interrupt.
10. On interrupt read the data register, or DMA will read out data if configured.
11. Status of conversion for individual channel can be seen in CEOCFR*n*.
12. Clear the registers by writing one to the fields.
13. Steps 9 to 12 will continue in loop until conversion finishes or is terminated.

58.6.3 Running injected conversion

Injected conversion has a higher priority than normal conversion and it runs in One-Shot mode only. It can be started in idle condition or when normal conversion is in process. The configuration sequence is almost similar to normal conversion above, except for the registers discussed in the following steps.

1. Channel mask register JCMR*n*, is used for different channel groups.
2. Configure other channel attributes as done in normal conversion.
3. Start conversion by writing 1b to MCR[JSTART].
4. Check status of ISR[JEOC] or wait for interrupt.
5. Read out conversion data and clear interrupts as done in normal conversion.
6. Configure channel mask registers NCMR*n* for different channel groups.
7. Configure desired sample time in CTR*n* for different channel groups.
8. Configure interrupt mask registers, IMR for global and *n*, for different channel groups.
9. Configure DMA registers, DMAE and DMAR*n*, for different channel groups.
10. Configure presampling registers, if needed, PSCR and PSR*n*, for different channel groups (if needed).
11. Configure watchdogs in WTIMR.
12. Configure One-Shot or Scan mode in MCR[MODE] and other settings (MCR[WLSIDE], MCR[OWREN], and so on).
13. Start conversion by writing 1b to MCR[NSTART].

14. Check status of ISR[EOC] or wait for interrupt.
15. On interrupt, read the data register or DMA will read out data if configured.
16. Status of conversion for an individual channel can be seen in CEOCFR*n*.
17. Clear the registers by writing one to the bits.
18. Steps 9 to 12 will continue in loop until conversion finishes or is terminated.

58.6.4 Running CTU triggered conversion

CTU conversion has a higher priority than normal or injected conversion and runs only a single conversion. It can be started in the idle condition or when normal or injected conversion is in progress. Conversions initiated by the CTU (one channel per trigger) are processed immediately by suspending normal conversions and discarding any ongoing injected conversion chain. The configuration sequence is similar to normal conversion above except for the following steps:

1. Enable CTU conversion in MCR[CTUEN].
2. Start CTU trigger.
3. Check EOCTU status in ISR.

58.6.5 Running self-test

Self-test conversions run after a programmable number of normal conversion chains or by CTU trigger. To run normal conversions, additional configurations for self-test need to be implemented before starting normal conversion (as shown in Self-test). Also, you may need to configure the self-test watchdog, timer, and baud rate as described in Self-test analog watchdog.

58.7 SAR_ADC register descriptions

58.7.1 SAR_ADC memory map

ADC_0 base address: 401F_8000h

ADC_1 base address: 402E_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Main Configuration Register (MCR)	32	RW	0000_3901h
4h	Main Status Register (MSR)	32	W1C	0000_0001h
10h	Interrupt Status Register (ISR)	32	W1C	0000_0000h
14h	Channel Pending Register (CEOCFR0)	32	W1C	0000_0000h
18h	Channel Pending Register 1 (CEOCFR1)	32	W1C	0000_0000h
20h	Interrupt Mask Register (IMR)	32	RW	0000_0000h
24h	Channel Interrupt Mask Register 0 (CIMR0)	32	RW	0000_0000h
28h	Channel Interrupt Mask Register 1 (CIMR1)	32	RW	0000_0000h
30h	Watchdog Threshold Interrupt Status Register (WTISR)	32	W1C	0000_0000h
34h	Watchdog Threshold Interrupt Mask Register (WTIMR)	32	RW	0000_0000h
40h	DMAE Register (DMAE)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
44h	DMA Register 0 (DMAR0)	32	RW	0000_0000h
48h	DMA Register 1 (DMAR1)	32	RW	0000_0000h
60h	Analog Watchdog Threshold Register 0 (THRHLR0)	32	RW	0FFF_0000h
64h	Analog Watchdog Threshold Register 1 (THRHLR1)	32	RW	0FFF_0000h
68h	Analog Watchdog Threshold Register 2 (THRHLR2)	32	RW	0FFF_0000h
6Ch	Analog Watchdog Threshold Register 2 (THRHLR3)	32	RW	0FFF_0000h
80h	Presampling Control Register (PSCR)	32	RW	0000_0000h
84h	Presampling Register 0 (PSR0)	32	RW	0000_0000h
88h	Presampling Register 1 (PSR1)	32	RW	0000_0000h
94h	Conversion Timing Register 0 (CTR0)	32	RW	0000_0014h
98h	Conversion Timing Register 1 (CTR1)	32	RW	0000_0014h
A4h	Normal Conversion Mask Register (NCMR0)	32	RW	0000_0000h
A8h	Normal Conversion Mask Register (NCMR1)	32	RW	0000_0000h
B4h	Injected Conversion Mask Register (JCMR0)	32	RW	0000_0000h
B8h	Injected Conversion Mask Register (JCMR1)	32	RW	0000_0000h
C0h	User OFFSET and Gain Register (USROFSGN)	32	RW	0000_0000h
C8h	Power Down Exit Delay Register (PDEDR)	32	RW	0000_0000h
100h - 11Ch	Precision Channel n Data Register (PCDR0 - PCDR7)	32	RO	0000_0000h
180h - 19Ch	Internal Channel n Data Register (ICDR32 - ICDR39)	32	RO	0000_0000h
280h	Analog Watchdog Threshold Register 4 (THRHLR4)	32	RW	0FFF_0000h
284h	Analog Watchdog Threshold Register 5 (THRHLR5)	32	RW	0FFF_0000h
288h	Analog Watchdog Threshold Register 6 (THRHLR6)	32	RW	0FFF_0000h
28Ch	Analog Watchdog Threshold Register 7 (THRHLR7)	32	RW	0FFF_0000h
2B0h	Channel Watchdog Select Register 0 (CWSELR0)	32	RW	0000_0000h
2C0h	Channel Watchdog Select Register 4 (CWSELR4)	32	RW	0000_0000h
2E0h	Channel Watchdog Enable Register (CWENR0)	32	RW	0000_0000h
2E4h	Channel Watchdog Enable Register (CWENR1)	32	RW	0000_0000h
2F0h	Analog Watchdog Out of Range Register (AWORR0)	32	W1C	0000_0000h
2F4h	Analog Watchdog Out of Range Register (AWORR1)	32	W1C	0000_0000h
340h	Self-Test Configuration Register 1 (STCR1)	32	RW	1800_A000h
344h	Self-Test Configuration Register 2 (STCR2)	32	RW	0000_0005h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
348h	Self-Test Configuration Register 3 (STCR3)	32	RW	0000_0300h
34Ch	Self-Test Baud Rate Register (STBRR)	32	RW	0005_0000h
350h	Self-Test Status Register 1 (STSR1)	32	W1C	0000_0000h
354h	Self-Test Status Register 2 (STSR2)	32	RO	0000_0000h
358h	Self-Test Status Register 3 (STSR3)	32	RO	0000_0000h
35Ch	Self-Test Status Register 4 (STSR4)	32	RO	0000_0000h
370h	Self-Test Data Register 1 (STDR1)	32	RO	0000_0000h
374h	Self-Test Data Register 2 (STDR2)	32	RO	0000_0000h
380h	Self-Test Analog Watchdog Register (STAW0R)	32	RW	0BA4_0992h
384h	Self-Test Analog Watchdog Register (STAW1AR)	32	RW	0001_0001h
388h	Self-Test Analog Watchdog Register 1B (STAW1BR)	32	RW	0B33_0586h
38Ch	Self-Test Analog Watchdog Register 2 (STAW2R)	32	RW	0000_0FF5h
390h	Self-Test Analog Watchdog Register 3 (STAW3R)	32	ROZ	0000_0000h
394h	Self-Test Analog Watchdog Register 4 (STAW4R)	32	RW	0028_0FD8h
398h	Self-Test Analog Watchdog Register 5 (STAW5R)	32	RW	0028_0FD8h
39Ch	Calibration Status register (CALSTAT)	32	RO	0000_0000h

58.7.2 Main Configuration Register (MCR)

Offset

Register	Offset
MCR	0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OWREN	WLSIDE	MODE	0	TRGEN	EDGE	XSTRTEN	NSTART	0	JTRGEN	JEDGE	JSTART	0		CTUEN	CTU_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STCL	CALSTART	AVGEN	NRSMPL		TSAMP		ADCLKSE	ABORTCH...	ABORT	ACKO	0	0			PWDN
W																
Reset	0	0	1	1	1	0	0	1	0	0	0	0	0	0	0	1

Fields

Field	Function
31 OWREN	Overwrite enable Sets whether the latest conversion overwrites current value in data registers. 0b - Older valid conversion data is not overwritten by newer conversion data 1b - Newer conversion result is always overwritten, irrespective of the validity of older conversion data
30 WLSIDE	Selects whether conversion data is left or right aligned. 0b - Write right-aligned conversion data (from 11 to 0) 1b - Write left-aligned conversion data (from 15 to 4)
29 MODE	Normal scan mode select Selects between One-Shot or Scan Operation modes; MODE must be changed in ADC's Idle condition (check MSR [ADCSTATUS] field) 0b - One-Shot Operation mode: configuration for one chain conversion in Normal Conversion mode 1b - Scan Operation mode: configuration for continuous chain conversion in Normal Conversion mode. When the chain conversion is finished a new scan starts immediately.
28 —	Reserved
27 TRGEN	External trigger enable Enables external triggers to start conversions. 0b - External trigger is disabled 1b - Enables the external trigger to start a conversion
26 EDGE	Trigger edge select Selects which edge of the external trigger starts a conversion. This field is only valid if MCR[TRGEN] = 1b. 0b - Falling edge is trigger 1b - Rising edge is trigger
25 XSTRTEN	External start enable If this field is set, a normal conversion starts when an external start signal is detected. This can be used to synchronize the start conversion events of two ADCs.
24 NSTART	Normal conversion start Setting this field starts One-Shot or Scan Operation mode. In Scan Operation mode, this field remains high after setting. Clearing this field in Scan Operation mode causes the current chain conversion to finish, then stop the scan.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	In One-shot Operation mode, this field is cleared immediately after the conversion starts. Setting this field during ongoing conversions keeps this field high until the requested conversion is started (or pending during Injected Operation mode).
23 —	Reserved
22 JTRGEN	Injection external trigger enable When set, enables the external trigger to start injected conversions. This field must be set if the external trigger feature is needed to start injected conversions. 0b - Injected conversion not started by external trigger 1b - Injected conversion started by external trigger
21 JEDGE	Injection trigger edge selection Selects which edge of the external trigger starts an injected conversion. This field is only valid if MCR[JTRGEN] = 1b. 0b - Falling edge is trigger 1b - Rising edge is trigger
20 JSTART	Start injection conversion Setting this field starts injected analog channels to be converted. Clearing this field has no effect since the injected chain conversion cannot be interrupted. If this field is set during on ongoing injected conversion, the field will remain set until the new conversion starts.
19-18 —	Reserved
17 CTUEN	Cross trigger unit enable This field enables conversions to starts by the CTU. **NOTE** This field must not be modified when CTU conversion is running (MSR[CTUSTART]= 1 b). 0b - The CTU is disabled and the triggered injected conversion cannot take place 1b - The CTU is enabled and the triggered injected conversion can take place
16 CTU_MODE	Cross trigger unit mode Sets the CTU operating mode of the SAR_ADC. This field can only be written in Power-Down mode. 0b - CTU control mode is selected 1b - CTU trigger mode is selected

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 STCL	Self-testing configuration lock 0b - Self-test registers are not locked 1b - The self-test configuration is locked (STCR1, STCR2, STCR3, STBRR, STAW0R, STAW1AR, STAW1BR, STAW2R, STAW3R, STAW4R, STAW5R are write-protected). It can be used only in CPU and Scan operation modes. It is cleared by reset only.
14 CALSTART	Calibration start Writing 1b to this field starts calibration. It is self-clearing, and is cleared by hardware when calibration completes or is terminated prematurely. 0b - No effect (default, return value) 1b - Start calibration
13 AVGEN	Average enable Allows averaging of calibration time. 0b - Disable 1b - Enable (default)
12-11 NRSMPL	Number of averaging samples Selects the number of averaging samples to be used during Calibration. 00b - 16 samples 01b - 32 samples 10b - 128 samples 11b - 512 samples
10-9 TSAMP	Sample period of calibration conversions 00b - 22 cycles of AD_CLK (default) 01b - 8 cycles of AD_CLK 10b - 16 cycle of AD_CLK 11b - 32 cycle of AD_CLK
8 ADCLKSE	Analog clock frequency select This field selects the clock frequency of the AD_CLK signal. It can be written in Power-Down mode only. 0b - AD_CLK frequency is half 1b - AD_CLK frequency is equal to bus clock frequency
7 ABORTCHAIN	Abort conversion chain If this field is set, the chain conversion in process is aborted. When the chain aborts, this field is reset by hardware.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Setting this field in idle state (for example, no normal/injected conversion is in process) has no effect. In this case, the field is reset immediately. During a CTU conversion, this field cannot be programmed. **NOTE** Abort and Abort Chain must not be applied simultaneously in SCAN mode operation. 0b - Chain conversion has been aborted, or chain conversion is not currently running. 1b - Abort current chain conversion.
6 ABORT	Abort conversion If this field is set during any ongoing conversion, it aborts that conversion and the next conversion starts. When the ongoing conversion aborts, this field is reset by hardware. This field is not written while the self-test channel conversion is in process. **NOTE** Abort and Abort Chain must not be applied simultaneously in SCAN mode operation.
5 ACKO	Auto-Clock-Off mode enable If this field is set, the Auto-Clock-Off feature is enabled. 0b - Auto-Clock-Off feature is disabled 1b - Auto-Clock-Off feature is enabled
4 —	Reserved
3-1 —	Reserved
0 PWDN	Power-down enable Request transitions to Power-Down or Idle modes. 0b - When ADC status is in Power-Down mode (MSR[ADCSTATUS] = 001b), start ADC transition to IDLE mode. 1b - Request to enter Power-Down mode.

58.7.3 Main Status Register (MSR)

Offset

Register	Offset
MSR	4h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-5481-48a8-b251-390d3b624c3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CALIBRTD	CALFAIL	CALBUSY	0				NSTART	JABORT	0		JSTART	0	SELF_TE...	Reserved	CTUSTART
W		W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHADDR								0		ACKO	0		ADCSTATUS		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Fields

Field	Function
31 CALIBRTD	Calibration status This field is updated after running high accuracy calibration (see Calibration). 0b - Uncalibrated or calibration unsuccessful 1b - Calibrated or calibration successful
30 CALFAIL	Calibration failed This field indicates the status of calibration. It is set automatically if any of the calibration step fails during the calibration process or calibration is aborted prematurely (by any normal conversion). Cleared by writing one, or on restarting calibration by writing one to MCR[CALSTART]. 0b - Calibration passed (must be checked with CALBUSY = 0b) 1b - Calibration failed
29 CALBUSY	Calibration busy This field indicates that the SAR_ADC is executing calibration. Set when MCR[CALSTART] is set. 0b - ADC is ready for use 1b - ADC is busy in a calibration process
28-25 —	Reserved
24 NSTART	Normal conversion status This status field is used to signal that a normal conversion is in process. 0b - Normal conversion is not in process 1b - Normal conversion is in process
23	Injected conversion abort status

Table continues on the next page...

Table continued from the previous page...

Field	Function
JABORT	This status field is used to signal that an injected conversion has been aborted. It is reset when a new injected conversion starts. 0b - Injected conversion has not been aborted 1b - Injected conversion has been aborted
22-21 —	Reserved
20 JSTART	Injected conversion status This field is used to show that an injected conversion is in process. 0b - Injected conversion is not in process 1b - Injected conversion is in process
19 —	Reserved
18 SELF_TEST_S	SELF_TEST_S signals that a self-test conversion is in process. 0b - Self-test conversion is not in process 1b - Self-test conversion is in process
17 —	Reserved
16 CTUSTART	CTUSTART is used to show that a CTU conversion is in process. It is set when a CTU trigger pulse is received and the CTU conversion starts. When CTU Trigger mode is enabled, this field is automatically reset when the conversion has completed. Otherwise, if Control mode is enabled, this field is reset when the CTU is disabled (MCR[CTUEN] = 0b).
15-9 CHADDR	Channel address This field shows the address of the channel that is currently being converted.
8-6 —	Reserved
5 ACKO	Auto-Clock-Off enable This status field is used to signal if the Auto-Clock-Off feature is enabled. 0b - Auto-Clock-Off feature is not enabled 1b - Auto-Clock-Off feature is enabled
4-3 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
2-0 ADCSTATUS	ADC status The table below describes how this field represents the ADC status. **Table 528. MSR[ADCSTATUS] field description**

ADCSTATUS field value	Description
000	IDLE
001	Power-down
100	Sample
110	Conversion
010	Wait state
011	Busy in calibration

58.7.4 Interrupt Status Register (ISR)

Offset

Register	Offset
ISR	10h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											EOCTU	JEOC	JECH	EOC	ECH
W												W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-5	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
4 EOCTU	End of CTU conversion Shows the status of the end of conversion for the CTU channel. 0b - CTU end of conversion has not occurred 1b - CTU end of conversion has occurred
3 JEOC	Injected channel end of conversion Shows end of conversion status for an injected channel. 0b - Injected channel end of conversion has not occurred 1b - Injected channel end of conversion has occurred
2 JECH	Injected end of conversion chain Shows end of conversion chain status for an injected channel. 0b - Injected channel end of conversion chain has not occurred 1b - Injected channel end of conversion chain has occurred
1 EOC	End of channel conversion Shows the digital end of conversion for a channel. 0b - Channel end of conversion has not occurred 1b - Channel end of conversion has occurred
0 ECH	End of conversion chain Shows the digital end of conversion chain for a channel. 0b - End of conversion chain has not occurred 1b - End of conversion chain has occurred

58.7.5 Channel Pending Register (CEOCFR0)

Offset

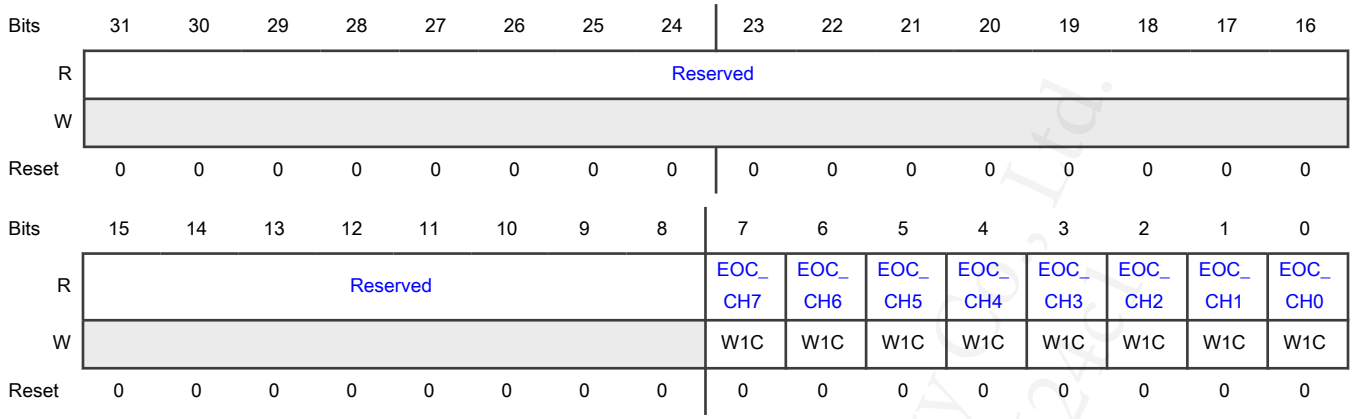
Register	Offset
CEOCFR0	14h

Function

End of conversion pending interrupt for SAR_ADC channels in Group0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c62e5-4821-43c8-b251-59d3ab624c1

Diagram



Fields

Field	Function
31-8 —	Reserved
7 EOC_CH7	Channel 7 conversion complete. 0b - Conversion not complete 1b - Conversion complete
6 EOC_CH6	Channel 6 conversion complete. 0b - Conversion not complete 1b - Conversion complete
5 EOC_CH5	Channel 5 conversion complete. 0b - Conversion not complete 1b - Conversion complete
4 EOC_CH4	Channel 4 conversion complete. 0b - Conversion not complete 1b - Conversion complete
3 EOC_CH3	Channel 3 conversion complete. 0b - Conversion not complete 1b - Conversion complete
2 EOC_CH2	Channel 2 conversion complete. 0b - Conversion not complete 1b - Conversion complete
1	Channel 1 conversion complete.

Table continues on the next page...

Table continued from the previous page...

Field	Function
EOC_CH1	0b - Conversion not complete 1b - Conversion complete
0 EOC_CH0	EOC_CH0 Channel 0 conversion complete. 0b - Conversion not complete 1b - Conversion complete

58.7.6 Channel Pending Register 1 (CEOCFR1)

Offset

Register	Offset
CEOCFR1	18h

Function

End of conversion pending interrupt for group32 channels.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								EOC_CH39	EOC_CH38	EOC_CH37	EOC_CH36	EOC_CH35	EOC_CH34	EOC_CH33	EOC_CH32
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-8 —	Reserved
7 EOC_CH39	Channel 39 conversion complete. 0b - Conversion not complete 1b - Conversion complete
6 EOC_CH38	Channel 38 conversion complete. 0b - Conversion not complete 1b - Conversion complete
5 EOC_CH37	Channel 37 conversion complete. 0b - Conversion not complete 1b - Conversion complete
4 EOC_CH36	Channel 36 conversion complete. 0b - Conversion not complete 1b - Conversion complete
3 EOC_CH35	Channel 35 conversion complete. 0b - Conversion not complete 1b - Conversion complete
2 EOC_CH34	Channel 34 conversion complete. 0b - Conversion not complete 1b - Conversion complete
1 EOC_CH33	Channel 33 conversion complete. 0b - Conversion not complete 1b - Conversion complete
0 EOC_CH32	Channel 32 conversion complete. 0b - Conversion not complete 1b - Conversion complete

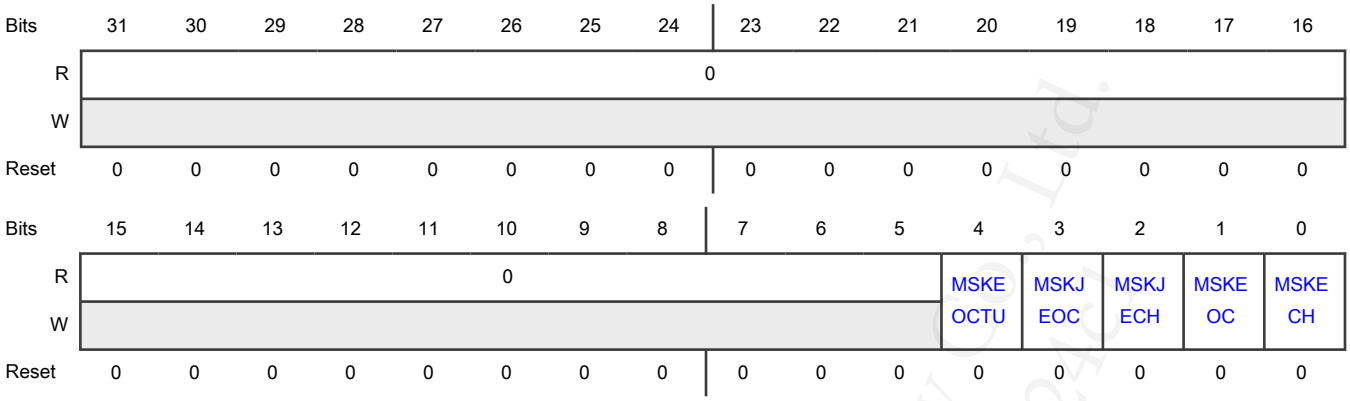
58.7.7 Interrupt Mask Register (IMR)

Offset

Register	Offset
IMR	20h

Provided by IHS Markit
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-5 —	Reserved
4 MSKEOCTU	End of CTU conversion interrupt mask When set, the EOCTU interrupt is enabled. 0b - EOCTU interrupt disabled 1b - EOCTU interrupt enabled
3 MSKJEOC	End of injected conversion interrupt mask When set, the JEOC interrupt is enabled. 0b - End of injected conversion interrupt disabled 1b - End of injected conversion interrupt enabled
2 MSKJECH	End of injected chain conversion interrupt mask When set, the JECH interrupt is enabled. 0b - End of injected chain conversion interrupt disabled 1b - End of injected chain conversion interrupt enabled
1 MSKEOC	End of conversion interrupt mask When set, the EOC interrupt is enabled. 0b - End of conversion interrupt disabled 1b - End of conversion interrupt enabled
0 MSKECH	End of chain conversion interrupt mask MSKECH: Mask bit for ECH. When set, the interrupt is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - End of chain conversion interrupt disabled 1b - End of chain conversion interrupt enabled

58.7.8 Channel Interrupt Mask Register 0 (CIMR0)

Offset

Register	Offset
CIMR0	24h

Function

Channel interrupt mask register for group0.

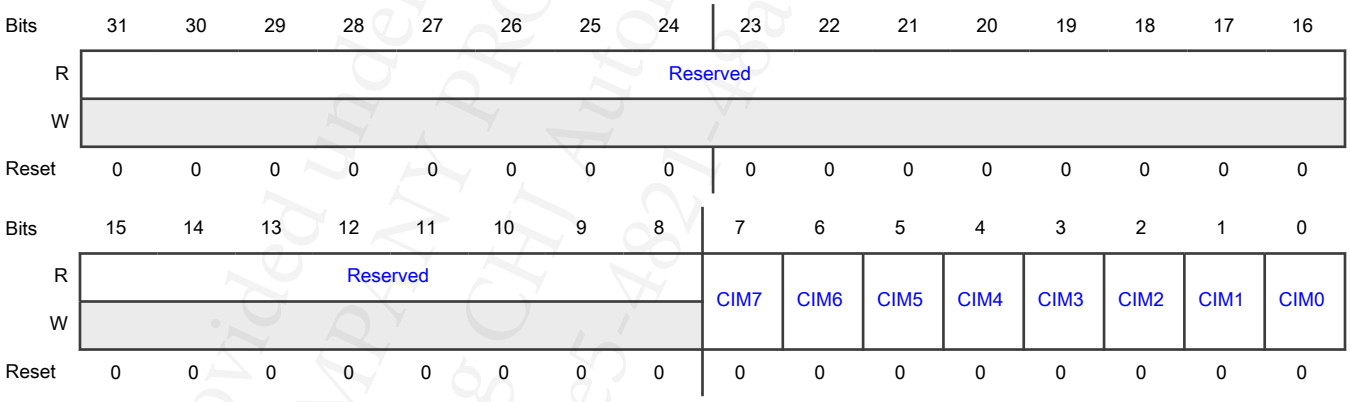
NOTE

The available CIMR0 fields are chip-specific. See chip-specific SAR_ADC information for details.

Table 529.

Bit 0	CIM0: Interrupt enable. When set (CIM0 = 1), interrupt for channel zero is enabled.
Bit n	CIMn: Interrupt enable. When set (CIMn = 1), interrupt for channel n is enabled.

Diagram



Fields

Field	Function
31-8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7 CIM7	Channel 7 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled
6 CIM6	Channel 6 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled
5 CIM5	Channel 5 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled
4 CIM4	Channel 4 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled
3 CIM3	Channel 3 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled
2 CIM2	Channel 2 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled
1 CIM1	Channel 1 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled
0 CIM0	Channel 0 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled

58.7.9 Channel Interrupt Mask Register 1 (CIMR1)

Offset

Register	Offset
CIMR1	28h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462626-4821-48a8-b251-390d3ab624c1

Function

Channel interrupt mask register for group32.

NOTE

The available CIMR1 fields are chip-specific. See chip-specific SAR_ADC information for details.

Table 530.

Bit 0	CIM0: Interrupt enable. When set (CIM0 = 1), interrupt for channel zero is enabled.
Bit n	CIMn: Interrupt enable. When set (CIMn = 1), interrupt for channel n is enabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								CIM39	CIM38	CIM37	CIM36	CIM35	CIM34	CIM33	CIM32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7 CIM39	Channel 39 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled
6 CIM38	Channel 38 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 CIM37	Channel 37 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled
4 CIM36	Channel 36 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled
3 CIM35	Channel 35 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled
2 CIM34	Channel 34 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled
1 CIM33	Channel 33 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled
0 CIM32	Channel 32 interrupt enable. 0b - Interrupt disabled 1b - Interrupt enabled

58.7.10 Watchdog Threshold Interrupt Status Register (WTISR)

Offset

Register	Offset
WTISR	30h

Function

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4888-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0		0		0		0		0		0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WDG7H	WDG7L	WDG6H	WDG6L	WDG5H	WDG5L	WDG4H	WDG4L	WDG3H	WDG3L	WDG2H	WDG2L	WDG1H	WDG1L	WDG0H	WDG0L
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15 WDG7H	This corresponds to the interrupt generated on the converted value being higher than the programmed higher threshold. 0b - Interrupt not asserted 1b - Interrupt asserted

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 WDG7L	This corresponds to the interrupt generated on the converted value being lower than the programmed lower threshold. 0b - Interrupt not asserted 1b - Interrupt asserted
13 WDG6H	This corresponds to the interrupt generated on the converted value being higher than the programmed higher threshold. 0b - Interrupt not asserted 1b - Interrupt asserted
12 WDG6L	This corresponds to the interrupt generated on the converted value being lower than the programmed lower threshold.
11 WDG5H	This corresponds to the interrupt generated on the converted value being higher than the programmed higher threshold. 0b - Interrupt not asserted 1b - Interrupt asserted
10 WDG5L	This corresponds to the interrupt generated on the converted value being lower than the programmed lower threshold. 0b - Interrupt not asserted 1b - Interrupt asserted
9 WDG4H	This corresponds to the interrupt generated on the converted value being higher than the programmed higher threshold. 0b - Interrupt not asserted 1b - Interrupt asserted
8 WDG4L	This corresponds to the interrupt generated on the converted value being lower than the programmed lower threshold.
7 WDG3H	This corresponds to the interrupt generated on the converted value being higher than the programmed higher threshold. 0b - Interrupt not asserted 1b - Interrupt asserted
6 WDG3L	This corresponds to the interrupt generated on the converted value being lower than the programmed lower threshold. 0b - Interrupt not asserted 1b - Interrupt asserted

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 WDG2H	This corresponds to the interrupt generated on the converted value being higher than the programmed higher threshold. 0b - Interrupt not asserted 1b - Interrupt asserted
4 WDG2L	This corresponds to the interrupt generated on the converted value being lower than the programmed lower threshold. 0b - Interrupt not asserted 1b - Interrupt asserted
3 WDG1H	This corresponds to the interrupt generated on the converted value being higher than the programmed higher threshold. 0b - Interrupt not asserted 1b - Interrupt asserted
2 WDG1L	This corresponds to the interrupt generated on the converted value being lower than the programmed lower threshold.
1 WDG0H	This corresponds to the interrupt generated on the converted value being higher than the programmed higher threshold. 0b - Interrupt not asserted 1b - Interrupt asserted
0 WDG0L	This corresponds to the interrupt generated on the converted value being lower than the programmed lower threshold. 0b - Interrupt not asserted 1b - Interrupt asserted

58.7.11 Watchdog Threshold Interrupt Mask Register (WTIMR)

Offset

Register	Offset
WTIMR	34h

Function

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482f-48a8-b251-390d3ab624d1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0		0		0		0		0		0		0		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MSKW DG7H	MSKW DG7L	MSKW DG6H	MSKW DG6L	MSKW DG5H	MSKW DG5L	MSKW DG4H	MSKW DG4L	MSKW DG3H	MSKW DG3L	MSKW DG2H	MSKW DG2L	MSKW DG1H	MSKW DG1L	MSKW DG0H	MSKW DG0L
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30 —	Reserved
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15 MSKWDG7H	This corresponds to the mask bit for the interrupt generated on the converted value being higher than the programmed higher threshold. When set, the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 MSKWDG7L	This corresponds to the mask bit for the interrupt generated on the converted value being lower than the programmed lower threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled
13 MSKWDG6H	This corresponds to the mask bit for the interrupt generated on the converted value being higher than the programmed higher threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled
12 MSKWDG6L	This corresponds to the mask bit for the interrupt generated on the converted value being lower than the programmed lower threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled
11 MSKWDG5H	This corresponds to the mask bit for the interrupt generated on the converted value being higher than the programmed higher threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled
10 MSKWDG5L	This corresponds to the mask bit for the interrupt generated on the converted value being lower than the programmed lower threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled
9 MSKWDG4H	This corresponds to the mask bit for the interrupt generated on the converted value being higher than the programmed higher threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled
8 MSKWDG4L	This corresponds to the mask bit for the interrupt generated on the converted value being lower than the programmed lower threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled
7 MSKWDG3H	This corresponds to the mask bit for the interrupt generated on the converted value being higher than the programmed higher threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled
6	This corresponds to the mask bit for the interrupt generated on the converted value being lower than the programmed lower threshold. When set the interrupt is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
MSKWDG3L	0b - Interrupt is disabled 1b - Interrupt is enabled
5 MSKWDG2H	This corresponds to the mask bit for the interrupt generated on the converted value being higher than the programmed higher threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled
4 MSKWDG2L	This corresponds to the mask bit for the interrupt generated on the converted value being lower than the programmed lower threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled
3 MSKWDG1H	This corresponds to the mask bit for the interrupt generated on the converted value being higher than the programmed higher threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled
2 MSKWDG1L	This corresponds to the mask bit for the interrupt generated on the converted value being lower than the programmed lower threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled
1 MSKWDG0H	This corresponds to the mask bit for the interrupt generated on the converted value being higher than the programmed higher threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled
0 MSKWDG0L	This corresponds to the mask bit for the interrupt generated on the converted value being lower than the programmed lower threshold. When set the interrupt is enabled. 0b - Interrupt is disabled 1b - Interrupt is enabled

58.7.12 DMAE Register (DMAE)

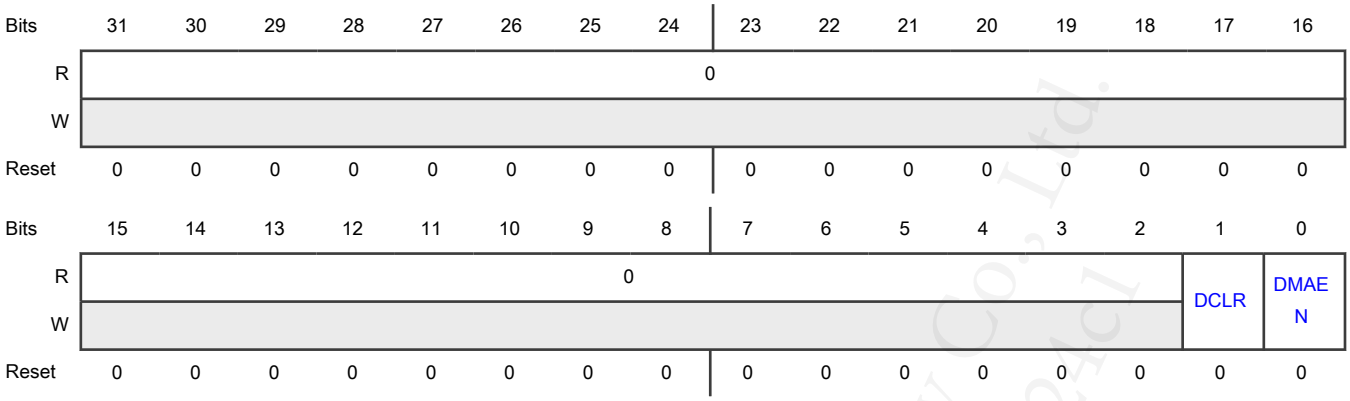
Offset

Register	Offset
DMAE	40h

Function

DMA settings must be changed while ADC is in idle state (MSR.ADCSTATUS = idle).

Diagram



Fields

Field	Function
31-2 —	Reserved
1 DCLR	DMA clear sequence enable 0b - DMA request cleared by acknowledge from DMA controller 1b - DMA request cleared on read of data registers
0 DMAEN	DMA global enable. 0b - DMA feature is disabled 1b - DMA feature is enabled

58.7.13 DMA Register 0 (DMAR0)

Offset

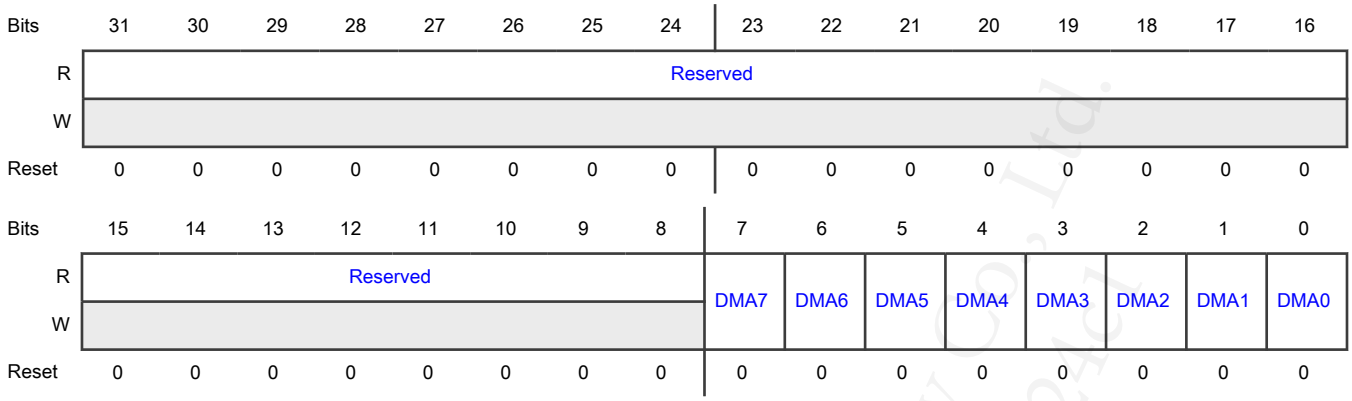
Register	Offset
DMAR0	44h

Function

DMA mode channel enables for Group0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-8 —	Reserved
7 DMA7	Channel 7 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
6 DMA6	Channel 6 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
5 DMA5	Channel 5 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
4 DMA4	Channel 4 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
3 DMA3	Channel 3 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
2 DMA2	Channel 2 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
1 DMA1	Channel 1 DMA enable.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
0 DMA0	Channel 0 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled

58.7.14 DMA Register 1 (DMAR1)

Offset

Register	Offset
DMAR1	48h

Function

Table 531.

Bit 0	DMA32: DMA enable. When set (DMA32 = 1), channel zero is enabled to transfer data in DMA mode.
Bit n	DMA*n*: DMA enable. When set (DMA*n* = 1), channel n is enabled to transfer data in DMA mode.

Channel Group32 DMA enables.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								DMA39	DMA38	DMA37	DMA36	DMA35	DMA34	DMA33	DMA32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHTA Automotive Technology Co., Ltd.
c8c462e5-48a2-4218a8-0-390d3ab624c1

Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7 DMA39	Channel 39 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
6 DMA38	Channel 38 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
5 DMA37	Channel 37 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
4 DMA36	Channel 36 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
3 DMA35	Channel 35 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
2 DMA34	Channel 34 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
1 DMA33	Channel 33 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled
0 DMA32	Channel 32 DMA enable. 0b - Transfer of data in DMA mode is disabled 1b - Transfer of data in DMA mode is enabled

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEC Anlong Active Technology Co., Ltd.
ca6625_4821-43a8-b251-390d3ab624c1

58.7.15 Analog Watchdog Threshold Register 0 (THRHLR0)

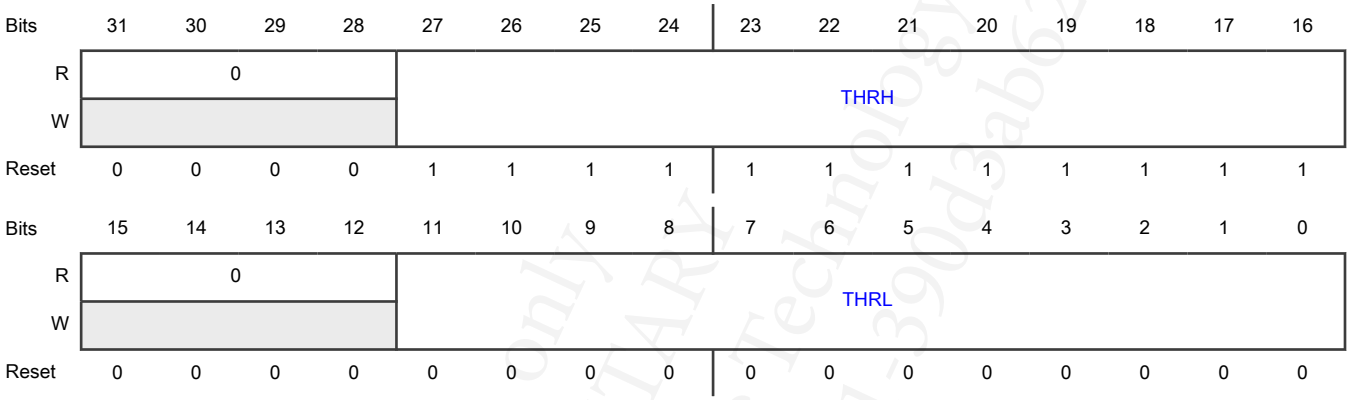
Offset

Register	Offset
THRHLR0	60h

Function

Analog Threshold Watchdog 0

Diagram



Fields

Field	Function
31-28 —	Reserved Reserved. Write of any value has no effect, read value is always zero.
27-16 THRH	THRH High threshold value for channel *n*.
15-12 —	Reserved
11-0 THRL	Low threshold value for channel *n*.

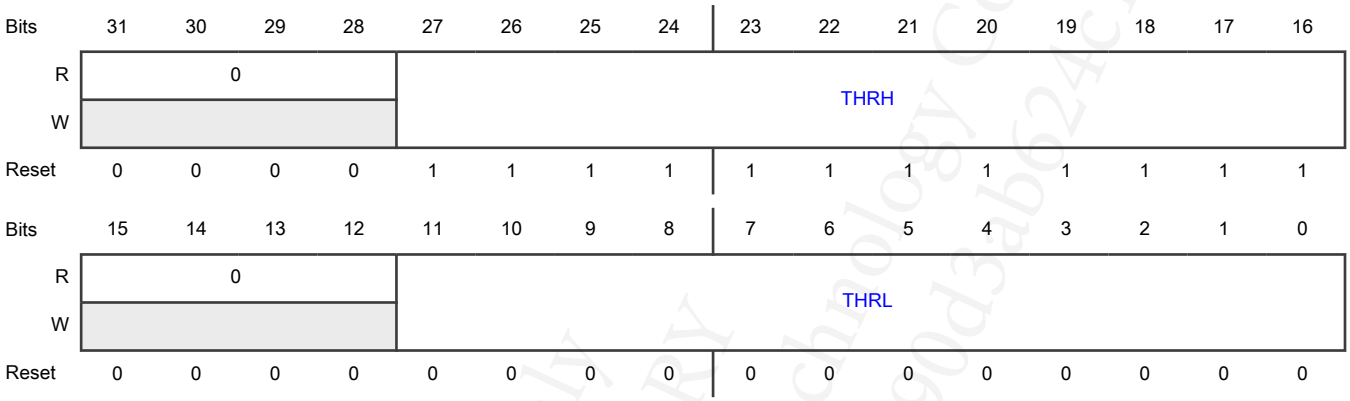
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390daab624c1

58.7.16 Analog Watchdog Threshold Register 1 (THRHLR1)

Offset

Register	Offset
THRHLR1	64h

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 THRH	High threshold value for channel *n*.
15-12 —	Reserved
11-0 THRL	Low threshold value for channel *n*.

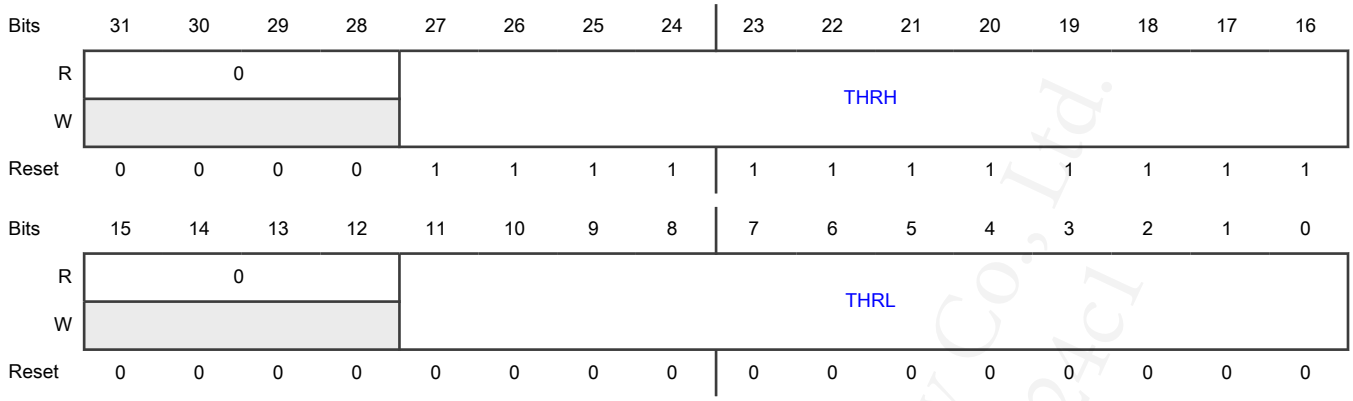
58.7.17 Analog Watchdog Threshold Register 2 (THRHLR2)

Offset

Register	Offset
THRHLR2	68h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 THRH	High threshold value for channel *n*.
15-12 —	Reserved
11-0 THRL	Low threshold value for channel *n*.

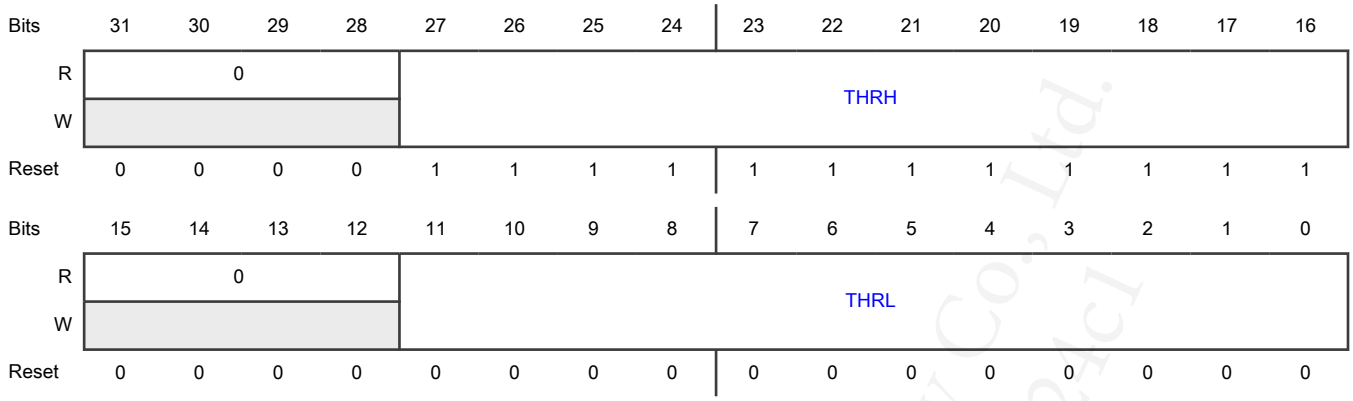
58.7.18 Analog Watchdog Threshold Register 2 (THRHLR3)

Offset

Register	Offset
THRHLR3	6Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8ab624c1

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 THRH	High threshold value for channel *n*.
15-12 —	Reserved
11-0 THRL	Low threshold value for channel *n*.

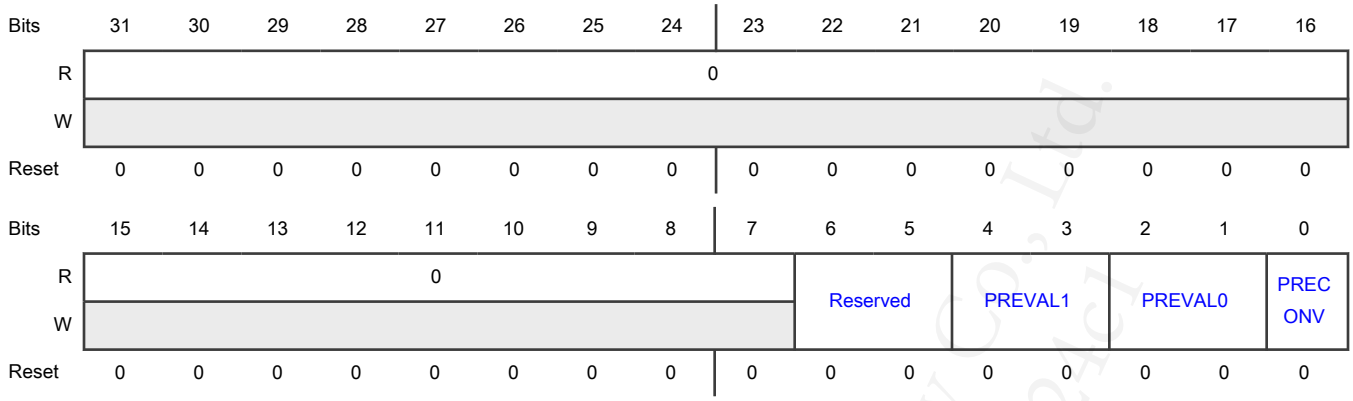
58.7.19 Presampling Control Register (PSCR)

Offset

Register	Offset
PSCR	80h

Provided by uma under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8ab624c1

Diagram



Fields

Field	Function
31-7 —	Reserved
6-5 —	Reserved
4-3 PREVAL1	Internal presampling voltage selection. Selects analog input voltages for presampling from four available internal voltages (channels in the range 32 to 63). See Presampling channel enable for details.
2-1 PREVAL0	Internal presampling voltage selection. Selects analog input voltages for presampling from four internal available voltages (channels in the range 0 to 31). See Presampling channel enable for details.
0 PRECONV	Convert presampled value If PRECONV is set, presampling is followed by the conversion. Sampling will be bypassed and conversion of presampled data will be done.

58.7.20 Presampling Register 0 (PSR0)

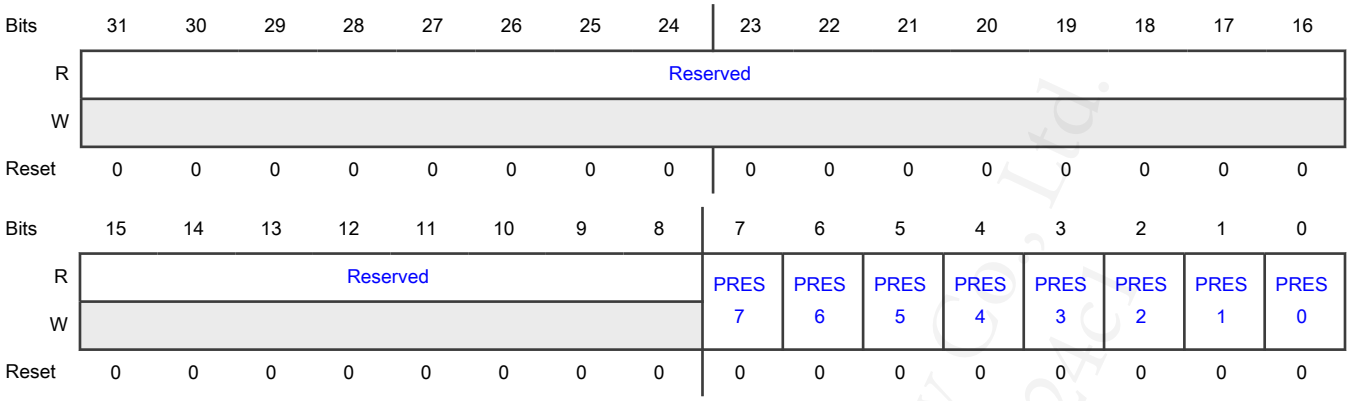
Offset

Register	Offset
PSR0	84h

Function

Enabled presampling for channel *n*, when set (PRES*n* = 1).

Diagram



Fields

Field	Function
31-8 —	Reserved
7 PRES7	Presampling enable for channel 7.
6 PRES6	Presampling enable for channel 6.
5 PRES5	Presampling enable for channel 5.
4 PRES4	Presampling enable for channel 4.
3 PRES3	Presampling enable for channel 3.
2 PRES2	Presampling enable for channel 2.
1 PRES1	Presampling enable for channel 1.
0 PRES0	Presampling enable for channel 0. 0b - Presampling is disabled 1b - Presampling is enabled

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-390d8ab624c2

58.7.21 Presampling Register 1 (PSR1)

Offset

Register	Offset
PSR1	88h

Function

Enabled presampling for channel *n*, when set (PRES*n* = 1).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								PRES 39	PRES 38	PRES 37	PRES 36	PRES 35	PRES 34	PRES 33	PRES 32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7 PRES39	Presampling enable for channel 39.
6 PRES38	Presampling enable for channel 38.
5 PRES37	Presampling enable for channel 37.

Table continues on the next page...

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c162e5-4821-48a8-b251-90d3a624c1

Table continued from the previous page...

Field	Function
4 PRES36	Presampling enable for channel 36.
3 PRES35	Presampling enable for channel 35.
2 PRES34	Presampling enable for channel 34.
1 PRES33	Presampling enable for channel 33.
0 PRES32	Presampling enable for channel 32. 0b - Presampling is disabled 1b - Presampling is enabled

58.7.22 Conversion Timing Register 0 (CTR0)

Offset

Register	Offset
CTR0	94h

Function

This register is associated with channels 0 to 31.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								INPSAMP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c16225-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-8 —	Reserved
7-0 INPSAMP	Sampling phase duration This field configures the sample period duration in terms of the SAR Controller clock. The minimum acceptable value is 8. Configuring to a value lower than 8 sets the sample period to 8 cycles. NOTE

58.7.23 Conversion Timing Register 1 (CTR1)

Offset

Register	Offset
CTR1	98h

Function

This register is associated with channels 32 to 63. An exception exists for the temperature sensor channel, which always uses the CTR1 value.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								INPSAMP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

Fields

Field	Function
31-8 —	Reserved
7-0	Sampling phase duration

Table continues on the next page...

Table continued from the previous page...

Field	Function
INPSAMP	This field configures the sample period duration in terms of the SAR Controller clock. The minimum acceptable value is 8. Configuring to a value lower than 8 sets the sample period to 8 cycles.

58.7.24 Normal Conversion Mask Register (NCMR0)

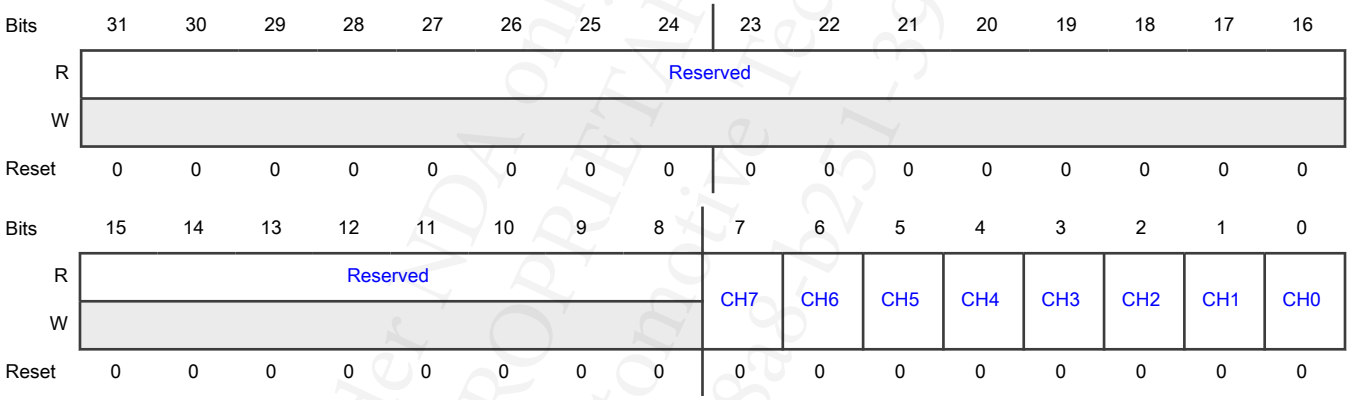
Offset

Register	Offset
NCMR0	A4h

Function

Sampling enable for Group0 channel *n* (*n* = Group0 channel number).

Diagram



Fields

Field	Function
31-8 —	Reserved
7 CH7	Normal sampling enable for channel 7.
6 CH6	Normal sampling enable for channel 6.
5 CH5	Normal sampling enable for channel 5.

Table continues on the next page...

Table continued from the previous page...

Field	Function
4 CH4	Normal sampling enable for channel 4.
3 CH3	Normal sampling enable for channel 3.
2 CH2	Normal sampling enable for channel 2.
1 CH1	Normal sampling enable for channel 1.
0 CH0	Normal sampling enable for channel 0. 0b - Normal sampling is disabled 1b - Normal sampling is enabled

58.7.25 Normal Conversion Mask Register (NCMR1)

Offset

Register	Offset
NCMR1	A8h

Function

Sampling enable for Group32 channel *n* (*n* = Group32 channel number).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								CH39	CH38	CH37	CH36	CH35	CH34	CH33	CH32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c16225-4821-42a8-b251-390d3ab624c1

Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved Reserved
7 CH39	Normal sampling enable for channel 39.
6 CH38	Normal sampling enable for channel 38.
5 CH37	Normal sampling enable for channel 37.
4 CH36	Normal sampling enable for channel 36.
3 CH35	Normal sampling enable for channel 35.
2 CH34	Normal sampling enable for channel 34.
1 CH33	Normal sampling enable for channel 33.
0 CH32	Normal sampling enable for channel 32. 0b - Normal sampling is disabled 1b - Normal sampling is enabled

58.7.26 Injected Conversion Mask Register (JCMR0)

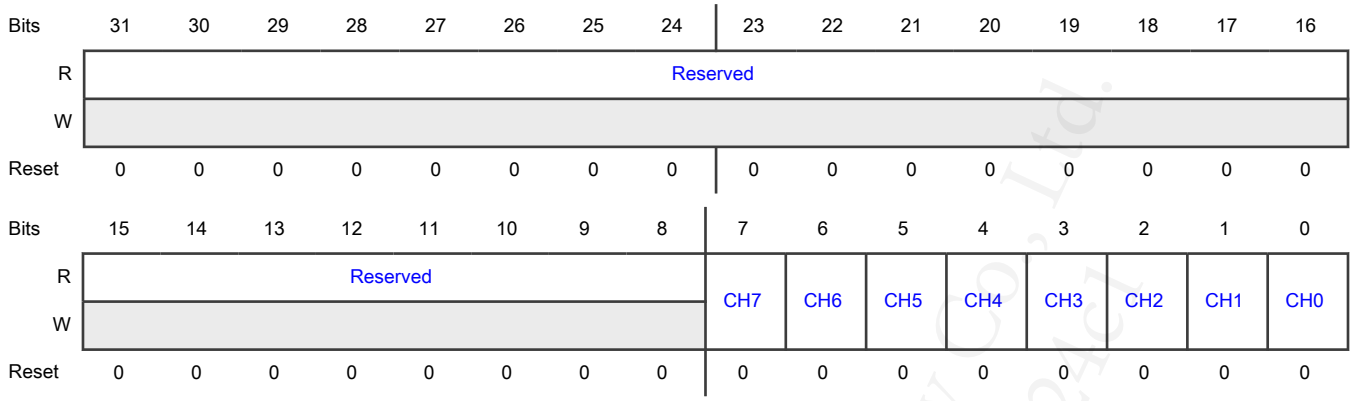
Offset

Register	Offset
JCMR0	B4h

Function

Injected sampling enable for Group0 channels.

Diagram



Fields

Field	Function
31-8 —	Reserved
7 CH7	Injected sampling enable for channel 7.
6 CH6	Injected sampling enable for channel 6.
5 CH5	Injected sampling enable for channel 5.
4 CH4	Injected sampling enable for channel 4.
3 CH3	Injected sampling enable for channel 3.
2 CH2	Injected sampling enable for channel 2.
1 CH1	Injected sampling enable for channel 1.
0 CH0	Injected sampling enable for channel 0. 0b - Injected sampling is disabled 1b - Injected sampling is enabled

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8d1e548-21-48a8-b251-390d8ab624c1

58.7.27 Injected Conversion Mask Register (JCMR1)

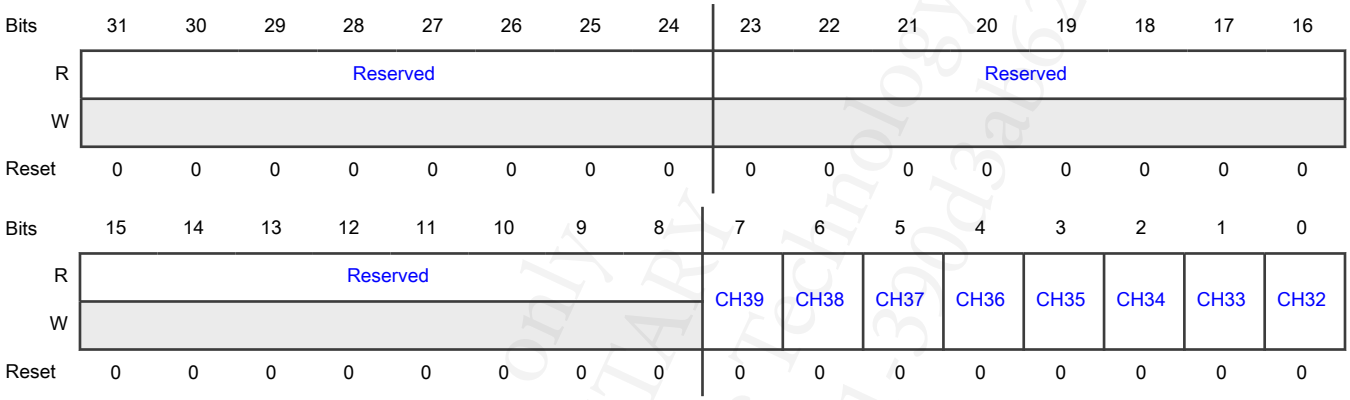
Offset

Register	Offset
JCMR1	B8h

Function

Injected sampling enable for Group32 channels.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7 CH39	Injected sampling enable for channel 39.
6 CH38	Injected sampling enable for channel 38.
5 CH37	Injected sampling enable for channel 37.
4	Injected sampling enable for channel 36.

Table continues on the next page...

Provided under NDA Only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c4625-1821-48a8-b251-390da1624c1

Table continued from the previous page...

Field	Function
CH36	
3 CH35	Injected sampling enable for channel 35.
2 CH34	Injected sampling enable for channel 34.
1 CH33	Injected sampling enable for channel 33.
0 CH32	Injected sampling enable for channel 32. 0b - Injected sampling is disabled 1b - Injected sampling is enabled

58.7.28 User OFFSET and Gain Register (USROFSGN)

Offset

Register	Offset
USROFSGN	C0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved								GAINUSER							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								OFFSUSER							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4828-b251-390d3ab624c1

Table continued from the previous page...

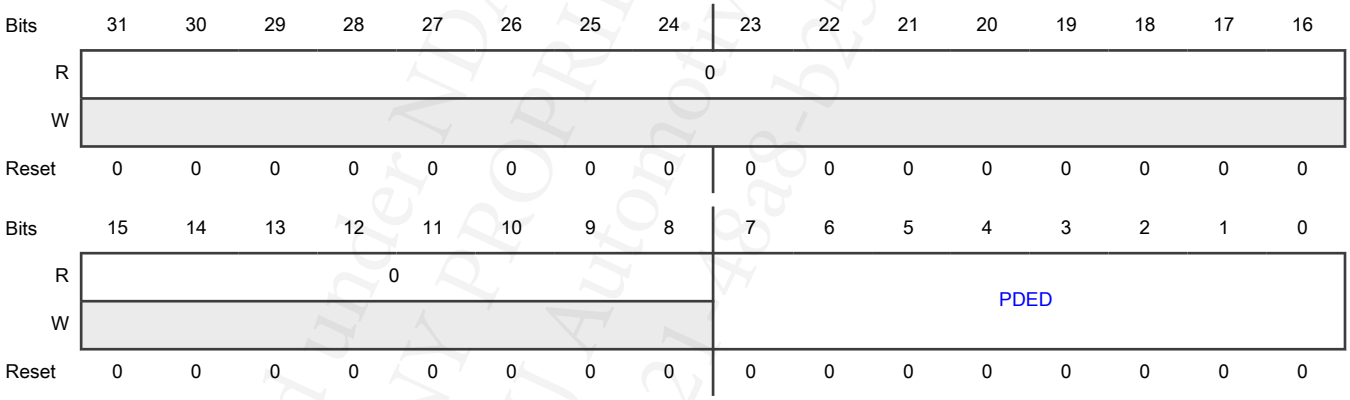
Field	Function
25-16 GAINUSER	User defined gain value Additional gain value in two's complement format used by ADC for data processing (for example, values can be either positive or negative; MSB represents sign bit).
15-8 —	Reserved
7-0 OFFSUSER	User defined offset Additional offset of ADC in two's complement format used for data processing (values can be either positive or negative; MSB represents sign bit).

58.7.29 Power Down Exit Delay Register (PDEDR)

Offset

Register	Offset
PDEDR	C8h

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 PDED	The delay between the power-down bit reset and the start of conversion. The power-down delay is at least (1/ADC Bus clock) + ((PDED+1) × 1/[ADC_clock_frequency]) where ADC_clock_frequency = frequency of ADC Hard Macro clock(AD_CLK) and ADC Bus clock = module interface clock for register access.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca62e5482114a8-b251-390d3ab624c1

58.7.30 Precision Channel n Data Register (PCDR0 - PCDR7)

Offset

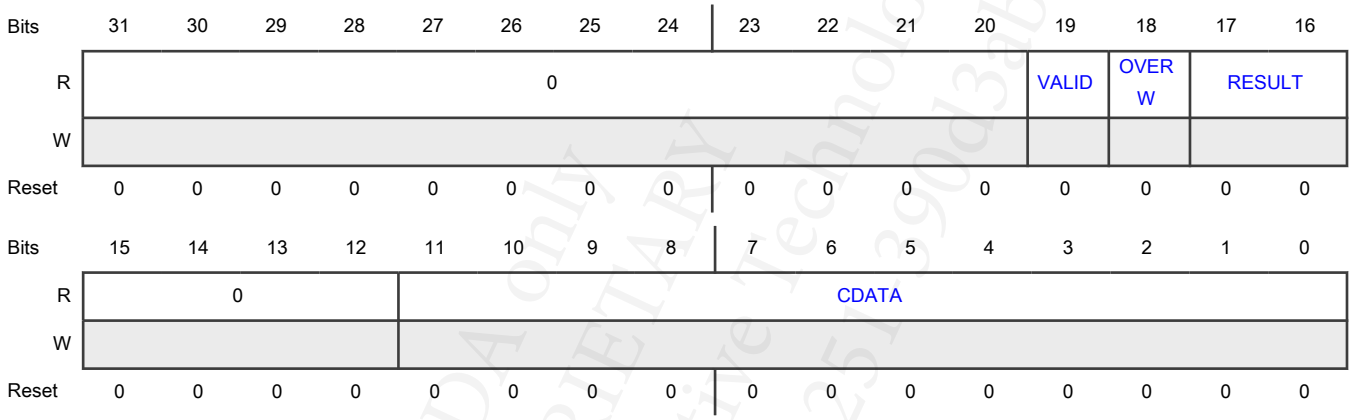
For a = 0 to 7:

Register	Offset
PCDRa	100h + (a × 4h)

Function

The conversion results for the Precision Channels are loaded into these data registers. Each data register also contains information regarding the result.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 VALID	Conversion data valid Used to notify when the data is valid (a new value has been written). It is automatically cleared when data is read.
18 OVERW	Data overwrite Indicates when conversion data was overwritten by a newer result. The new data is written or discarded according to MCR[OWREN].
17-16 RESULT	This field reflects the mode of conversion for the corresponding channel. 00b - Data is a result of Normal conversion mode 01b - Data is a result of Injected conversion mode 10b - Data is a result of CTU conversion mode 11b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 —	Reserved
11-0 CDATA	Channel converted data.

58.7.31 Internal Channel n Data Register (ICDR32 - ICDR39)

Offset

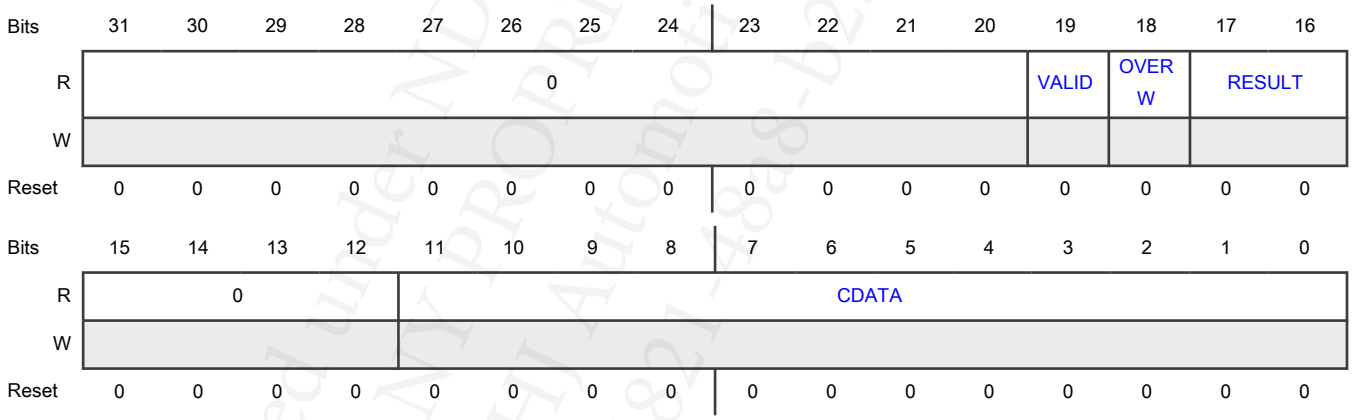
For a = 32 to 39:

Register	Offset
ICDRa	100h + (a × 4h)

Function

The conversion results for the Internal Channels are loaded into these data registers. Each data register also contains information regarding the result.

Diagram



Fields

Field	Function
31-20 —	Reserved
19 VALID	Conversion data valid

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

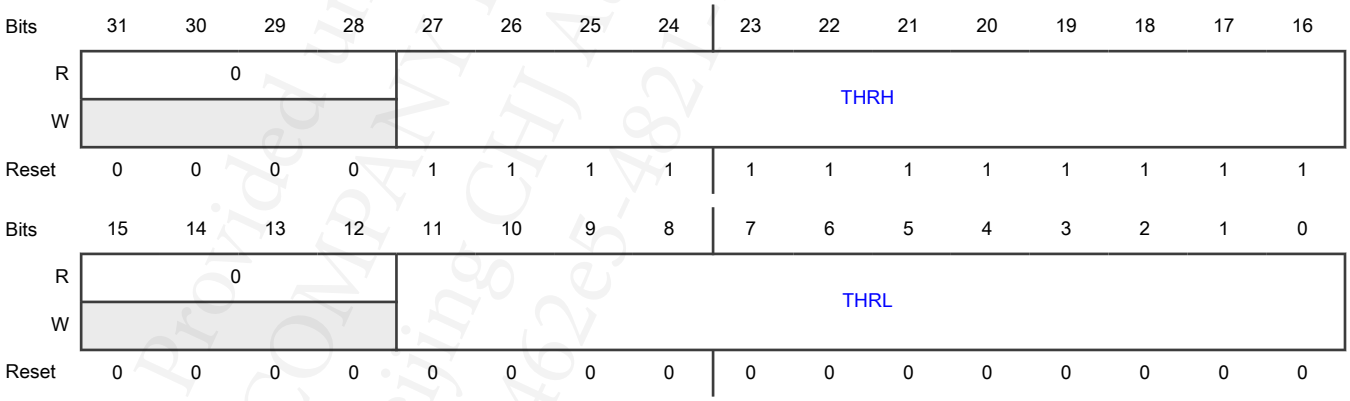
Field	Function
	Used to notify when the data is valid (a new value has been written). It is automatically cleared when data is read.
18 OVERW	Data overwrite Indicates when conversion data was overwritten by a newer result. The new data is written or discarded according to MCR[OWREN].
17-16 RESULT	This field reflects the mode of conversion for the corresponding channel. 00b - Data is a result of Normal conversion mode 01b - Data is a result of Injected conversion mode 10b - Data is a result of CTU conversion mode 11b - Reserved
15-12 —	Reserved
11-0 CDATA	Channel converted data.

58.7.32 Analog Watchdog Threshold Register 4 (THRHLR4)

Offset

Register	Offset
THRHLR4	280h

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Fields

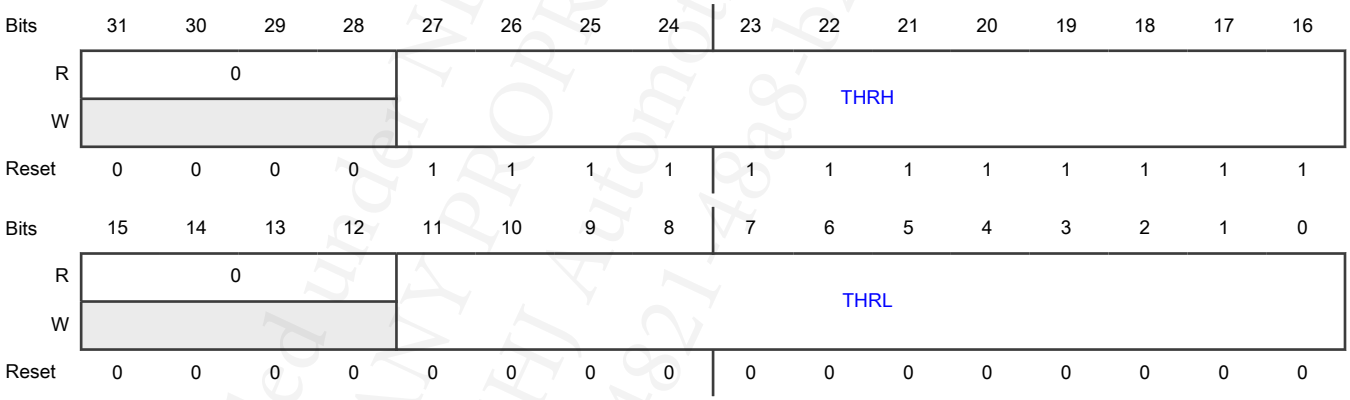
Field	Function
31-28 —	Reserved
27-16 THRH	High threshold value for channel *n*.
15-12 —	Reserved
11-0 THRL	Low threshold value for channel *n*.

58.7.33 Analog Watchdog Threshold Register 5 (THRHLR5)

Offset

Register	Offset
THRHLR5	284h

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 THRH	High threshold value for channel *n*.

Table continues on the next page...

Table continued from the previous page...

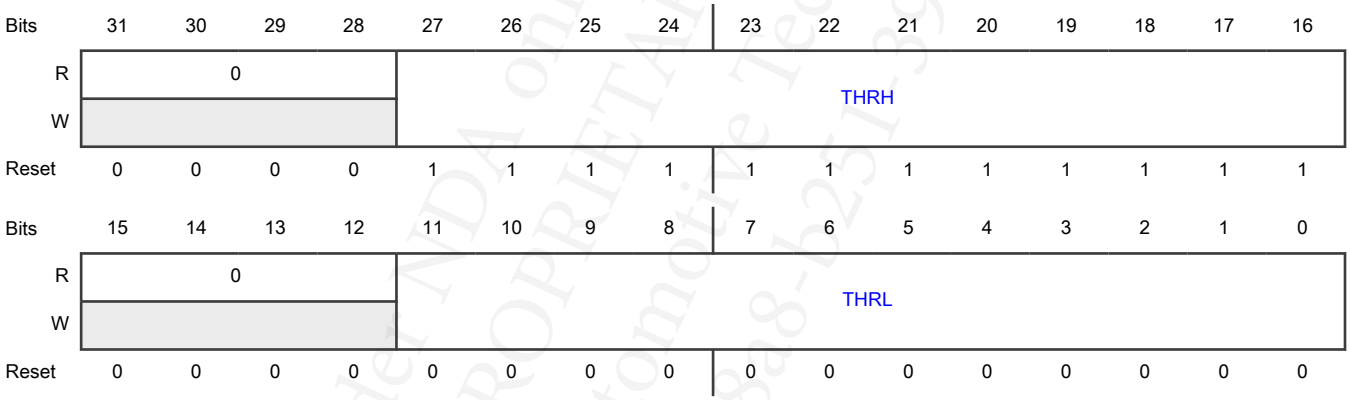
Field	Function
15-12 —	Reserved
11-0 THRL	Low threshold value for channel *n*.

58.7.34 Analog Watchdog Threshold Register 6 (THRHLR6)

Offset

Register	Offset
THRHLR6	288h

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 THRH	High threshold value for channel *n*.
15-12 —	Reserved
11-0 THRL	Low threshold value for channel *n*.

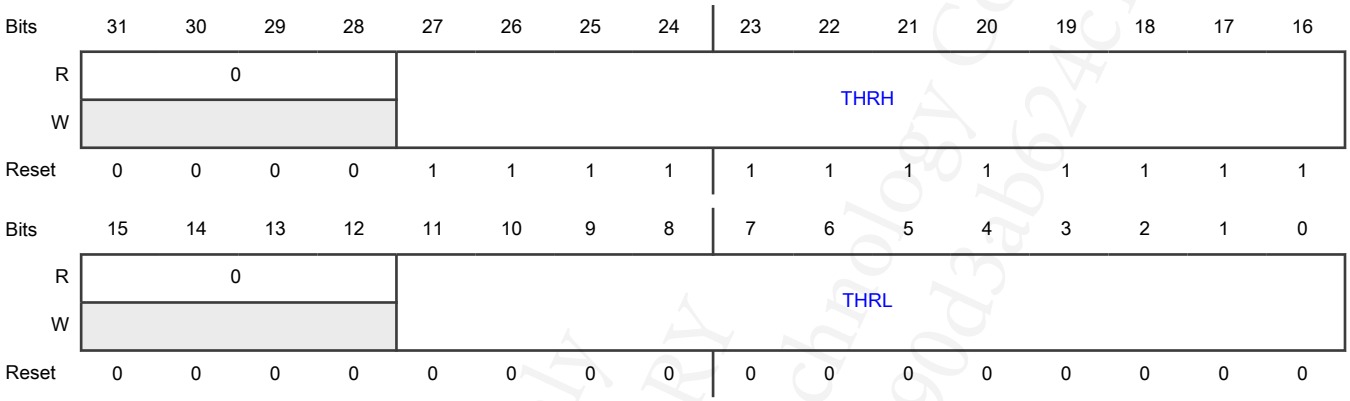
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca462e5-4821-48a8-b251-390d3ab624c1

58.7.35 Analog Watchdog Threshold Register 7 (THRHLR7)

Offset

Register	Offset
THRHLR7	28Ch

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 THRH	High threshold value for channel *n*.
15-12 —	Reserved
11-0 THRL	Low threshold value for channel *n*.

58.7.36 Channel Watchdog Select Register 0 (CWSELR0)

Offset

Register	Offset
CWSELR0	2B0h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Function

This register controls the selection of the threshold (or alternate threshold) register to provide upper and lower threshold values for channels 0 through 7.

Field	Description
WSEL_CH*n*	Channel Watchdog select for channel n. Selects the threshold (or alternate threshold) register for channel n. 0h THRHLR0 register is selected 1h THRHLR1 register is selected 2h THRHLR2 register is selected 3h THRHLR3 register is selected 4h THRHLR4 register is selected 5h THRHLR5 register is selected 6h THRHLR6 register is selected 7h THRHLR7 register is selected

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	WSEL_CH7			0	WSEL_CH6			0	WSEL_CH5			0	WSEL_CH4		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	WSEL_CH3			0	WSEL_CH2			0	WSEL_CH1			0	WSEL_CH0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-28 WSEL_CH7	Channel Watchdog select for channel 7. See the register description for values.
27 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
26-24 WSEL_CH6	Channel Watchdog select for channel 6. See the register description for values.
23 —	Reserved
22-20 WSEL_CH5	Channel Watchdog select for channel 5. See the register description for values.
19 —	Reserved
18-16 WSEL_CH4	Channel Watchdog select for channel 4. See the register description for values.
15 —	Reserved
14-12 WSEL_CH3	Channel Watchdog select for channel 3. See the register description for values.
11 —	Reserved
10-8 WSEL_CH2	Channel Watchdog select for channel 2. See the register description for values.
7 —	Reserved
6-4 WSEL_CH1	Channel Watchdog select for channel 1. See the register description for values.
3 —	Reserved
2-0 WSEL_CH0	Channel Watchdog select for channel 0. See the register description for values.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c46035-4821-48a8-b251-390d3ab624c1

58.7.37 Channel Watchdog Select Register 4 (CWSELR4)

Offset

Register	Offset
CWSELR4	2C0h

Function

This register controls the selection of the threshold (or alternate threshold) register to provide upper and lower threshold values for channels 32 through 39.

Field	Description
WSEL_CH*n*	Channel Watchdog select for channel n. Selects the threshold (or alternate threshold) register for channel n. 0h THRHLR0 register is selected 1h THRHLR1 register is selected 2h THRHLR2 register is selected 3h THRHLR3 register is selected 4h THRHLR4 register is selected 5h THRHLR5 register is selected 6h THRHLR6 register is selected 7h THRHLR7 register is selected

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	WSEL_CH39			0	WSEL_CH38			0	WSEL_CH37			0	WSEL_CH36		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	WSEL_CH35			0	WSEL_CH34			0	WSEL_CH33			0	WSEL_CH32		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA to COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e3-4821-48a8-b251-390d3a6b2401

Fields

Field	Function
31 —	Reserved
30-28 WSEL_CH39	Channel Watchdog select for channel 39. See the register description for values.
27 —	Reserved
26-24 WSEL_CH38	Channel Watchdog select for channel 38. See the register description for values.
23 —	Reserved
22-20 WSEL_CH37	Channel Watchdog select for channel 37. See the register description for values.
19 —	Reserved
18-16 WSEL_CH36	Channel Watchdog select for channel 36. See the register description for values.
15 —	Reserved
14-12 WSEL_CH35	Channel Watchdog select for channel 35. See the register description for values.
11 —	Reserved
10-8 WSEL_CH34	Channel Watchdog select for channel 34. See the register description for values.
7 —	Reserved
6-4 WSEL_CH33	Channel Watchdog select for channel 33. See the register description for values.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8a62e3e2-821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
3 —	Reserved
2-0 WSEL_CH32	Channel Watchdog select for channel 32. See the register description for values.

58.7.38 Channel Watchdog Enable Register (CWENR0)

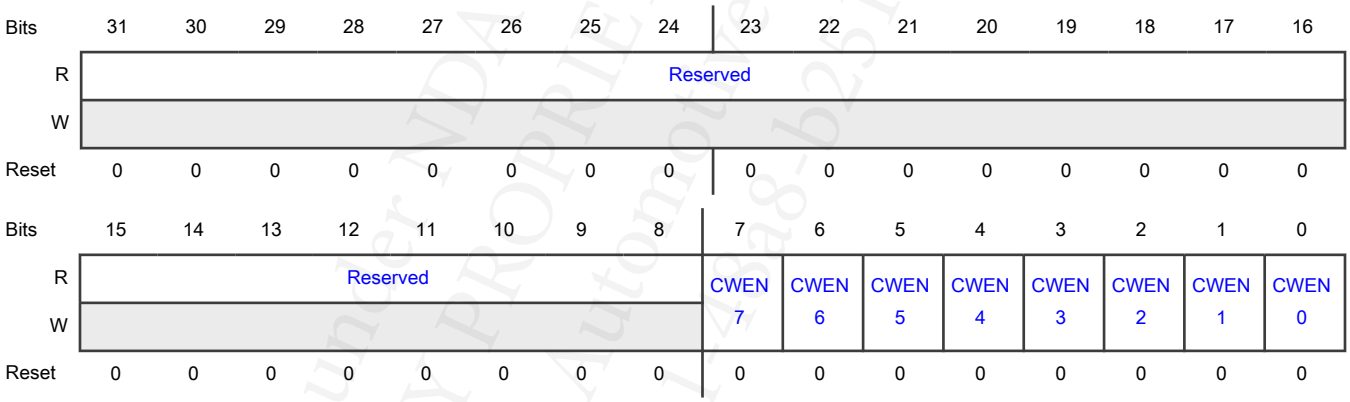
Offset

Register	Offset
CWENR0	2E0h

Function

Enabled Watchdog feature for channel *n*, when set (CWEN*n* = 1).

Diagram



Fields

Field	Function
31-8 —	Reserved
7 CWEN7	Watchdog enable for channel 7.
6	Watchdog enable for channel 6.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4b48-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
CWEN6	
5 CWEN5	Watchdog enable for channel 5.
4 CWEN4	Watchdog enable for channel 4.
3 CWEN3	Watchdog enable for channel 3.
2 CWEN2	Watchdog enable for channel 2.
1 CWEN1	Watchdog enable for channel 1.
0 CWEN0	Watchdog enable for channel 0. 0b - Watchdog is disabled 1b - Watchdog is enabled

58.7.39 Channel Watchdog Enable Register (CWENR1)

Offset

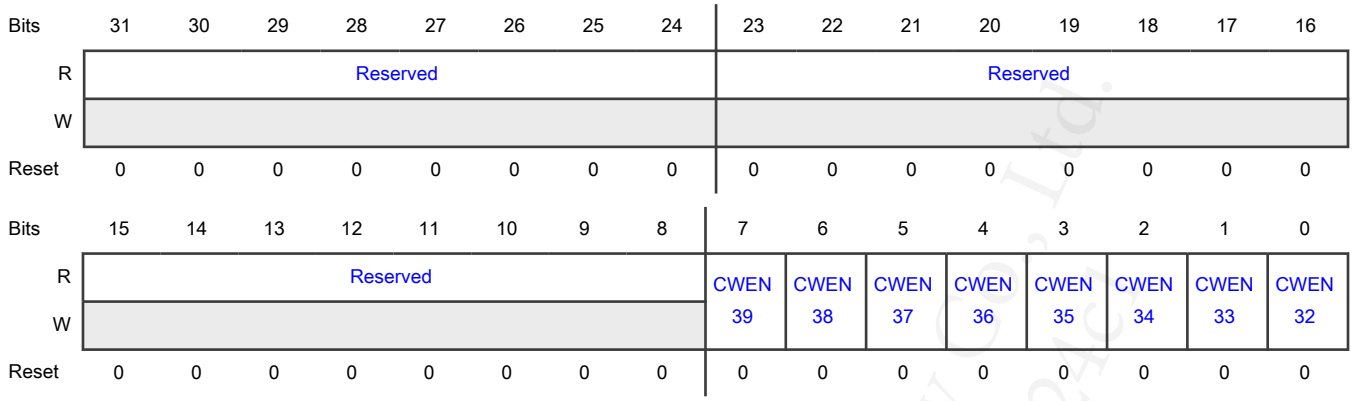
Register	Offset
CWENR1	2E4h

Function

Enabled Watchdog feature for channel *n*, when set (CWEN*n* = 1).

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7 CWEN39	Watchdog enable for channel 39.
6 CWEN38	Watchdog enable for channel 38.
5 CWEN37	Watchdog enable for channel 37.
4 CWEN36	Watchdog enable for channel 36.
3 CWEN35	Watchdog enable for channel 35.
2 CWEN34	Watchdog enable for channel 34.
1 CWEN33	Watchdog enable for channel 33.

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 CWEN32	Watchdog enable for channel 32. 0b - Watchdog is disabled 1b - Watchdog is enabled

58.7.40 Analog Watchdog Out of Range Register (AWORR0)

Offset

Register	Offset
AWORR0	2F0h

Function

AWORR_CH*n*: When set, indicates channel *n* converted data is out of range.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								AWOR_CH7	AWOR_CH6	AWOR_CH5	AWOR_CH4	AWOR_CH3	AWOR_CH2	AWOR_CH1	AWOR_CH0
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7 AWOR_CH7	Indicates channel 7 converted data is out of range.
6 AWOR_CH6	Indicates channel 6 converted data is out of range.

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 AWOR_CH5	Indicates channel 5 converted data is out of range.
4 AWOR_CH4	Indicates channel 4 converted data is out of range.
3 AWOR_CH3	Indicates channel 3 converted data is out of range.
2 AWOR_CH2	Indicates channel 2 converted data is out of range.
1 AWOR_CH1	Indicates channel 1 converted data is out of range.
0 AWOR_CH0	Indicates channel 0 converted data is out of range. 0b - Converted data is in range 1b - Converted data is out of range

58.7.41 Analog Watchdog Out of Range Register (AWORR1)

Offset

Register	Offset
AWORR1	2F4h

Function

AWORR_CH*n*: When set, indicates channel *n* converted data is out of range.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...	AWOR_CH...
W									W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-482-48a8-b251-390d3ab624c1

Fields

Field	Function
31-24 —	Reserved
23-16 —	Reserved
15-8 —	Reserved
7 AWOR_CH39	Indicates channel 39 converted data is out of range.
6 AWOR_CH38	Indicates channel 38 converted data is out of range.
5 AWOR_CH37	Indicates channel 37 converted data is out of range.
4 AWOR_CH36	Indicates channel 36 converted data is out of range.
3 AWOR_CH35	Indicates channel 35 converted data is out of range.
2 AWOR_CH34	Indicates channel 34 converted data is out of range.
1 AWOR_CH33	Indicates channel 33 converted data is out of range.
0 AWOR_CH32	Indicates channel 32 converted data is out of range. 0b - Converted data is in range 1b - Converted data is out of range

58.7.42 Self-Test Configuration Register 1 (STCR1)

Offset

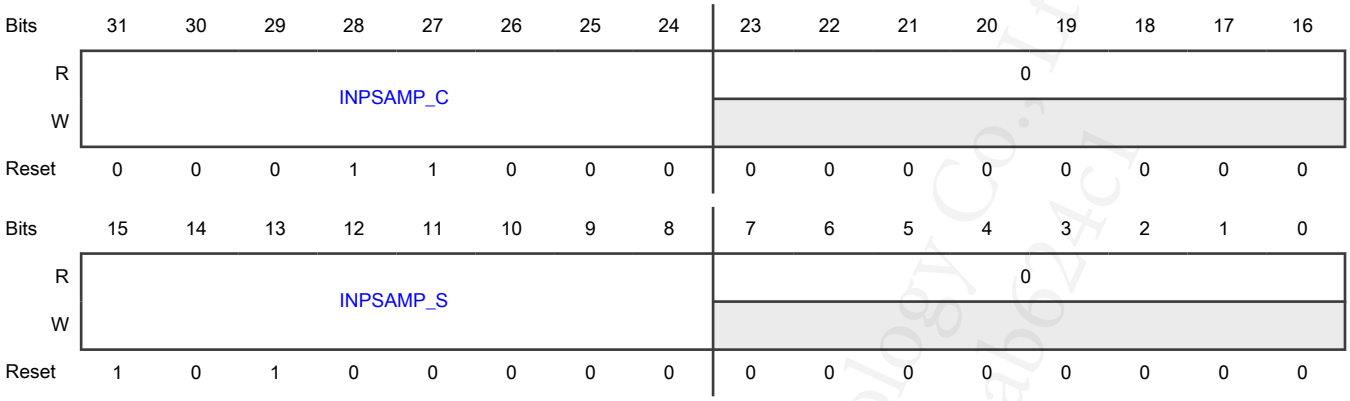
Register	Offset
STCR1	340h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJ Automotive Technology Co., Ltd.
c8c462e5-821-48a8-b251-390d3ab624c1

Function

This register defines ADC self-testing capabilities in CPU mode or CTU mode. The ADC parameters not defined by the ADC self-testing registers, these are controlled according to the setting defined for the functional conversions.

Diagram



Fields

Field	Function
31-24 INPSAMP_C	Sampling phase duration for the test conversions related to Algorithm C.
23-16 —	Reserved
15-8 INPSAMP_S	Sampling phase duration for the test conversions related to Algorithm S.
7-0 —	Reserved

58.7.43 Self-Test Configuration Register 2 (STCR2)

Offset

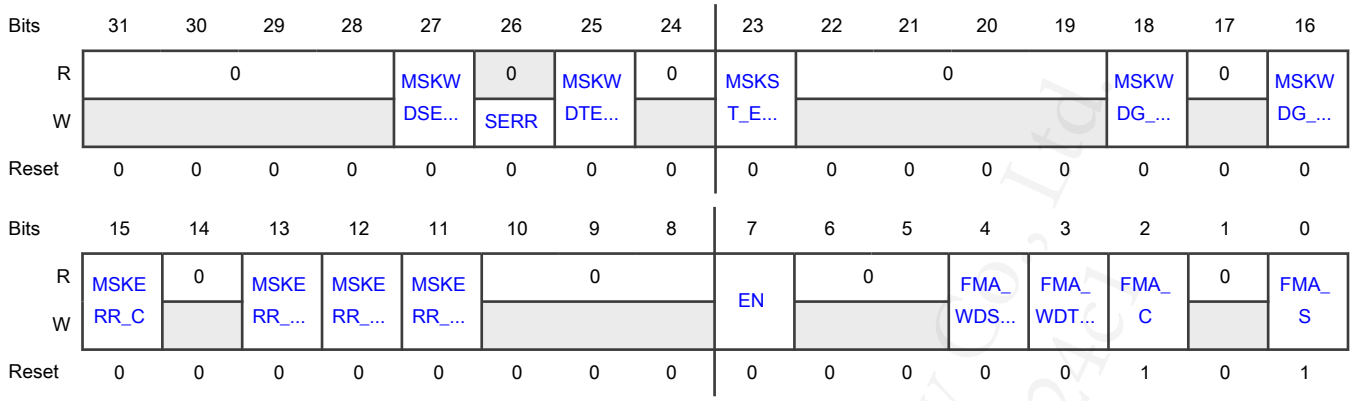
Register	Offset
STCR2	344h

Function

This register defines ADC self-testing capabilities in CPU mode or CTU mode.

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-9251-90d3ab624c1

Diagram



Fields

Field	Function
31-28 —	Reserved
27 MSKWDSERR	Watchdog sequence error interrupt mask Enables the Watchdog sequence error interrupt (STSR1[WDSERR] = 1). 0b - Interrupt disabled 1b - Interrupt enabled
26 SERR	Error fault injection field (write-only). Writing 1b to this field sets STSR1[ERR_C/S1/S2/S3] only once. Read always returns 0b. Setting this field will not affect any of the self-test data registers.
25 MSKWDTERR	Watchdog timer error interrupt mask Enables the Watchdog timer error interrupt (STSR[WDTERR] = 1). 0b - Interrupt disabled 1b - Interrupt enabled
24 —	Reserved
23 MSKST_EOC	Self-Test EOC interrupt mask Enables the self-test EOC interrupt (STSR1[ST_EOC] = 1). Also, enables IMR[MSKEOC] for the self-test EOC interrupt to be asserted. 0b - Interrupt disabled 1b - Interrupt enabled
22-19 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 MSKWDG_EOA_C	End of algorithm C interrupt mask Enables the end of Algorithm C interrupt (STSR1[WDG_EOA_C] = 1). 0b - Interrupt disabled 1b - Interrupt enabled
17 —	Reserved
16 MSKWDG_EOA_S	End of algorithm S interrupt mask Enables the end of Algorithm S interrupt (STSR1[WDG_EOA_S] = 1). 0b - Interrupt disabled 1b - Interrupt enabled
15 MSKERR_C	Error on algorithm C channel interrupt mask Enables the error on Algorithm C channel interrupt (STSR1[ERR_C] = 1). 0b - Interrupt disabled 1b - Interrupt enabled
14 —	Reserved
13 MSKERR_S2	Error on algorithm S2 channel interrupt mask Enables the error on Algorithm S2 channel (SUPPLY, step 2) interrupt (STSR1[ERR_S2] = 1). 0b - Interrupt disabled 1b - Interrupt enabled
12 MSKERR_S1	Error on algorithm S1 channel interrupt mask Enables the error on Algorithm S1 channel (SUPPLY, step 1) interrupt (STSR1[ERR_S1] = 1). 0b - Interrupt disabled 1b - Interrupt enabled
11 MSKERR_S0	Error on algorithm S0 channel interrupt mask Enables the error on Algorithm S1 channel (SUPPLY, step0) interrupt (STSR1[ERR_S0] = 1). 0b - Interrupt disabled 1b - Interrupt enabled
10-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 EN	Self-testing channel enable Enables the TEST channel only in CPU mode. In CTU trigger/control mode the enable is provided directly by the CTU. This field must be set before starting normal conversion and should not be changed while the conversion is in process. This field should only be reset after end of conversion of the last self-test channel has been received. 0b - OFF 1b - ON
6-5 —	Reserved
4 FMA_WDSERR	Fault mapping for watchdog sequence error By default, a failure on the Watchdog Sequence is mapped to drive the NCF line. 0b - NCF mapping 1b - CF mapping
3 FMA_WDTERR	Fault mapping for watchdog timer error By default, a failure on the Watchdog Timer is mapped to drive the NCF line. 0b - NCF mapping 1b - CF mapping
2 FMA_C	Fault mapping for algorithm C By default, a failure during Algorithm C is mapped to drive the CF fault line. 0b - NCF mapping 1b - CF mapping
1 —	Reserved
0 FMA_S	Fault mapping for BGAP algorithm By default, a failure during the supply algorithm is mapped to drive the Critical Fault (CF) line. 0b - Non-Critical Fault (NCF) mapping 1b - Critical Fault (CF) mapping

58.7.44 Self-Test Configuration Register 3 (STCR3)

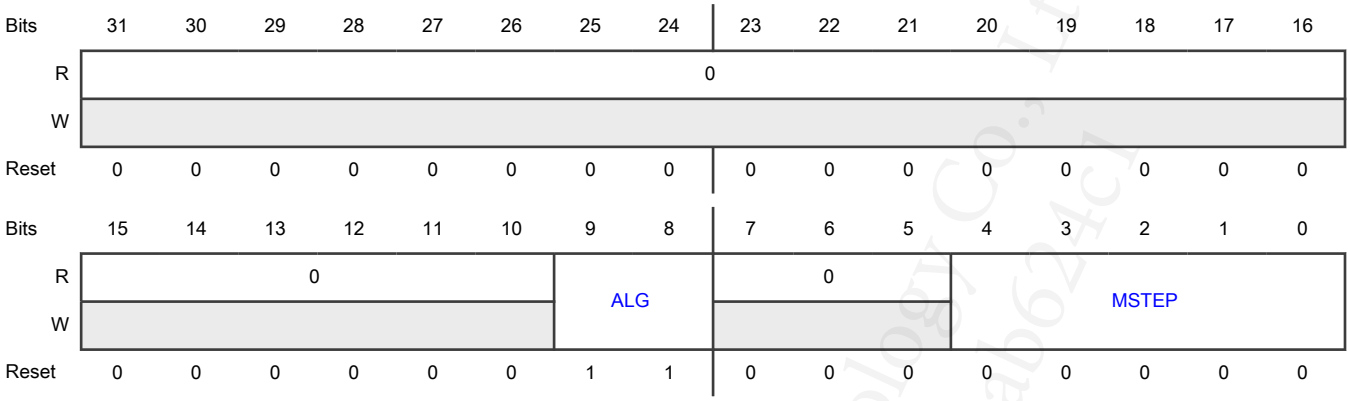
Offset

Register	Offset
STCR3	348h

Function

This register defines ADC self-testing capabilities in CPU mode (CTU disabled ≥ MCR[CTUEN] = 0). The contents of this register should be programmed before starting the conversion, and should not be changed while conversion is in process.

Diagram



Fields

Field	Function
31-10 —	Reserved
9-8 ALG	• One-Shot Operation mode algorithm scheduling: 00b - Algorithm S (single step=MSTEP) 01b - Reserved 10b - Algorithm C (single step=MSTEP) 11b - Algorithm S (default) For test/debug purposes. • Scan Operation mode algorithm scheduling: 00b - Algorithm S 01b - Reserved 10b - Algorithm C 11b - Algorithm S + Algorithm C (default): The execution baud rate of the selected algorithm is defined by STBRR configuration.
7-5 —	Reserved
4-0 MSTEP	For One-Shot Operation mode: Current step for Algorithm S//C.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• MSTEP = 0 to NUM_SUPPLY_STEPS - 1 for Algorithm S • MSTEP = 0 to NUM_C_STEPS - 1 for Algorithm C The unused codes are RESERVED (should not be used). This field is not used in Scan Operation mode as single-step execution (interleaved mode) is always performed. It should be programmed to zero in Scan Operation mode.

58.7.45 Self-Test Baud Rate Register (STBRR)

Offset

Register	Offset
STBRR	34Ch

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													WDT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								BR							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18-16 WDT	The watchdog timer value is used to monitor the algorithm sequence to verify that it is correctly executing within the safe time period. The self-testing watchdog is enabled by setting STAW*n*R[WDTE]. Default value is 10 ms. A fixed prescaler of 1024 is used to obtain the timer value from the ADCdig clock (80 MHz). Prescaler value is fixed at 1024. 000b - 0.1 ms ((0008h × Prescaler) cycles at 80 MHz) 001b - 0.5 ms ((0027h × Prescaler) cycles at 80 MHz)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	010b - 1 ms ((004Eh × Prescaler) cycles at 80 MHz) 011b - 2 ms ((009Ch × Prescaler) cycles at 80 MHz) 100b - 5 ms ((0187h × Prescaler) cycles at 80 MHz) 101b - 10 ms ((030Dh × Prescaler) cycles at 80 MHz) 110b - 20 ms (061Ah × Prescaler) cycles at 80 MHz) 111b - 50 ms (0F42h ×Prescaler) cycles at 80 MHz)
15-8 —	Reserved
7-0 BR	Algorithm baud rate Baud rate for the selected algorithm in Scan Operation mode (MCR[MODE] = 1). 00h ≥ maximum scheduling rate (NOMINAL rate) ... FFh ≥ minimum scheduling rate (NOMINAL rate scaled by 255) This field should be programmed before enabling a self-test channel.

58.7.46 Self-Test Status Register 1 (STSR1)

Offset

Register	Offset
STSR1	350h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				WDSERR	0	WDTERR	OVERWR	ST_EOC	0				WDG_EOA...	0	WDG_EOA...
W					W1C		W1C	W1C	W1C					W1C		W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERR_C	0	ERR_S2	ERR_S1	ERR_S0	0	STEP_C					0				
W	W1C		W1C	W1C	W1C		W1C									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Bernd Gehrmann Automotive Technology Co., Ltd.
c8c46c25-4f1-48a8-b251390d1ab624c1

Fields

Field	Function
31-28 —	Reserved
27 WDSERR	Watchdog sequence errors Watchdog sequence errors of the ADC subsystem (check for algorithm step sequence). If a failure occurs, an interrupt will be generated if enabled (STCR2[MSKWDSERR] = 1). It provides the fault indication to the FCCU, asserting a CF or NCF according to the STCR2[FMA_WDSERR] configuration. 0b - No failure 1b - Failure occurred
26 —	Reserved
25 WDTERR	Watchdog timer error Watchdog timer error of the ADC subsystem (algorithm check for completion within safe time). If a failure occurs, an interrupt will be generated if enabled (STCR2[MSKWDTERR] = 1). It provides the fault indication to the FCCU, asserting a CF or NCF according to the STCR2[FMA_WDTERR] configuration. 0b - No failure 1b - Failure occurred
24 OVERWR	Overwrite error Used to notify when STSR1[ERR*x*] is overwritten by a new error. The new error status is written or discarded according to the MCR[OWREN] value. To avoid Overwrite Error indication, STSR1[ERR*x*] must be cleared by software. 0b - No overwrite error 1b - Overwrite error occurred
23 ST_EOC	Self-test EOC This field is set along with EOC bit when end of conversion occurs from the analog SAR_ADC for self-test channel. An interrupt is generated if STCR2[MSKST_EOC] = 1, and IMR[MSKEOC] = 1. Writing 0 has no effect. 0b - Self-test end of conversion is not complete 1b - Self-test end of conversion is complete
22-19 —	Reserved
18 WDG_EOA_C	Indicates that Algorithm C has completed. This field is set after the last STEP of algorithm is executed. An interrupt is generated if STCR2[MSKWDG_EOA_C] = 1. This field will set only if STAW4R[WDTE] = 1. For CTU conversions, this field is significant only for Burst mode of operation. Writing 0 has no effect. 0b - Self-test end of Algorithm C conversion is not complete

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Self-test end of Algorithm C conversion is complete
17 —	Reserved
16 WDG_EOA_S	Indicates that Algorithm S has been completed. This field is set after the last Step of Algorithm S is executed. An interrupt is generated if STCR2[MSKWDG_EOA_S] = 1. This field is set only if STAW0R[WDTE] = 1. For CTU conversions, this field is significant only for Burst mode of operation. Writing 0 has no effect. 0b - Self-test end of Algorithm S conversion is not complete. 1b - Self-test end of Algorithm S conversion is complete.
15 ERR_C	Algorithm C error STSR1[ERR_C] indicates an error on the self-testing channel (Algorithm C). An interrupt is generated if STCR2[MSKERR_C] = 1. It provides the fault indication to the FCCU, asserting the programmed fault line (STCR2[FMA_*x*]). STSR1[ERR_C] can also be set by software (fault injection) by writing 1 to STCR2[SERR]. In this case, a CF or NCF will be asserted according to the STCR2[FMA_*x*] configuration. Writing 0 has no effect. 0b - No Algorithm C error 1b - Algorithm C error occurred
14 —	Reserved
13 ERR_S2	STSR1[ERR_S2] indicates an error on the self-test channel (Algorithm S (SUPPLY), step2). An interrupt is generated if STCR2[MSKERR_S2] = 1. It provides the fault indication to the FCCU, asserting the programmed fault line (STCR2[FMA_*x*]). STSR1[ERR_S2] can be also set by software (fault injection) by writing 1 to STCR2[SERR]. In this case, a CF or NCF will be asserted according to the STCR2[FMA_*x*] configuration. Writing 0 has no effect. 0b - No error occurred on the sampled signal 1b - Error occurred on the sampled signal
12 ERR_S1	STSR1[ERR_S1] indicates an error on the self-test channel (Algorithm S (SUPPLY), step1). An interrupt is generated if STCR2[MSKERR_S1] = 1. It provides the fault indication to the FCCU, asserting the programmed fault line (STCR2[FMA_*x*]). STSR1[ERR_S1] can be also set by software (fault injection) by writing 1 to STCR2[SERR]. In this case, a CF or NCF will be asserted according to the STCR2[FMA_*x*] configuration. Writing 0 has no effect. 0b - No VDD ERROR 1b - VDD ERROR occurred
11 ERR_S0	STSR1[ERR_S0] indicates an error on the self-test channel (Algorithm S (SUPPLY), step0). An interrupt is generated if STCR2[MSKERR_S0] = 1. It provides the fault indication to the FCCU, asserting the programmed fault line (STCR2[FMA_*x*]). STSR1[ERR_S0] can be also set by software (fault injection) by

Table continues on the next page...

Table continued from the previous page...

Field	Function
	writing 1 to STCR2[SERR]. In this case, a CF or NCF will be asserted according to the STCR2[FMA_*x*] configuration. Writing 0 has no effect. 0b - No VREF error 1b - VREF error occurred
10 —	Reserved
9-5 STEP_C	Algorithm C step number error Step of the Algorithm C step when an Algorithm C error occurred (as indicated by STSR1[ERR_C]). 0...(NUM_C_STEPS-1) ≥ Algorithm C
4-0 —	Reserved

58.7.47 Self-Test Status Register 2 (STSR2)

Offset

Register	Offset
STSR2	354h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	OVFL	0			DATA1											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				DATA0											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	Overflow bit

Table continues on the next page...

Table continued from the previous page...

Field	Function
OVFL	Overflow bit is set when divisor is zero. Also, STSR1.ERR_S1 is set if overflow occurs.
30-28 —	Reserved
27-16 DATA1	Test channel converted data when the ERR_S1 has occurred. Algorithm S (step1) ≥ fractional part of the ratio TEST (step1)/TEST (step0) = VDD/VBGAP.
15-12 —	Reserved
11-0 DATA0	Test channel converted data when the ERR_S1 has occurred. Algorithm S (step1) ≥ integer part of the ratio TEST (step1)/TEST (step0) = VDD/VBGAP.

58.7.48 Self-Test Status Register 3 (STSR3)

Offset

Register	Offset
STSR3	358h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				DATA1											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				DATA0											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-16 DATA1	Test channel converted data when the ERR_S2 has occurred. Algorithm S (step2) ≥ TEST channel data = VREF/VREF
15-12 —	Reserved
11-0 DATA0	Test channel converted data when the ERR_S0 has occurred. Algorithm S (step0) ≥ TEST channel data = VBGAP/VREF

58.7.49 Self-Test Status Register 4 (STSR4)

Offset

Register	Offset
STSR4	35Ch

Function

NOTE

If MCR.OVERWR is set, registers STSR[2:4] are overwritten if another error occurs.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				DATA1											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

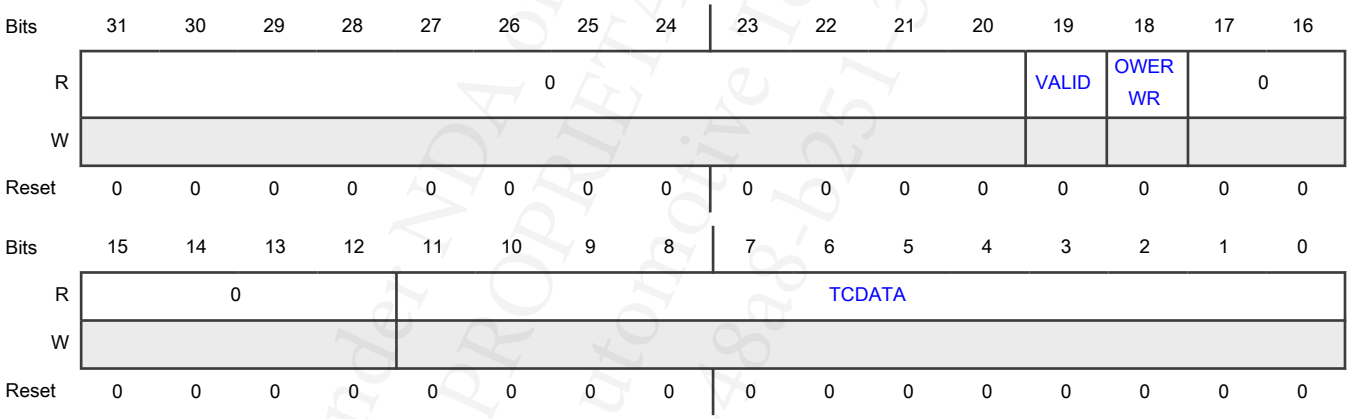
Field	Function
27-16 DATA1	DATA1 Test channel converted data when ERR_C has occurred. Algorithm C ≥ TEST channel data
15-0 —	Reserved

58.7.50 Self-Test Data Register 1 (STDR1)

Offset

Register	Offset
STDR1	370h

Diagram



Fields

Field	Function
31-20 —	Reserved
19 VALID	Valid data Indicates when data is valid (a new value is written). The field clears automatically when data is read.
18 OWERWR	Overwrite data Used to indicate when conversion data is overwritten by a newer result. The new data is written or discarded according to the MCR[OWREN] value.

Table continues on the next page...

Table continued from the previous page...

Field	Function
17-12 —	Reserved
11-0 TCDATA	Test channel converted data Test channel data is in 2's complement format.

58.7.51 Self-Test Data Register 2 (STDR2)

Offset

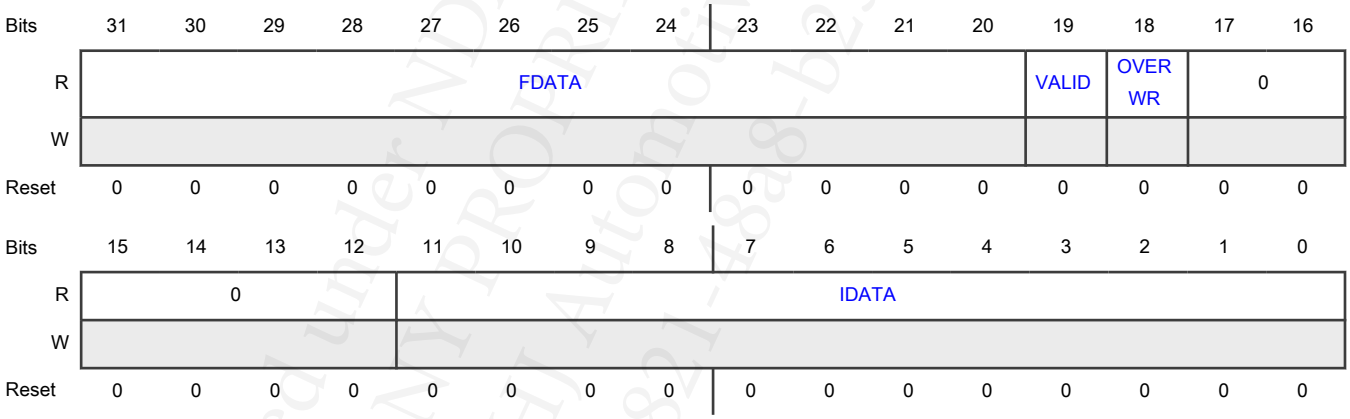
Register	Offset
STDR2	374h

Function

NOTE

If an overflow error occurs for Step1 of Algorithm S, STDR2 is not written.

Diagram



Fields

Field	Function
31-20 FDATA	Fractional data Fractional part of the ratio TEST (step1)/TEST (step0) = VDD/VBGAP for Algorithm S.
19 VALID	Valid data Indicates when data is valid (a new value is written). This field clears automatically when data is read.

Table continues on the next page...

Table continued from the previous page...

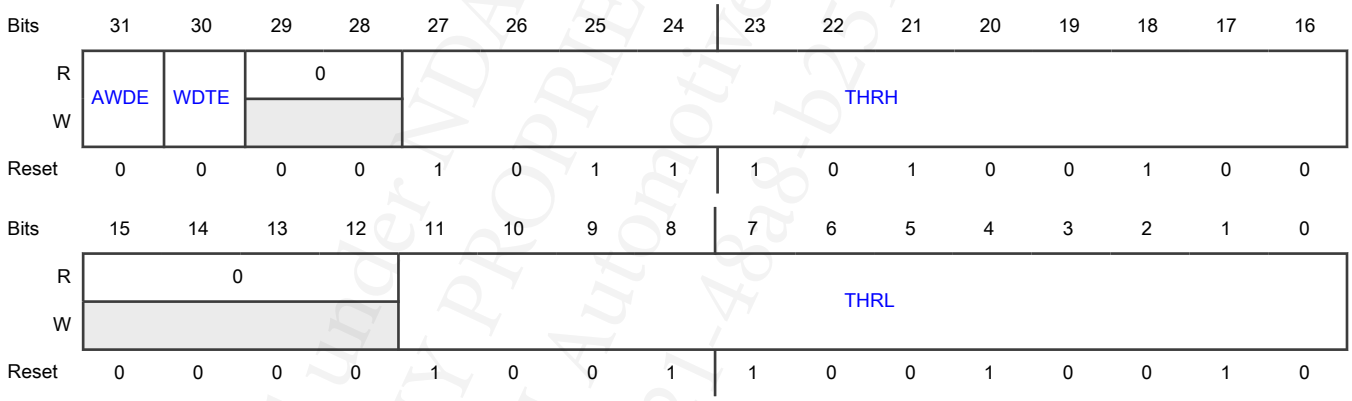
Field	Function
18 OVERWR	Overwrite data Indicates when conversion data is overwritten by a newer result. The new data is written or discarded according to the MCR[OWREN] configuration.
17-12 —	Reserved
11-0 IDATA	Integer data Integer part of the ratio TEST (step1)/TEST (step0) = VDD/VBGAP for Algorithm S.

58.7.52 Self-Test Analog Watchdog Register (STAW0R)

Offset

Register	Offset
STAW0R	380h

Diagram



Fields

Field	Function
31 AWDE	Enables/disables the comparison of the conversion result from the ADC supply self-test step0 to the thresholds contained in this register (THRH and THRL). 0b - Disabled 1b - Enabled
30 WDTE	Watchdog timer enable (related to the Algorithm S) Enables/disables the watchdog timer monitoring function for all ADC supply self-test steps.

Table continues on the next page...

Table continued from the previous page...

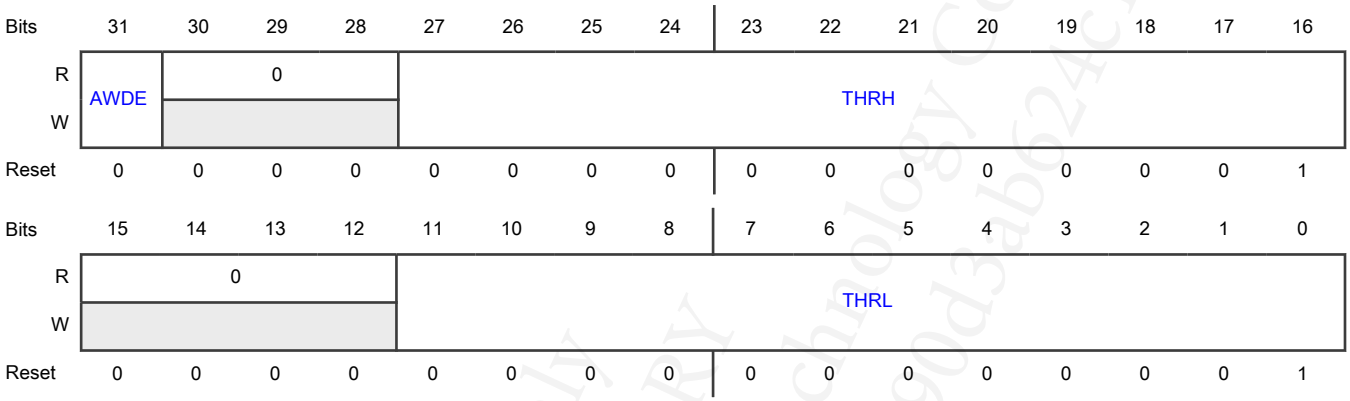
Field	Function
	The watchdog timer verifies: • Correct sequence of the algorithm (step sequence) • Execution of the algorithm within the safe time period as defined by STBRR[WDT] As soon as the watchdog timer is enabled the algorithm execution must be detected within the safe time period. The watchdog timer is reset each time the algorithm restarts. Set this field only in Scan mode. 0b - Disabled 1b - Enabled
29-28 —	Reserved
27-16 THRH	High threshold value for Algorithm S step0 If the analog watchdog is enabled, STSR1[ERR_*x*] is set to 1 if STDR1[TCDATA] > THRH. The reset value of this field is calculated based on a bandgap voltage of 1.2 V. Program the value 2912d (or B60h) for a bandgap voltage of 1.15 V. The formula for calculating the higher threshold is (Bandgap_Voltage_Maximum)*4095/(Reference_Voltage_Minimum). **NOTE** The recommended watchdog threshold values are set according to expected and tested results in a noise-controlled environment (for example, introduction of noise from outside of the device minimized). The setting may need to be adjusted to prevent a given threshold value from falsely reporting fault detections in applications operating in noisier environments.
15-12 —	Reserved
11-0 THRL	Low threshold value for Algorithm S step0 If the analog watchdog is enabled, STSR1[ERR_*x*] is set to 1 if STDR1[TCDATA] < THRL. The reset value of this field is calculated based on a bandgap voltage of 1.2 V. Program the value 2367d (or 93Fh) for a bandgap voltage of 1.15 V. The formula for calculating the lower threshold is (Bandgap_Voltage_Minimum)*4095/(Reference_Voltage_Maximum). **NOTE** The recommended watchdog threshold values are set according to expected and tested results in a noise-controlled environment (for example, introduction of noise from outside of the device minimized). This value may need to be adjusted to prevent a given threshold value from falsely reporting fault detections in applications operating in noisier environments.

58.7.53 Self-Test Analog Watchdog Register (STAW1AR)

Offset

Register	Offset
STAW1AR	384h

Diagram



Fields

Field	Function
31 AWDE	Analog watchdog enable related to Algorithm S (step1).
30-28 —	Reserved
27-16 THRH	High threshold value (integer part) for Algorithm S (step1) test channel (unsigned coding).
15-12 —	Reserved
11-0 THRL	Low threshold value (integer part) for Algorithm S (step1) test channel (unsigned coding).

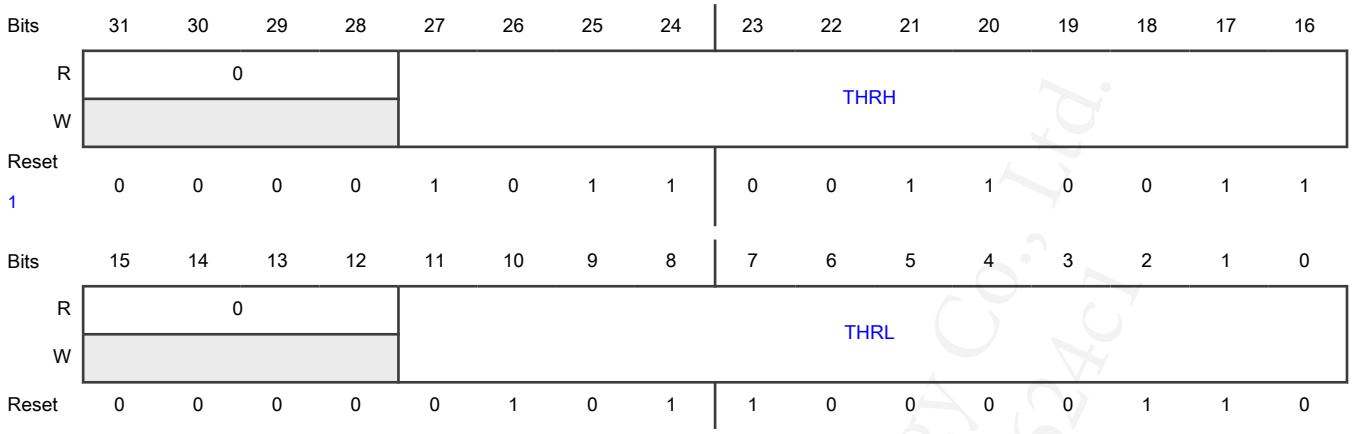
58.7.54 Self-Test Analog Watchdog Register 1B (STAW1BR)

Offset

Register	Offset
STAW1BR	388h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e5-4821-48a8-b251-390d3ab624c1

Diagram



1. 12-bit instance

Fields

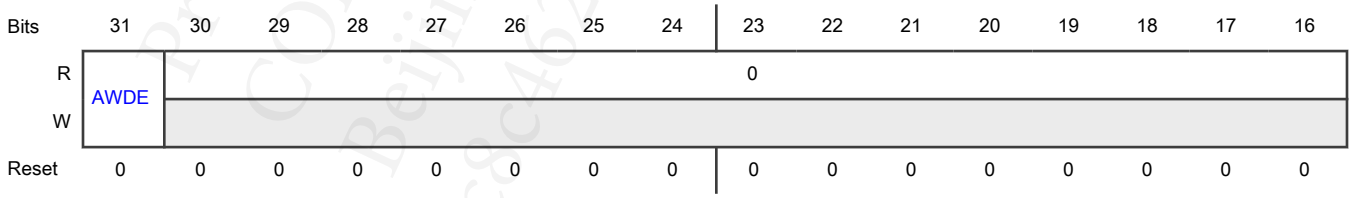
Field	Function
31-28 —	Reserved
27-16 THRH	High threshold value (fractional part) for Algorithm S (step1) test channel (unsigned coding).
15-12 —	Reserved
11-0 THRL	Low threshold value (fractional part) for Algorithm S (step1) test channel (unsigned coding).

58.7.55 Self-Test Analog Watchdog Register 2 (STAW2R)

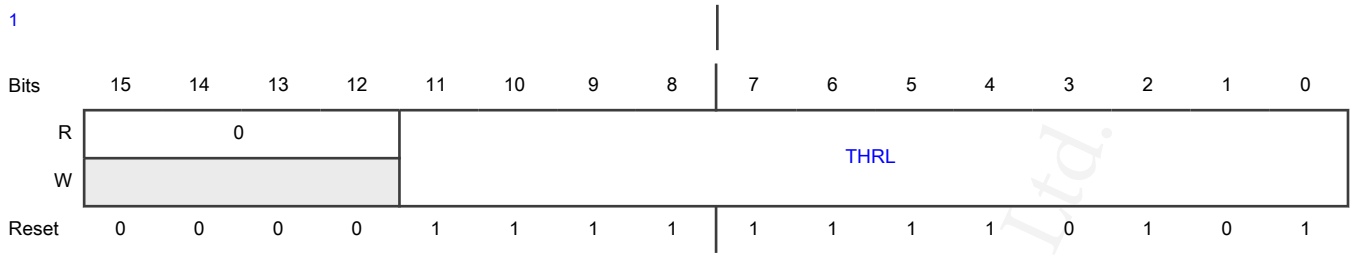
Offset

Register	Offset
STAW2R	38Ch

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4b2e5-4821-48a2-b251-390d3ab614c1



1. 12-bit instance

Fields

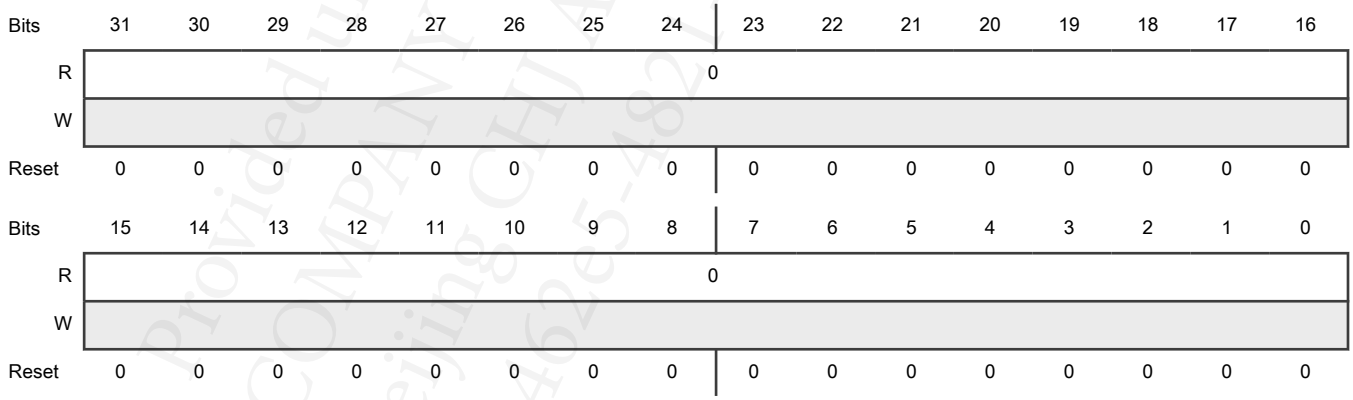
Field	Function
31 AWDE	Analog watchdog enable related to Algorithm S (step2).
30-12 —	Reserved
11-0 THRL	Threshold level low Low threshold value for channel *n* (unsigned coding). If the analog watchdog is enabled, STSR1[ERR_S2] is set if STDR1[TCDATA] < THRL.

58.7.56 Self-Test Analog Watchdog Register 3 (STAW3R)

Offset

Register	Offset
STAW3R	390h

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4824-48a8-b251-590d3ab624c1

Fields

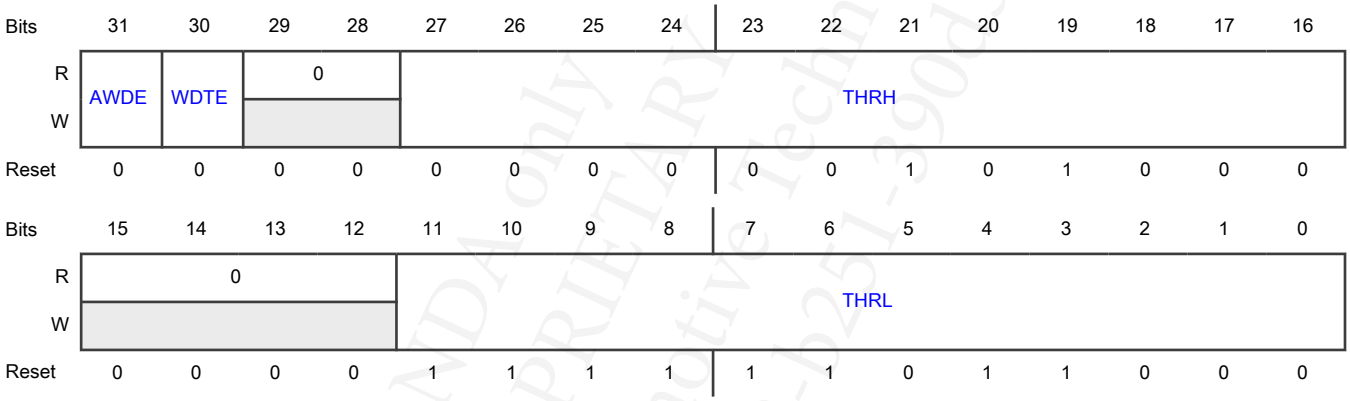
Field	Function
31-0 —	Reserved

58.7.57 Self-Test Analog Watchdog Register 4 (STAW4R)

Offset

Register	Offset
STAW4R	394h

Diagram



Fields

Field	Function
31 AWDE	Analog watchdog enable (related to Algorithm C) Enables/disables the analog watchdog monitoring function for all ADC capacitive self-test steps. 0b - Disabled 1b - Enabled
30 WDTE	Watchdog timer enable (related to Algorithm C). Enables/disables the watchdog timer monitoring function for all ADC capacitive self-test steps. The watchdog timer verifies: • Correct sequence of the algorithm (step sequence) • Execution of the algorithm within the safe time period as defined by STBRR[WDT] As soon as the watchdog timer is enabled, execution must be detected within the safe time period. The watchdog timer is reset each time the algorithm restarts.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field should be set only in scan mode. 0b - Disabled 1b - Enabled
29-28 —	Reserved
27-16 THRH	High threshold value for step0 of Algorithm C. This value is read by the ADC as two's-complement, but should be written positive. If the watchdog is enabled (STAW4R[AWDE] = 1) and STDR1[TCDATA] > STAW4R[THRH], then STSR1[ERR_C] is set. **NOTE** The recommended watchdog threshold values are set according to expected and tested results in a noise-controlled environment (for example, introduction of noise from outside of the device minimized). The setting may need to be adjusted to prevent a given threshold value from falsely reporting fault detections in applications operating in noisier environments.
15-12 —	Reserved
11-0 THRL	Low threshold value for step0 of Algorithm C. This value is read by the ADC as two's-complement, and it should be the inverse of STAW4R[THRH]. If the watchdog is enabled (STAW4R[AWDE] = 1) and STDR1[TCDATA] < STAW4R[THRL], then STSR1[ERR_C] is set. **NOTE** 1. The settings in this register are applicable only to step0 of the ADC's capacitive self-test. Refer to the STAW5R register for settings applicable to other capacitive self-test steps. 2. The recommended watchdog threshold values are set according to expected and tested results in a noise-controlled environment (for example, introduction of noise from outside of the device minimized). The setting may need to be adjusted to prevent a given threshold value from falsely reporting fault detections in applications operating in noisier environments.

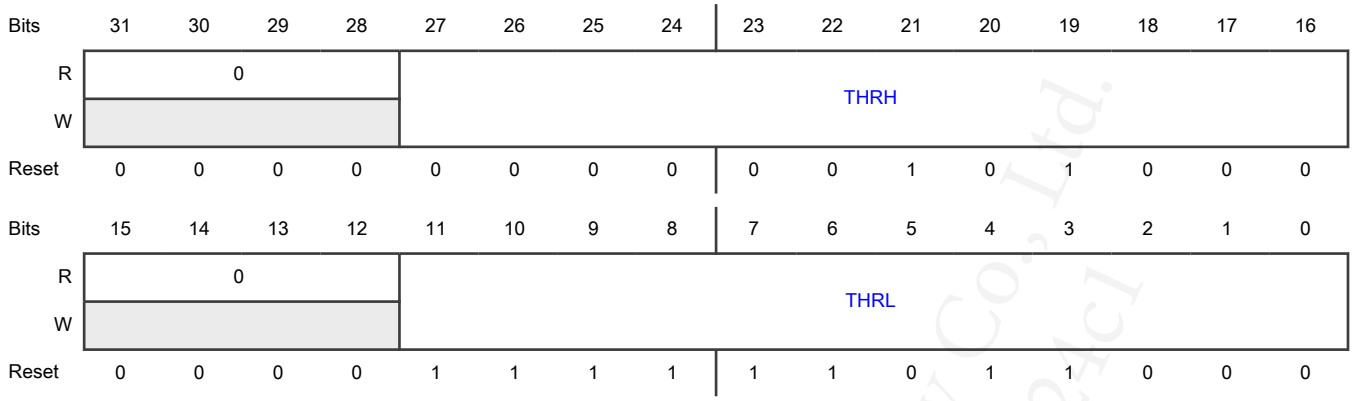
58.7.58 Self-Test Analog Watchdog Register 5 (STAW5R)

Offset

Register	Offset
STAW5R	398h

Provided under NDA only
CONFIDENTIAL
Beijing CHEETAH Automotive Technology Co., Ltd.
8462654-4581-4258-9251-3900ab024c1

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 THRH	High threshold value for step N of Algorithm C (N = 1 to CS-1). This value is read by the ADC as two's-complement, but should be written positive. If the watchdog is enabled (STAW4R[AWDE] = 1) and (STDR1[TCDATA] (step-N)–STDR1[TCDATA] (step-0)) > STAW5R[THRH], then STSR1[ERR_C] is set. **NOTE** The recommended watchdog threshold values are set according to expected and tested results in a noise-controlled environment (for example, introduction of noise from outside of the device minimized). The setting may need to be adjusted to prevent a given threshold value from falsely reporting fault detections in applications operating in noisier environments.
15-12 —	Reserved
11-0 THRL	Low threshold value for step0 of Algorithm C. This value is read by the ADC as two's-complement, and it should be the inverse of STAW5R[THRH]. If the watchdog is enabled (STAW4R[AWDE = 1]) and (STDR1[TCDATA] (step-N)–STDR1[TCDATA] (step-0)) < STAW5R[THRL], then STSR1[ERR_C] is set. **NOTE** The recommended watchdog threshold values are set according to expected and tested results in a noise-controlled environment (for example, introduction of noise from outside of the device minimized). The setting may need to be adjusted to prevent a given threshold value from falsely reporting fault detections in applications operating in noisier environments.

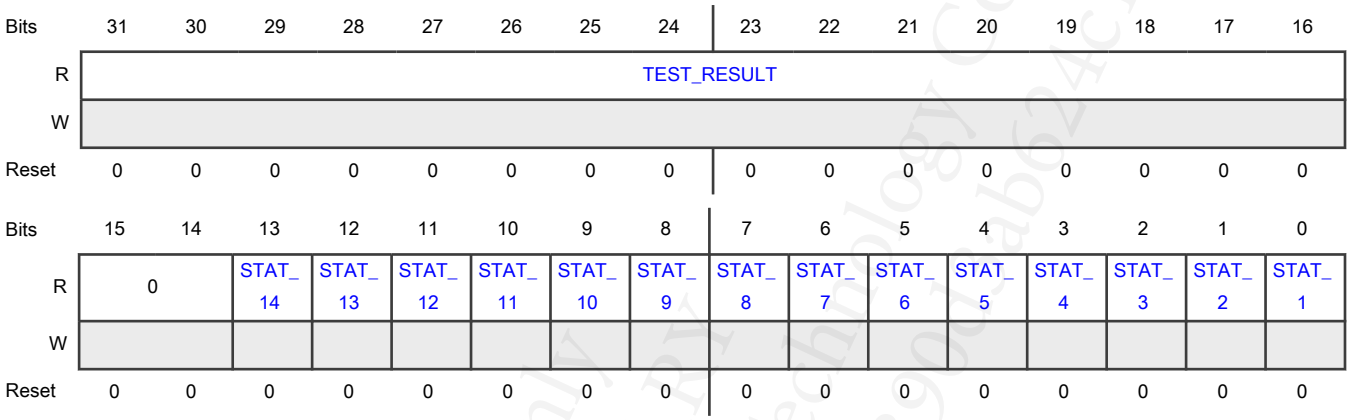
Provided under NDA
COMPANY PROPRIETARY
Bosch CSP Automotive Electronics (Suzhou) Co., Ltd.
c8c6e5-521-48ad-9251390d8ab624c1

58.7.59 Calibration Status register (CALSTAT)

Offset

Register	Offset
CALSTAT	39Ch

Diagram



Fields

Field	Function
31-16 TEST_RESULT	TEST_RESULT Test Result for last failing test ($FIXED_VALUE–$RAW_SAR_DATA). Bit [31] is a sign bit, and negative numbers are in two's complement.
15-14 —	Reserved
13 STAT_14	Status of calibration step 14 Status of test *n* (where *n* = 1...12). This field is cleared when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. Bit for each test is updated dynamically during test and the test number field reflects the last failing test.
12 STAT_13	Status of calibration step 13 This field is cleared by hardware when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. This field must be checked after end of calibration. 0b - Calculated gain within range 1b - Calculated gain out of range
11	Status of Calibration step 12

Table continues on the next page...

Table continued from the previous page...

Field	Function
STAT_12	Status of test *n* (where *n* = 1...12). This field is cleared when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. Bit for each test is updated dynamically during test and the test number field reflects the last failing test. 0b - Test passed 1b - Test failed
10 STAT_11	Status of calibration step 11 Status of test *n* (where *n* = 1...12). This field is cleared when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. Bit for each test is updated dynamically during test and the test number field reflects the last failing test. 0b - Test passed 1b - Test failed
9 STAT_10	Status of calibration step 10 Status of test *n* (where *n* = 1...12). This field is cleared when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. Bit for each test is updated dynamically during test and the test number field reflects the last failing test. 0b - Test passed 1b - Test failed
8 STAT_9	Status of calibration step 9 Status of test *n* (where *n* = 1...12). This field is cleared when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. Bit for each test is updated dynamically during test and the test number field reflects the last failing test. 0b - Test passed 1b - Test failed
7 STAT_8	Status of calibration step 8 Status of test *n* (where *n* = 1...12). This field is cleared when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. Bit for each test is updated dynamically during test and the test number field reflects the last failing test. 0b - Test passed 1b - Test failed

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 STAT_7	Status of calibration step 7 Status of test *n* (where *n* = 1...12). This field is cleared when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. Bit for each test is updated dynamically during test and the test number field reflects the last failing test. 0b - Test passed 1b - Test failed
5 STAT_6	Status of calibration step 6 Status of test *n* (where *n* = 1...12). This field is cleared when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. Bit for each test is updated dynamically during test and the test number field reflects the last failing test. 0b - Test passed 1b - Test failed
4 STAT_5	Status of calibration step 5 Status of test *n* (where *n* = 1...12). This field is cleared when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. Bit for each test is updated dynamically during test and the test number field reflects the last failing test. 0b - Test passed 1b - Test failed
3 STAT_4	Status of calibration step 4 Status of test *n* (where *n* = 1...12). This field is cleared when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. Bit for each test is updated dynamically during test and the test number field reflects the last failing test. 0b - Test passed 1b - Test failed
2 STAT_3	Status of calibration step 3 Status of test *n* (where *n* = 1...12). This field is cleared when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. Bit for each test is updated dynamically during test and the test number field reflects the last failing test.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Test passed 1b - Test failed
1 STAT_2	Status of calibration step 2 Status of test *n* (where *n* = 1...12). This field is cleared when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. Bit for each test is updated dynamically during test and the test number field reflects the last failing test. 0b - Test passed 1b - Test failed
0 STAT_1	Status of calibration step 1 Status of test *n* (where *n* = 1...12). This field is cleared when MCR[CALSTART] = 1, which initiates the test. This field updates automatically as the test progresses. Bit for each test is updated dynamically during test and the test number field reflects the last failing test. 0b - Test passed 1b - Test failed

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-290d3ab6401

Chapter 59
Thermal Monitoring Unit (TMU)

59.1 Chip-specific TMU information

59.1.1 Local temperature sensor placements

There are three remote sensors. The table below provides the local temperature sensor placements.

Table 532. Sensor placement

Temperature sensor ID	Placement
0	Between the DDR Sub-System and the internal RAM
1	Near A53 Cluster 0
2	Near LLCE/HSE_H and the Temp Diode

59.1.2 Recommended TCMCFG register settings

The Central Module Configuration Register (TCMCFG) enables you to control accuracy-enhancing features in the TMU, such as offset cancellation mode, dynamic element matching, or dynamic power management, among several others.

It is recommended to make full use of these features to get the most accurate temperature reading possible.

Use the following settings:

- OCS = 0
- OCM = 1
- DEMA = 1
- DFD = 3
- RCTC = 4
- CLKDIV = 4

The combined effect of these settings should leverage the module's ability to increase the number of samples taken from each sensor and fine-tune some of the analog circuitry for optimal performance. The RCTC and CLKDIV settings specifically control the duration of each analog conversion. These settings deliver a stable reading and quick conversion times. See the register description for further details.

59.1.3 Calibration table

To obtain valid temperature results from the TMU in monitoring mode, first initialize the TMU's calibration table.

The contents of the table below have been characterized by NXP and highly recommended. Please do diverge from these settings.

Table 533. Calibration table

TTCFGR[CAL_PNT]	TTRCR[TEMP(K)]	TSCFGR[SENSOR]
0	0x0E9	0x02C
1	0x0E6	0x026
2	0x0F2	0x03D
3	0x10A	0x06B

Table continues on the next page...

Table 533. Calibration table (continued)

TTCFGR[CAL_PNT]	TTRCR[TEMP(K)]	TSCFGR[SENSOR]
4	0x123	0x09B
5	0x13B	0x0C9
6	0x153	0x0F7
7	0x161	0x112 + TMU_CFG_DAC_TRIM_0[4:0]
8	0x16B	0x125 + TMU_CFG_DAC_TRIM_0[4:0]
9	0x184	0x155 + TMU_CFG_DAC_TRIM_0[4:0]
10	0x190	0x16C + TMU_CFG_DAC_TRIM_0[4:0]
11	0x193	0x172 + TMU_CFG_DAC_TRIM_0[4:0]

The table's values should be written to the proper bitfields in the TTCFGR, TTRCR, and TSCFGR registers, as shown. These are generic registers, and their contents will be lost after POR or shutdown events. Always initialize the table before engaging the TMU's monitoring mode.

The last five entries of the table are meant to be used together with a hot temperature trim, which will vary from part to part. This value must be read from the TMU_CFG_DAC_TRIM_0[4:0] fuses, and the end user's software must offset the last five entries of the calibration table with it.

This is not automated. Therefore, the end user's software must fetch the fuse value, add it to the last five entries of the generic table above, and program each part with the newly computed values.

59.2 Introduction

TMU has access to multiple remote temperature measurement sites, which are strategically located on the chip. The module monitors these sites and reports the temperature from one or more of them. You can access the data related to the current and average temperature, which the module constantly captures, for each site. The module can also signal an alarm if a programmed event occurs.

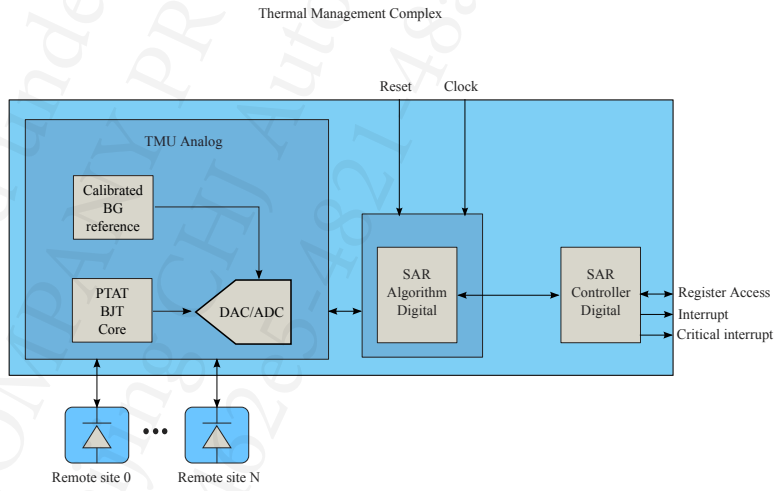


Figure 543. TMU block diagram

59.2.1 Features

TMU includes the following features:

- Temperature measurement ranges from 233 K to 398 K (-40 °C to + 125 °C)
- Configuration
  - TMU software configuration loads and initializes the calibration data
- Monitoring
  - Single- or multi-site monitoring
  - Programmable monitoring interval
  - Out-of-range indication
  - High/low temperature range monitoring
  - Temperature rate change monitoring
  - Immediate and average temperature monitoring
  - Average temperature monitoring based on programmable low-pass filtering
  - Programmable monitoring of high/low temperature thresholds for triggering normal and/or critical alarms
  - Programmable monitoring of the rate of rising/falling temperatures for triggering critical alarms
- Reporting
  - Immediate and average temperature reporting for all monitor sites

59.2.2 Modes of operation

The TMU has the following modes of operation as defined by TMR[MODE]:

- TMU disabled; low-power mode
- Monitoring

59.3 TMU register descriptions

The table below shows the memory map for management of the TMU resources.

59.3.1 TMU memory map

TMU base address: 400A_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Mode (TMR)	32	RW	0000_0000h
4h	Status (TSR)	32	W1C	0000_0000h
8h	Monitor Site (TMSR)	32	RW	0000_0000h
Ch	Monitor Temperature Measurement Interval (TMTMIR)	32	RW	0000_0000h
20h	Interrupt Enable (TIER)	32	RW	0000_0000h
24h	Interrupt Detect (TIDR)	32	W1C	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
30h	Interrupt Immediate Site Capture (TIISCR)	32	RW	0000_0000h
34h	Interrupt Average Site Capture (TIASCR)	32	RW	0000_0000h
38h	Interrupt Critical Site Capture (TICSCR)	32	RW	0000_0000h
40h	Monitor High Temperature Capture (TMHTCR)	32	W1C	0000_0000h
44h	Monitor Low Temperature Capture (TMLTCR)	32	W1C	0000_0000h
48h	Monitor Rising Temperature Rate Capture (TMRTRCR)	32	W1C	0000_0000h
4Ch	Monitor Falling Temperature Rate Capture (TMFTRCR)	32	W1C	0000_0000h
50h	Monitor High Temperature Immediate Threshold (TMHTITR)	32	RW	0000_0000h
54h	Monitor High Temperature Average Threshold (TMHTATR)	32	RW	0000_0000h
58h	Monitor High Temperature Average Critical Threshold (TMHTACTR)	32	RW	0000_0000h
60h	Monitor Low Temperature Immediate Threshold (TMLTITR)	32	RW	0000_0000h
64h	Monitor Low Temperature Average Threshold (TMLTATR)	32	RW	0000_0000h
68h	Monitor Low Temperature Average Critical Threshold (TMLTACTR)	32	RW	0000_0000h
70h	Monitor Rising Temperature Rate Critical Threshold (TMRTRCTR)	32	RW	0000_0000h
74h	Monitor Falling Temperature Rate Critical Threshold (TMFTRCTR)	32	RW	0000_0000h
80h	Temperature Configuration (TTCFGR)	32	RW	0000_0000h
84h	Sensor Configuration (TSCFGR)	32	RW	0000_0000h
100h	Report Immediate Temperature at Site (TRITSR0)	32	RO	0000_0000h
104h	Report Average Temperature at Site (TRATSR0)	32	RO	0000_0000h
110h	Report Immediate Temperature at Site (TRITSR1)	32	RO	0000_0000h
114h	Report Average Temperature at Site (TRATSR1)	32	RO	0000_0000h
120h	Report Immediate Temperature at Site (TRITSR2)	32	RO	0000_0000h
124h	Report Average Temperature at Site (TRATSR2)	32	RO	0000_0000h
F00h	Central Module Configuration (TCMCFG)	32	RW	See description
F10h - F4Ch	Temperature Range Control a (TTRCR0 - TTRCR15)	32	RW	0000_0000h

59.3.2 Mode (TMR)

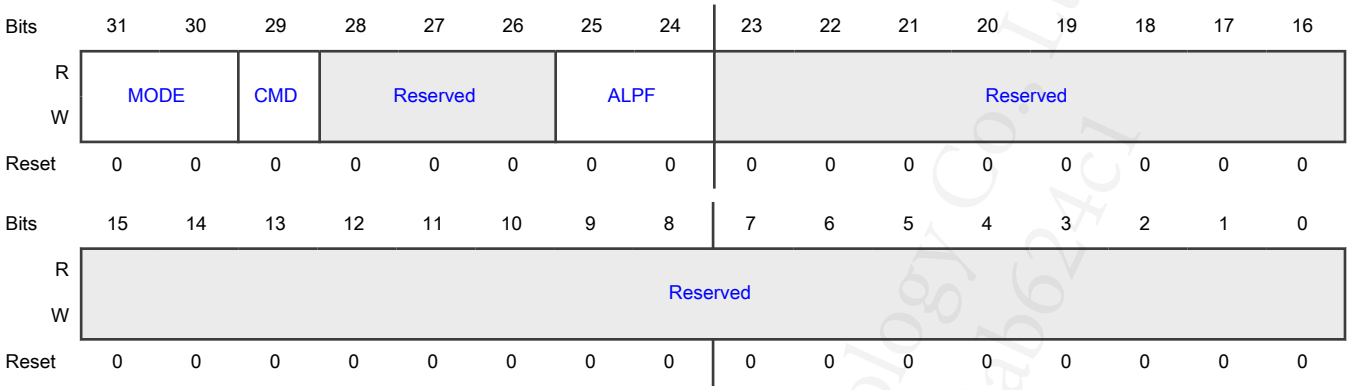
Offset

Register	Offset
TMR	0h

Function

This register allows the software to control the thermal monitoring operation. You must wait for at least 500 µs before changing modes.

Diagram



Fields

Field	Function
31-30 MODE	Mode Before you enable the TMU for monitoring, you must first configure it; see Initialization information. Failure to properly initialize the configuration table may result in undefined behavior. 00b - Idle; low-power mode 01b - Reserved 10b - Monitoring of sites as defined by TMSR[SITE] 11b - Reserved
29 CMD	Central module disable Before disabling the central module, you must first deassert the MODE field and verify that the TMU is idle. You must then enable the circuit during normal operation by writing an appropriate non-zero value in the MODE field. This field allows for statically disabling the centralized circuit and the associated digital logic to reduce power. If the central circuit is re-enabled, TMU must wait before operations can continue as defined by TCMCFG[CMET]. 0b - Enabled 1b - Disabled
28-26 —	Reserved
25-24 ALPF	Average low pass filter setting The average temperature is calculated using this formula: ALPF x Current_Temp + (1 - ALPF) x Average_Temp. If no previous (average) temperature is valid, then the module uses the current temperature for Average_Temp in this calculation. For proper operation, you must change this field only when monitoring is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	00b - 1.0 01b - 0.5 10b - 0.25 11b - 0.125
23-0 —	Reserved

59.3.3 Status (TSR)

Offset

Register	Offset
TSR	4h

Function

This register reports the monitoring and calibration status during operation.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TB	MIE	ORL	ORH	Reserved											
W		W1C	W1C	W1C												
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 TB	TMU busy In monitoring mode, this indicates that a temperature measurement is pending. 0b - TMU is idle 1b - TMU is busy

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c62e5-4821-4888-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
30 MIE	Monitoring interval exceeded The time required to perform measurement of all monitored sites has exceeded the monitoring interval as defined by Monitor Temperature Measurement Interval (TMTMIR). 0b - Monitoring interval not exceeded 1b - Monitoring interval exceeded
29 ORL	Out-of-range low temperature measurement A temperature sensor detected a temperature reading below the lowest measurable temperature of 233 K (-40 °C). 0b - No out-of-range low temperature measurement detected 1b - Out-of-range low temperature measurement detected
28 ORH	Out-of-range high temperature measurement A temperature sensor detected a temperature reading above the highest measurable temperature of 398 K (125 °C). 0b - No out-of-range high temperature measurement detected 1b - Out-of-range high temperature measurement detected
27-0 —	Reserved

59.3.4 Monitor Site (TMSR)

Offset

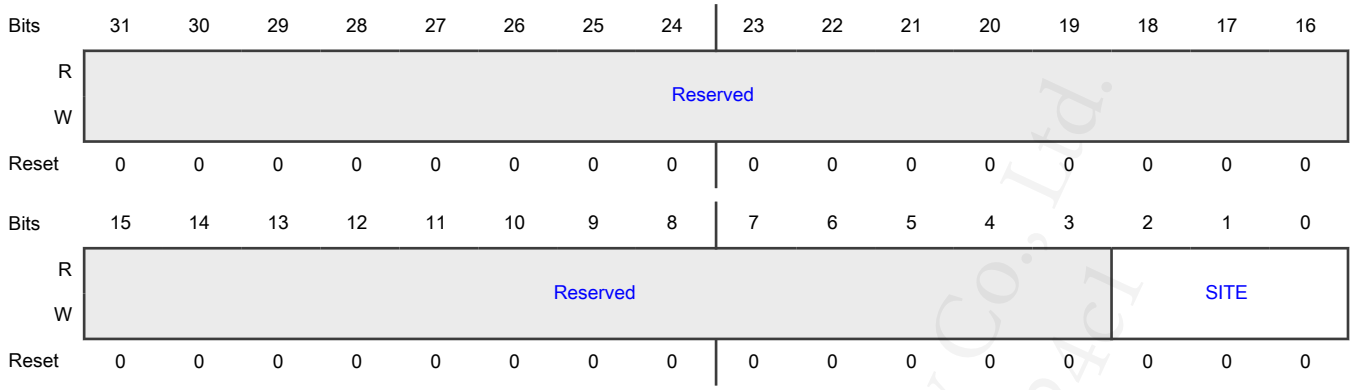
Register	Offset
TMSR	8h

Function

This register indicates which remote site sensor is actively being monitored by TMU when TMR[MODE]=b10.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-590d3ab624c1

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 SITE	Monitoring site select 2-0 When you set the select bit for a temperature sensor site, it is enabled and included in all monitoring functions. For proper operation, you should only change this field when monitoring is disabled. Multiple sites can be enabled at the same time. SITE[x] controls monitoring of site x, such as: • SITE[0] - controls monitoring site 0 — SITE[0]==0: Disables site 0 — SITE[0]==1: Enables site 0 • SITE[1] - controls monitoring site 1 — SITE[1]==0: Disables site 1 — SITE[1]==1: Enables site 1 • SITE[2] - controls monitoring site 2 — SITE[2]==0: Disables site 2 — SITE[2]==1: Enables site 2

59.3.5 Monitor Temperature Measurement Interval (TMTMIR)

Offset

Register	Offset
TMTMIR	Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Function

This register allows you to control at what frequency the module reads the temperature sensors. The module reads all enabled monitored sites once in this time interval. The value of the status field, TSR[MIE], is 1 if the temperature measument takes longer than the set interval. The software should consider increasing the interval or reducing the number of active sites if the interval exceeds the temperature monitoring interval determined by the TMI field value in this register. If you disable the interval, then monitoring is continuous.

NOTE

The time it takes for one temperature measurement depends on the temperatures measured and mode settings; thus, the time for a single temperature measurement is not fixed.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved												TMI			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3-0 TMI	Temperature monitoring interval in seconds For proper operation, you should only change this field when monitoring is disabled, that is, when TMR[MODE]=b00. For the TMU frequencies calculation, use the following formula for the field description: • $2^{(23+TMI)}$ × TMU clock period in seconds For the monitoring interval calculation, use the following formula: • 0b – 1110b: Interval(s) = $2^{(23+TMI)}$ × TMU clock period in seconds • 1111b: Continuous monitoring mode

59.3.6 Interrupt Enable (TIER)

Offset

Register	Offset
TIER	20h

Function

This register determines if a detected status condition should cause a system interrupt. A system interrupt occurs if the value of any field in this register is 1 and the corresponding field in Interrupt Detect (TIDR) is 1. To clear the interrupt, you write a 1 to Interrupt Detect (TIDR). The following table provides the details of TMU interrupts.

Table 534. TMU interrupts

Interrupt description	Enable field	Reported as	
		Interrupt	Critical interrupt
Immediate high temperature threshold interrupt enable	IHTTIE	Y	—
Immediate low temperature threshold interrupt enable	ILTTIE	Y	—
Average high temperature threshold interrupt enable	AHTTIE	Y	—
Average low temperature threshold interrupt enable	ALTTIE	Y	—
Average high temperature critical threshold interrupt enable	AHTCTIE	—	Y
Average low temperature critical threshold interrupt enable	ALTCTIE	—	Y
Rising temperature rate critical threshold interrupt enable	RTRCTIE	—	Y
Falling temperature rate critical threshold interrupt enable	FTRCTIE	—	Y

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IHTTIE	AHTTIE	AHTCTIE	ILTTIE	ALTTIE	ALTCTIE	RTRCTIE	FTRCTIE	Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 IHTTIE	Immediate high temperature threshold interrupt enable 0b - Disabled 1b - Interrupt enabled; generates an interrupt if TIDR[IHTT] is 1
30 AHTTIE	Average high temperature threshold interrupt enable 0b - Disabled 1b - Interrupt enabled; generates an interrupt if TIDR[AHTT] is 1
29	Average high temperature critical threshold interrupt enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
AHTCTIE	0b - Disabled 1b - Interrupt enabled; generates an interrupt if TIDR[AHTCT] is 1
28 ILTTIE	Immediate low temperature threshold interrupt enable 0b - Disabled 1b - Interrupt enabled; generates an interrupt if TIDR[ILTT] is 1
27 ALTTIE	Average low temperature threshold interrupt enable 0b - Disabled 1b - Interrupt enabled; generates an interrupt if TIDR[ALTT] is 1
26 ALTCTIE	Average low temperature critical threshold interrupt enable 0b - Disabled 1b - Interrupt enabled; generates an interrupt if TIDR[ALTCT] is 1
25 RTRCTIE	Rising temperature rate critical threshold interrupt enable 0b - Disabled 1b - Interrupt enabled; generates an interrupt if TIDR[RTRCT] is 1
24 FTRCTIE	Falling temperature rate critical threshold interrupt enable 0b - Disabled 1b - Interrupt enabled; generates an interrupt if TIDR[FTRCT] is 1
23-0 —	Reserved

59.3.7 Interrupt Detect (TIDR)

Offset

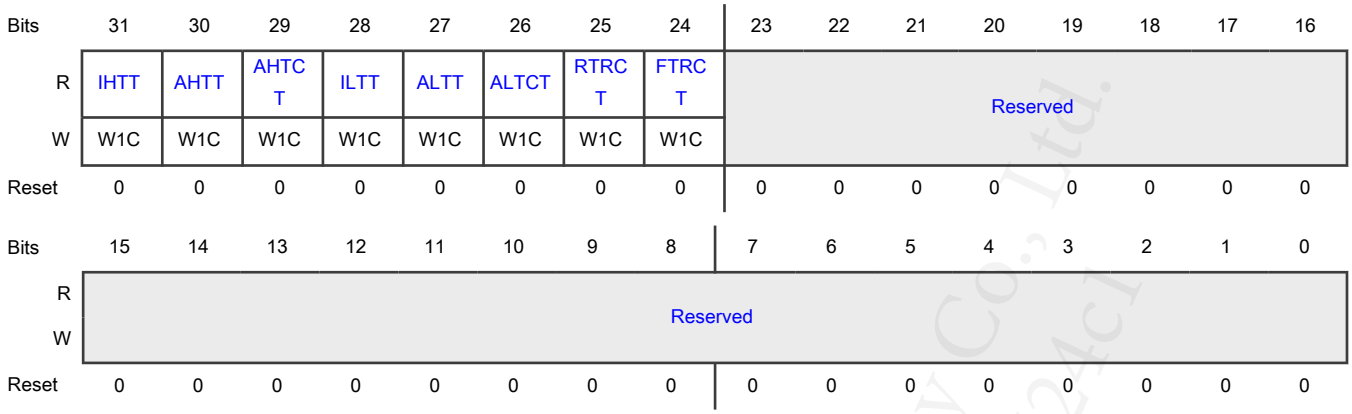
Register	Offset
TIDR	24h

Function

This register indicates if a status condition is detected that could generate an interrupt. If enabled, you write a 1 to clear the detected condition and the interrupt.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-9251-390d3ab624c1

Diagram



Fields

Field	Function
31 IHTT	Immediate high temperature threshold exceeded Write 1 to clear. 0b - No threshold exceeded 1b - One or more monitored sites exceed the immediate high temperature threshold, as defined by Monitor High Temperature Immediate Threshold (TMHTITR). This includes an out-of-range measured temperature above 398 K (125 °C). TIISCR[SITE] captures the sites that exceed the threshold.
30 AHTT	Average high temperature threshold exceeded Write 1 to clear. 0b - No threshold exceeded 1b - One or more monitored sites exceed the average high temperature threshold, as defined by Monitor High Temperature Average Threshold (TMHTATR). TIASCR[SITE] captures the sites that exceed the threshold.
29 AHTCT	Average high temperature critical threshold exceeded Write 1 to clear. 0b - No threshold exceeded 1b - One or more monitored sites exceed the average high temperature critical threshold, as defined by Monitor High Temperature Average Critical Threshold (TMHTACTR). TICSCR[SITE] captures the sites that exceed the threshold.
28 ILTT	Immediate low temperature threshold Write 1 to clear. 0b - No threshold exceeded 1b - One or more monitored sites pass the immediate low temperature threshold, as defined by Monitor Low Temperature Immediate Threshold (TMLTITR). This includes an out-of-range

Table continues on the next page...

Table continued from the previous page...

Field	Function
	measured temperature below 233 K (-40 °C). TIISCR[SITE] captures the sites that pass the threshold.
27 ALTT	Average low temperature threshold Write 1 to clear. 0b - No threshold exceeded 1b - One or more monitored sites pass the average low temperature threshold, as defined by Monitor Low Temperature Average Threshold (TMLTATR). TIASCR[SITE] captures the sites that pass the threshold.
26 ALTCT	Average low temperature critical threshold Write 1 to clear. 0b - No threshold exceeded 1b - One or more monitored sites pass the average low temperature threshold, as defined by Monitor Low Temperature Average Critical Threshold (TMLTACTR). TICSCR[SITE] captures the sites that pass the threshold.
25 RTRCT	Rising temperature rate critical threshold Write 1 to clear. 0b - No threshold exceeded 1b - One or more monitored sites pass the rising temperature rate critical threshold, as defined by Monitor Rising Temperature Rate Critical Threshold (TMRTRCTR). TICSCR[SITE] captures the sites that exceed the threshold.
24 FTRCT	Falling temperature rate critical threshold Write 1 to clear. 0b - No threshold exceeded 1b - One or more monitored sites exceed the falling temperature rate critical threshold, as defined by Monitor Falling Temperature Rate Critical Threshold (TMFTRCTR). TICSCR[SITE] captures the sites that exceed the threshold.
23-0 —	Reserved

59.3.8 Interrupt Immediate Site Capture (TIISCR)

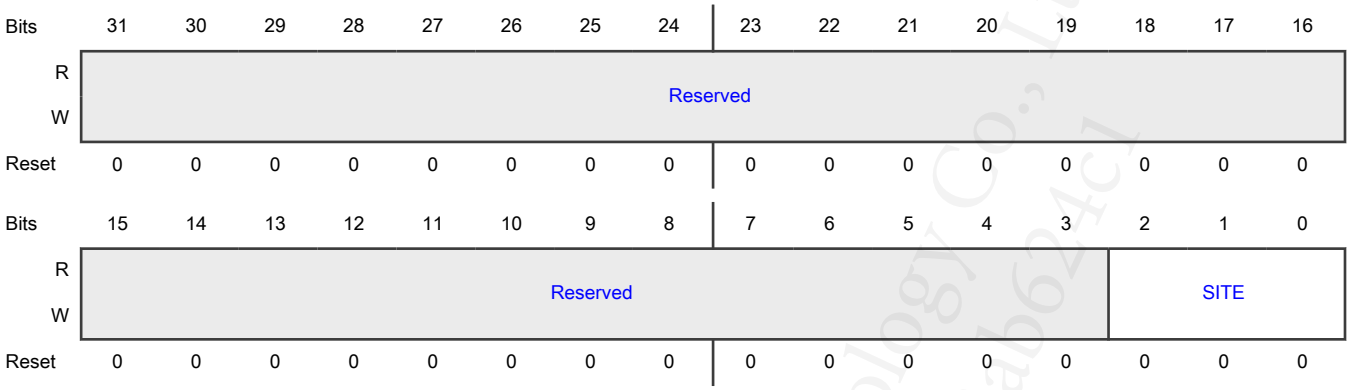
Offset

Register	Offset
TIISCR	30h

Function

This register indicates which remote sensor sites are associated with the detection of an interrupt event related to current reported temperatures.

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 SITE	Temperature sensor site Temperature sensor site associated with the setting of TIDR[IHTT/ILTT]. This field has the same bit representation as TMSR[SITE]. The software must clear this field after handling the detected interrupt events IHTT/ILTT.

59.3.9 Interrupt Average Site Capture (TIASCR)

Offset

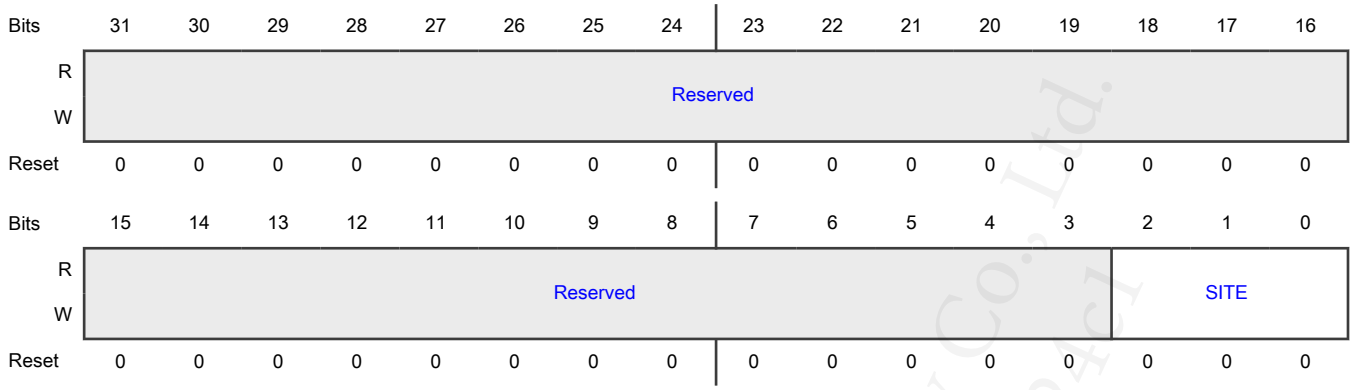
Register	Offset
TIASCR	34h

Function

This register indicates which remote sensor sites are associated with the detection of an interrupt event related to average reported temperatures.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-9251-390d3ab624c1

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 SITE	Temperature sensor site Temperature sensor site associated with the setting of TIDR[AHTT] and TIDR[ALTT]. This field has the same bit representation as TMSR[SITE]. The software must clear this field after handling the detected interrupt event AHTT/ALTT.

59.3.10 Interrupt Critical Site Capture (TICSCR)

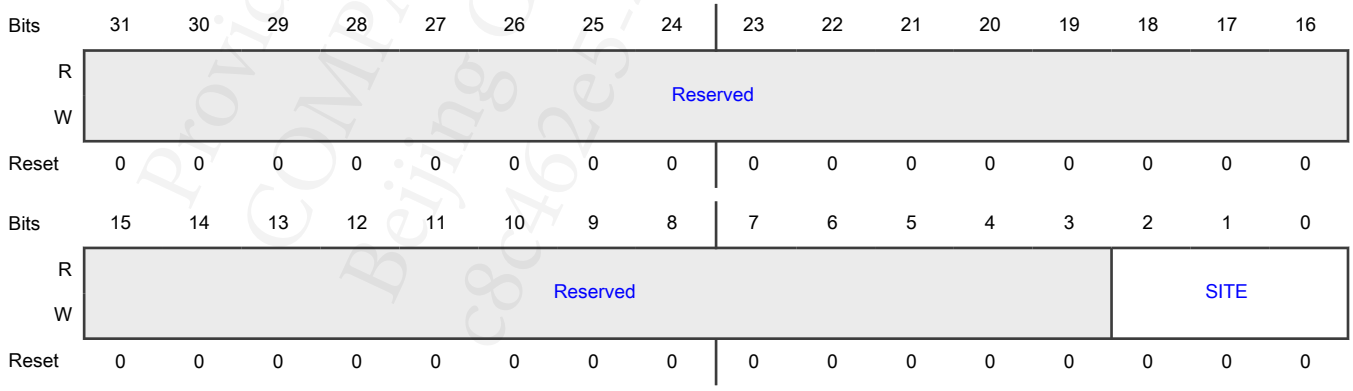
Offset

Register	Offset
TICSCR	38h

Function

This register indicates which remote sensor sites are associated with the detection of a critical interrupt event.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8462e5-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-3 —	Reserved
2-0 SITE	Temperature sensor site Temperature sensor site associated with the setting of TIDR[RCTC/AHTCT/ALTCT]. This field has the same bit representation as TMSR[SITE]. The software must clear this field after handling the detected critical interrupt event FTRCT/RTRCT/AHTCT/ALTCT.

59.3.11 Monitor High Temperature Capture (TMHTCR)

Offset

Register	Offset
TMHTCR	40h

Function

This register captures and records the highest temperature for any one enabled monitored site within the temperature sensor range.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	V	Reserved														
W	W1C															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						TP5	TEMP								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 V	Valid reading Reenabling the TMU automatically clears this field and starts a new search. 0b - Temperature reading is not valid due to no measured temperature within the sensor range of 233 K to 398 K for an enabled monitored site

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Temperature reading is valid
30-10 —	Reserved
9 TP5	Highest temperature recorded in Kelvin by any enabled monitored site Valid when V=1. When this field is 1, you must add 0.5 K to the temperature that TEMP reports. For example, if TEMP is 300 K and TP5=1, then the final temperature is 300.5 K. Sensor range is 233 K to 398 K.
8-0 TEMP	Highest temperature recorded in Kelvin by any enabled monitored site Valid when V=1. Sensor range is 233 K to 398 K.

59.3.12 Monitor Low Temperature Capture (TMLTCR)

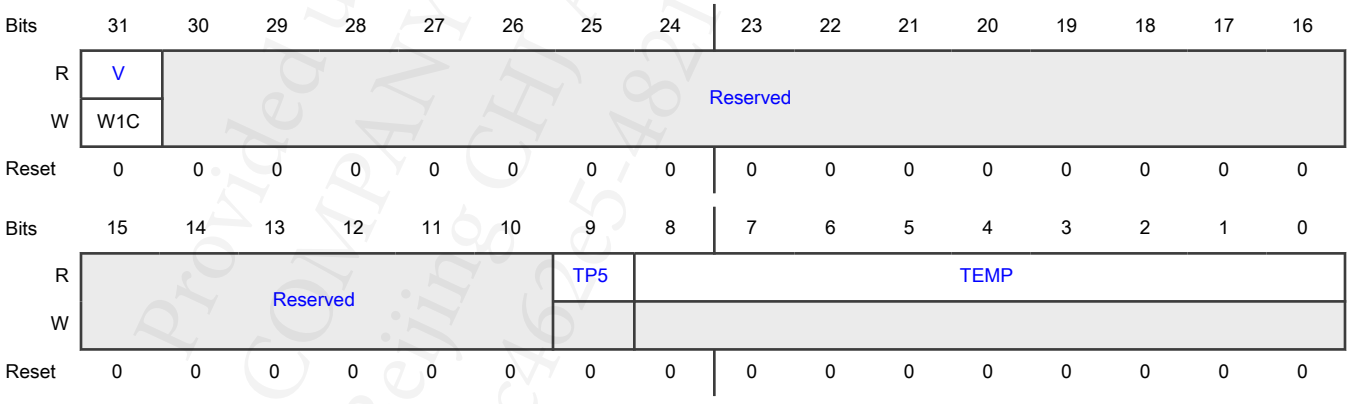
Offset

Register	Offset
TMLTCR	44h

Function

This register captures and records the lowest temperature reached for any one enabled monitored site within the temperature sensor range.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c1625-4821-4818-b251-390d3ab624c1

Fields

Field	Function
31 V	Valid reading Reenabling the TMU automatically clears this field and starts a new search. 0b - Temperature reading is not valid because of no measured temperature within the sensor range of 233 K to 398 K for an enabled monitored site. 1b - Temperature reading is valid.
30-10 —	Reserved
9 TP5	Lowest temperature in Kelvin that any enabled monitored site records Valid when V=1. When this field is 1, you must add 0.5 K to the temperature that TEMP reports. For example, if TEMP is 300 K and TP5=1, then the final temperature is 300.5 K. Sensor range is 233 K to 398 K.
8-0 TEMP	Lowest temperature in Kelvin that any enabled monitored site records Valid when V=1. Sensor range is 233 K to 398 K.

59.3.13 Monitor Rising Temperature Rate Capture (TMRTRCR)

Offset

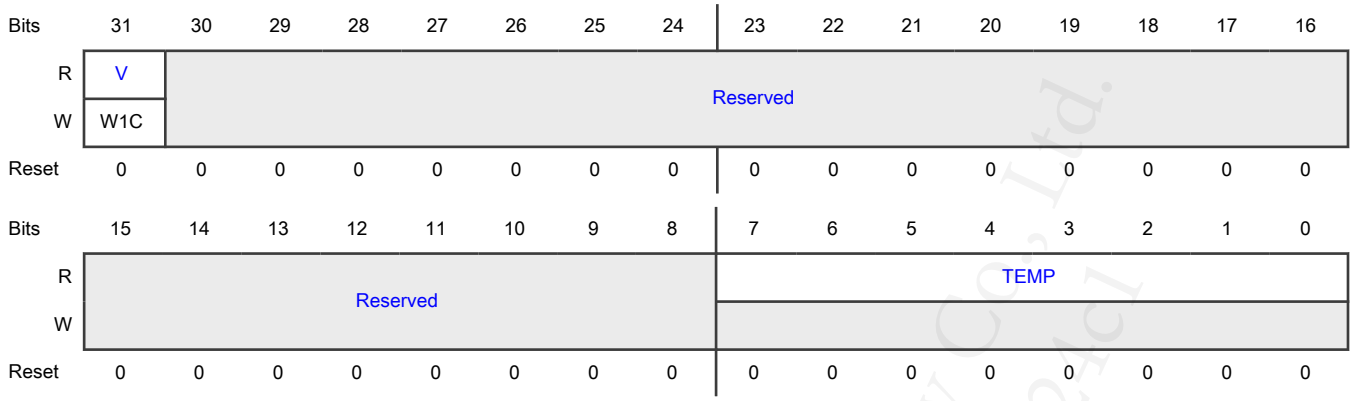
Register	Offset
TMRTRCR	48h

Function

This register captures and records the highest temperature rising rate change between two successive measurements from the same monitored site based on the monitoring interval setting.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHS Automotive Technology Co., Ltd.
c8c462e5-4821-4828-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 V	Valid reading Reenabling the TMU automatically clears this field and starts a new search. 0b - Temperature reading is not valid because of no two successive measured temperatures within the sensor range of 233 K to 398 K for an enabled monitored site 1b - Temperature reading is valid
30-8 —	Reserved
7-0 TEMP	Highest rising temperature rate change in Kelvin that any enabled monitored site records Valid when V=1. Rate change is between 0 K to 200 K.

59.3.14 Monitor Falling Temperature Rate Capture (TMFTRCR)

Offset

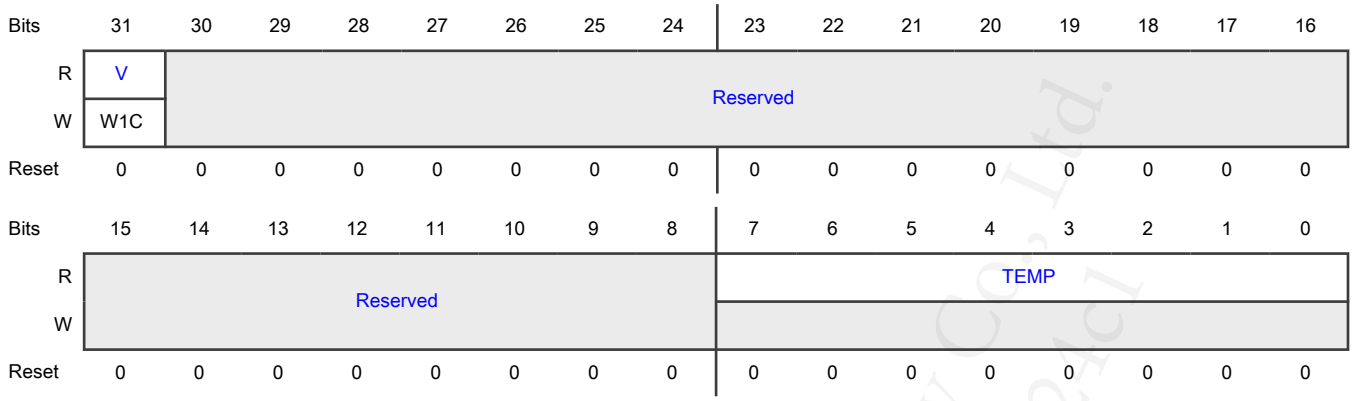
Register	Offset
TMFTRCR	4Ch

Function

This register captures and records the highest temperature falling rate change between two successive measurements from the same monitored site based on the monitoring interval setting.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5e5-4821-48a8-b251-30ed3ab624c1

Diagram



Fields

Field	Function
31 V	Valid reading Reenabling the TMU automatically clears this field and starts a new search. 0b - Temperature reading is not valid because of no two successive measured temperatures within the sensor range of 233 K to 398 K for an enabled monitored site 1b - Temperature reading is valid
30-8 —	Reserved
7-0 TEMP	Highest falling temperature rate change in Kelvin that any enabled monitored site records Valid when V=1. Rate change is a positive number between 0 K to 200 K.

59.3.15 Monitor High Temperature Immediate Threshold (TMHTITR)

Offset

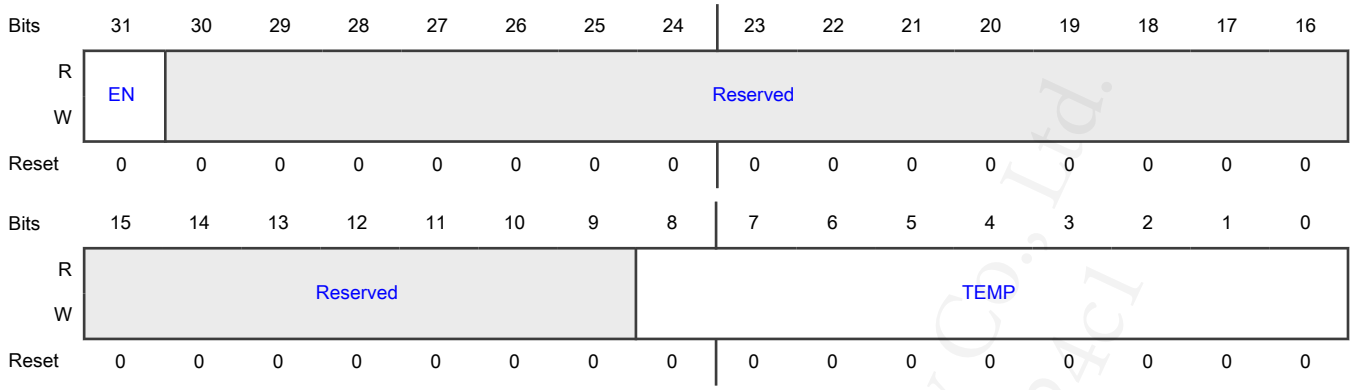
Register	Offset
TMHTITR	50h

Function

This register determines the current high-temperature threshold for generating the TIDR[IHTT] event.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e54821-4828-9251-30d8ab624c1

Diagram



Fields

Field	Function
31 EN	Enable threshold 0b - Disabled 1b - Threshold enabled
30-9 —	Reserved
8-0 TEMP	High temperature immediate threshold value Determines the current high-temperature threshold for any enabled monitored site. If this threshold is exceeded, the module sets the value of TIDR[IHTT] to 1 if the value of the corresponding EN field is 1. Sensor range is 233 K to 398 K.

59.3.16 Monitor High Temperature Average Threshold (TMHTATR)

Offset

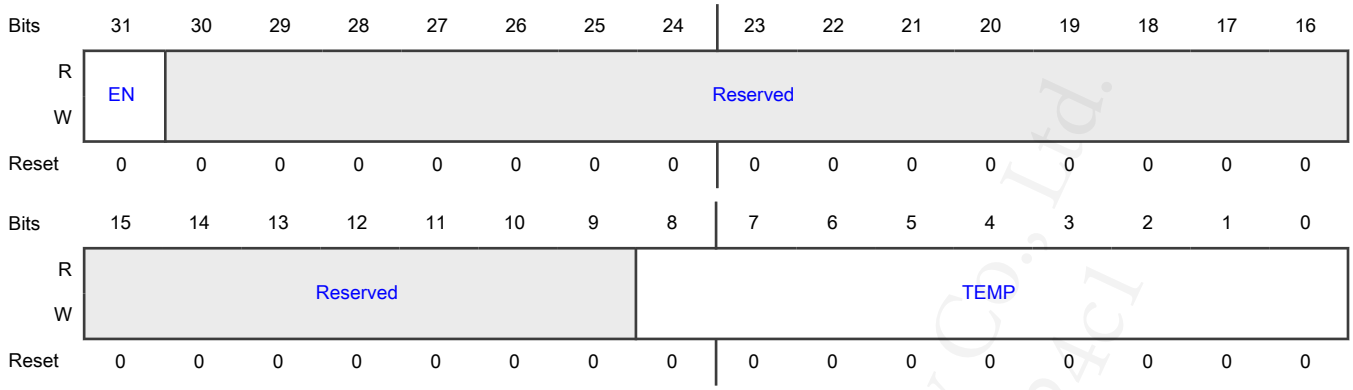
Register	Offset
TMHTATR	54h

Function

This register determines the average high temperature threshold for generating the TIDR[AHTT] event. The low-pass filter setting, TMR[ALPF], determines the function for calculating the average temperature.

Preliminary Draft Only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co. Ltd.
c8c4625-4821-4888-9251-390d3ab624c1

Diagram



Fields

Field	Function
31 EN	Enable threshold 0b - Disabled 1b - Threshold enabled
30-9 —	Reserved
8-0 TEMP	High temperature average threshold value Determines the current high-temperature threshold for any enabled monitored site. If this threshold is exceeded, the module sets the value of TIDR[AHTT] to 1 if the value of the corresponding EN field is 1. Sensor range is 233 K to 398 K.

59.3.17 Monitor High Temperature Average Critical Threshold (TMHTACTR)

Offset

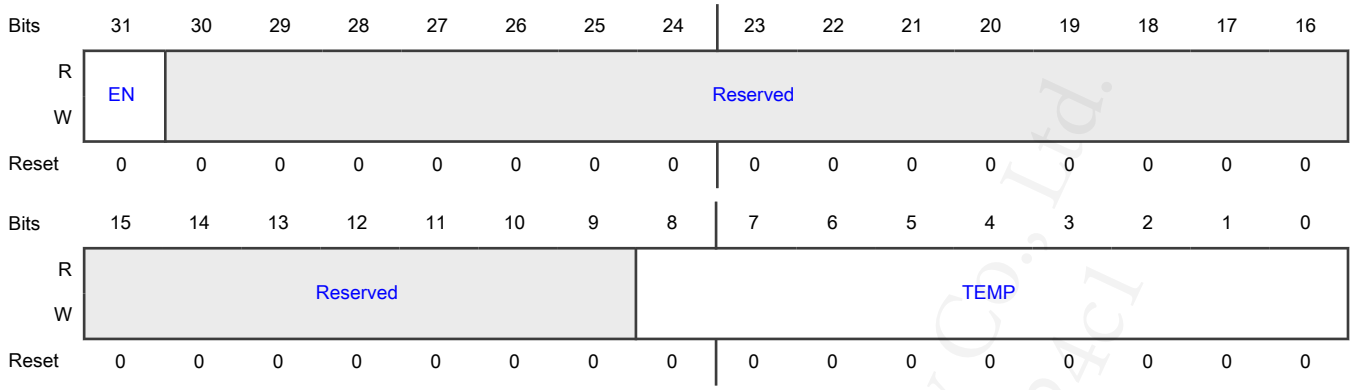
Register	Offset
TMHTACTR	58h

Function

This register determines the average critical high temperature threshold for generating the TIDR[AHTCT] event. The low-pass filter setting, TMR[ALPF], determines the function for calculating the average temperature.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c4625-4821-48a8-a251-390d3ab624c1

Diagram



Fields

Field	Function
31 EN	Enable threshold 0b - Disabled 1b - Threshold enabled
30-9 —	Reserved
8-0 TEMP	High temperature average critical threshold value Determines the average critical high temperature threshold, for any enabled monitored site, that if exceeded will cause TIDR[AHTCT] to be set when EN=1. Sensor range is 233 K to 398 K.

59.3.18 Monitor Low Temperature Immediate Threshold (TMLTITR)

Offset

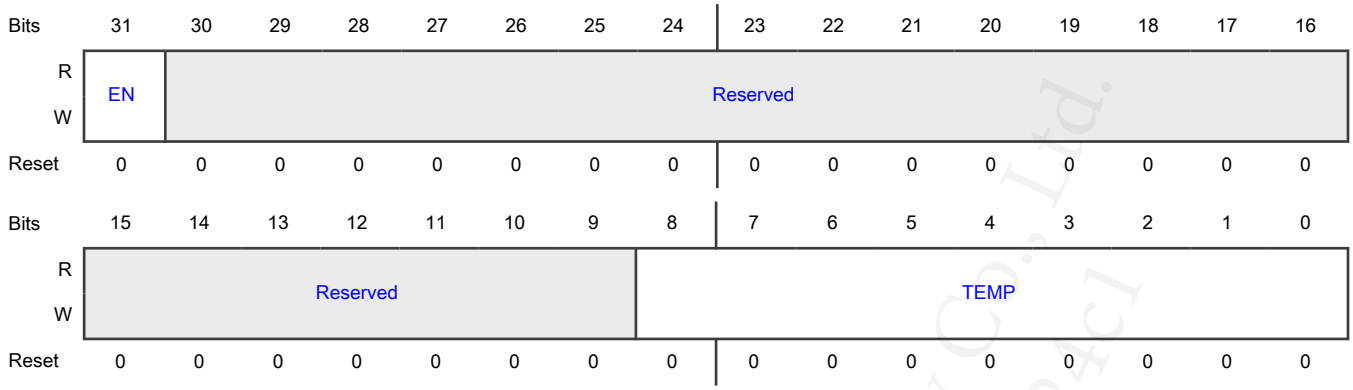
Register	Offset
TMLTITR	60h

Function

This register determines the current low temperature threshold for generating the TIDR[ILTT] event.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-4a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 EN	Enable threshold 0b - Disabled 1b - Threshold enabled
30-9 —	Reserved
8-0 TEMP	Low temperature immediate threshold value Determines the current low temperature threshold for any enabled monitored site. If the temperature is below the threshold it will cause TIDR[ILTT] to be set when EN=1. Sensor range is 233 K to 398 K.

59.3.19 Monitor Low Temperature Average Threshold (TMLTATR)

Offset

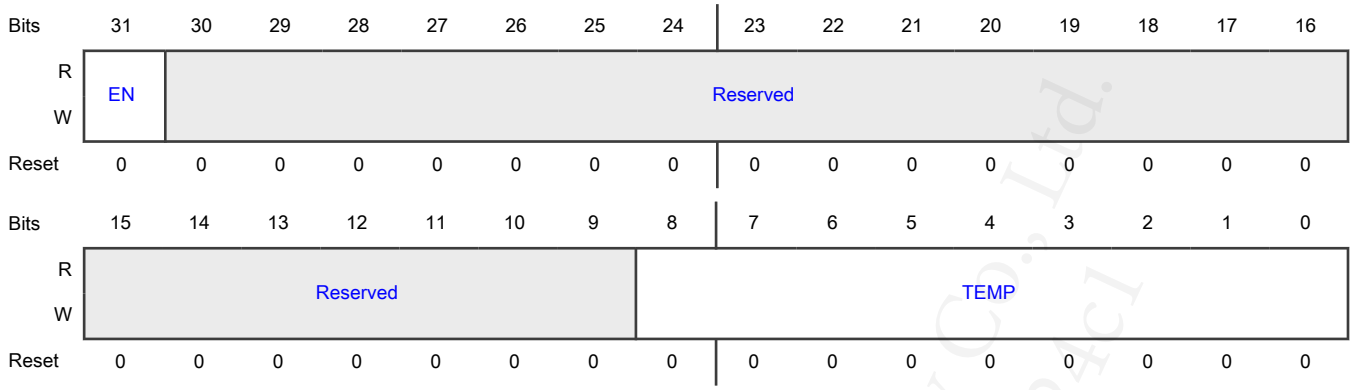
Register	Offset
TMLTATR	64h

Function

This register determines the average low temperature threshold for generating the TIDR[ALTT] event. The low-pass filter setting, TMR[ALPF], determines the function for calculating the average temperature.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c4625-4821-4848-9251-390d3ab624c1

Diagram



Fields

Field	Function
31 EN	Enable threshold 0b - Disabled 1b - Threshold enabled
30-9 —	Reserved
8-0 TEMP	Low temperature average threshold value Determines the average low temperature threshold for any enabled monitor site. If the temperature is below the threshold it will cause TIDR[ALTT] to be set when EN=1. Sensor range is 233 K to 398 K.

59.3.20 Monitor Low Temperature Average Critical Threshold (TMLTACTR)

Offset

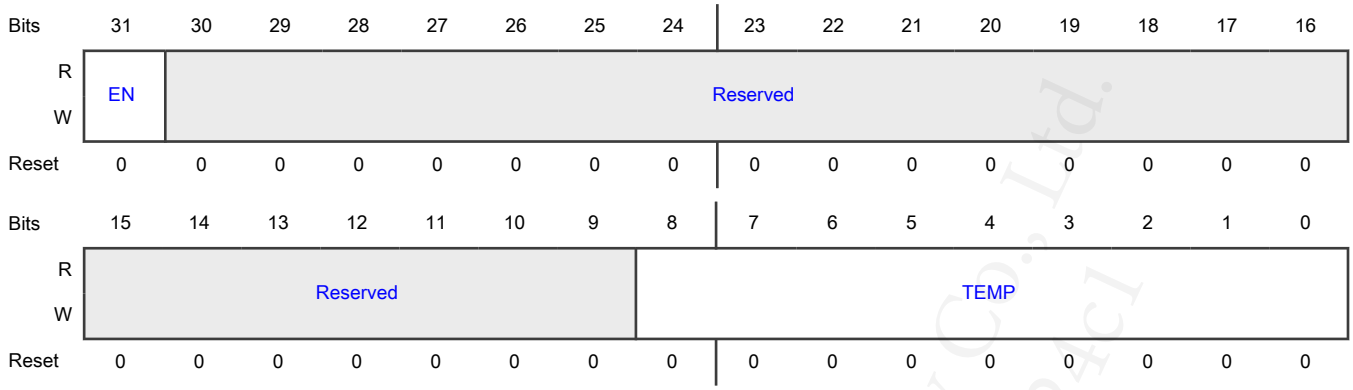
Register	Offset
TMLTACTR	68h

Function

This register determines the average critical low temperature threshold for generating the TIDR[ALTCT] event. The low-pass filter setting, TMR[ALPF], determines the function for calculating the average temperature.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c4625-4821-4848-9251-390d3ab624c1

Diagram



Fields

Field	Function
31 EN	Enable threshold 0b - Disabled 1b - Threshold enabled
30-9 —	Reserved
8-0 TEMP	Low temperature average critical threshold value Determines the average critical high temperature threshold, for any enabled monitored site, that if passed will cause TIDR[ALTCT] to be set when EN=1. Sensor range is 233 K to 398 K.

59.3.21 Monitor Rising Temperature Rate Critical Threshold (TMRTRCTR)

Offset

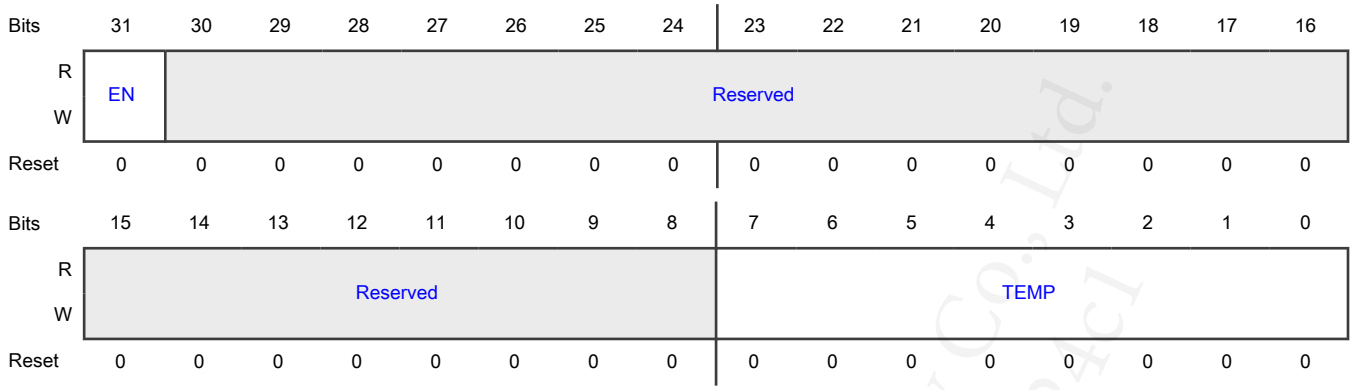
Register	Offset
TMRTRCTR	70h

Function

This register is used to detect a fast-rising temperature by calculating the difference between two valid successive measurements per site, based on the monitoring interval setting. This could indicate a critical event such as heatsink separation or cooling fan malfunction.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 EN	Enable threshold 0b - Disabled 1b - Threshold enabled
30-8 —	Reserved
7-0 TEMP	Temperature difference between two measurements from the same site This is a number between 0 K and 200 K. If this threshold is exceeded, the module sets the value of TIDR[RTRCT] to 1.

59.3.22 Monitor Falling Temperature Rate Critical Threshold (TMFTRCTR)

Offset

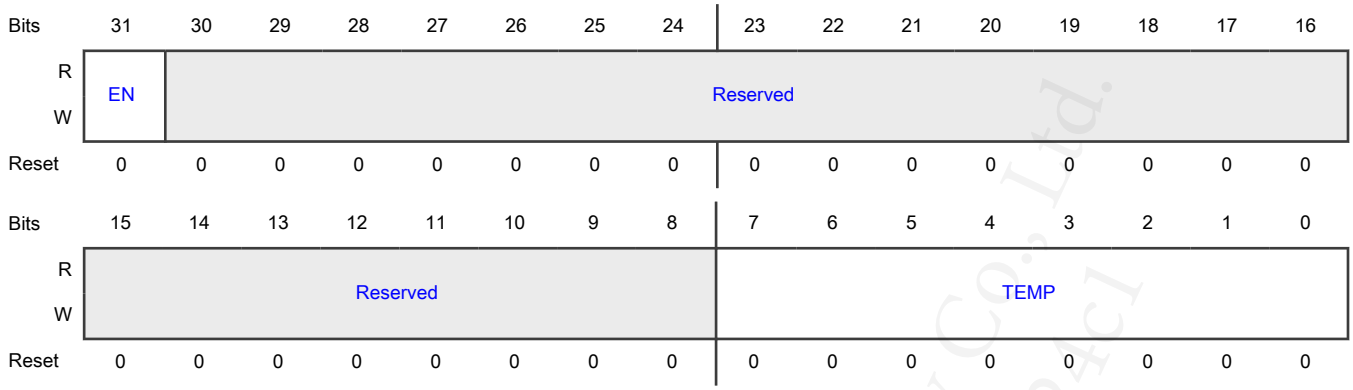
Register	Offset
TMFTRCTR	74h

Function

This register is used to detect a fast-falling temperature by calculating the difference between two valid successive measurements per site, based on the monitoring interval setting. Use case may include detecting stress to the chip because of excessive cooling, or cold-boot attacks on volatile memory.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 EN	Enable threshold 0b - Disabled 1b - Threshold enabled
30-8 —	Reserved
7-0 TEMP	Temperature difference between two measurements from the same site This is a (positive) number between 0 K and 200 K. If this threshold is exceeded, the module sets the value of TIDR[FTRCT] to 1.

59.3.23 Temperature Configuration (TTCFGR)

Offset

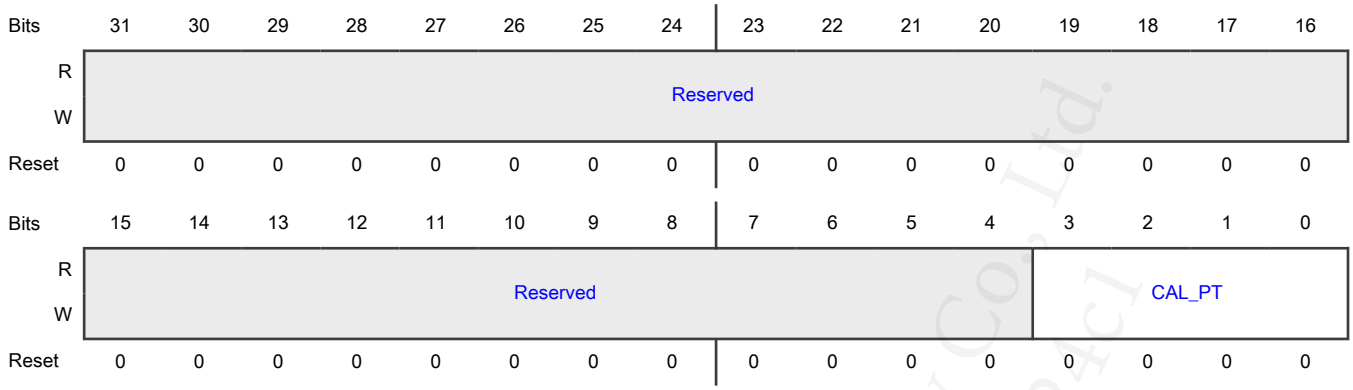
Register	Offset
TTCFGR	80h

Function

This register, in conjunction with Sensor Configuration (TSCFGR), is used to initialize the internal sensor translation table used during monitoring. The register pair defines an indirect access to the table.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 CAL_PT	Calibration point CAL_PT 0 is the lowest calibrated temperature. Each calibration point is related to the TTRCRn registers as shown below.

CAL_PT	Pointer to TTRCRn
0	TTRCR0
1	TTRCR1
2	TTRCR2
3	TTRCR3
4	TTRCR4
5	TTRCR5
6	TTRCR6
7	TTRCR7
8	TTRCR8
9	TTRCR9
10	TTRCR10
11	TTRCR11
12	TTRCR12
13	TTRCR13
14	TTRCR14
15	TTRCR15

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
9c4625-4821-4848-b251-390d3ab624c1

59.3.24 Sensor Configuration (TSCFGR)

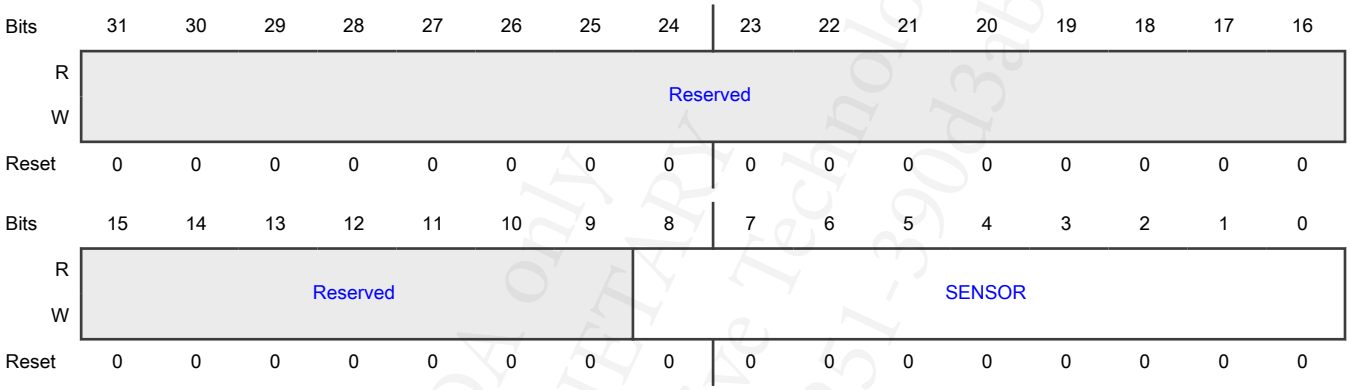
Offset

Register	Offset
TSCFGR	84h

Function

This register, in conjunction with Temperature Configuration (TTCFGR), is used to initialize the internal sensor translation table used during monitoring. The register pair defines indirect access to the table. Reading Sensor Configuration (TSCFGR) returns the data from the translation table as defined in Temperature Configuration (TTCFGR).

Diagram



Fields

Field	Function
31-9 —	Reserved
8-0 SENSOR	Sensor value

59.3.25 Report Immediate Temperature at Site (TRITSR0 - TRITSR2)

Offset

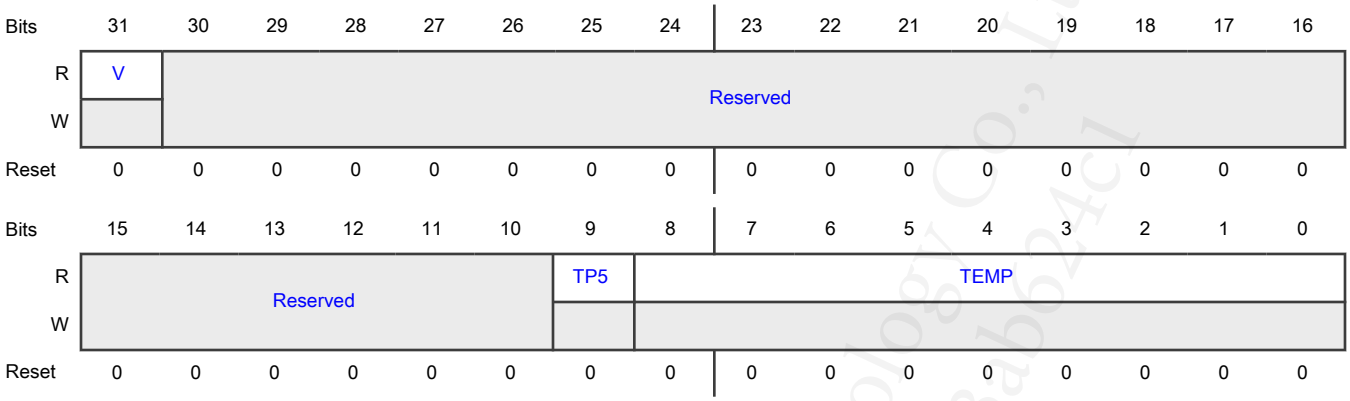
Register	Offset
TRITSR0	100h
TRITSR1	110h
TRITSR2	120h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-821-48a8-b251-590d3ab62461

Function

This register returns the last measured temperature at the site, which must be a part of the list of enabled monitored sites as defined by TMSR[SITE].

Diagram



Fields

Field	Function
31 V	Valid measured temperature 0b - Not valid; temperature is out of sensor range or the first measurement is still pending 1b - Valid
30-10 —	Reserved
9 TP5	Last temperature reading in Kelvin at the site Valid when V=1. When this field is 1, you must add 0.5 K to the temperature that TEMP reports. For example, if TEMP is 300 K and TP5=1, then the final temperature is 300.5 K. Sensor range is 233 K to 398 K.
8-0 TEMP	Last temperature reading in Kelvin at the site Valid when V=1.

59.3.26 Report Average Temperature at Site (TRATSR0 - TRATSR2)

Offset

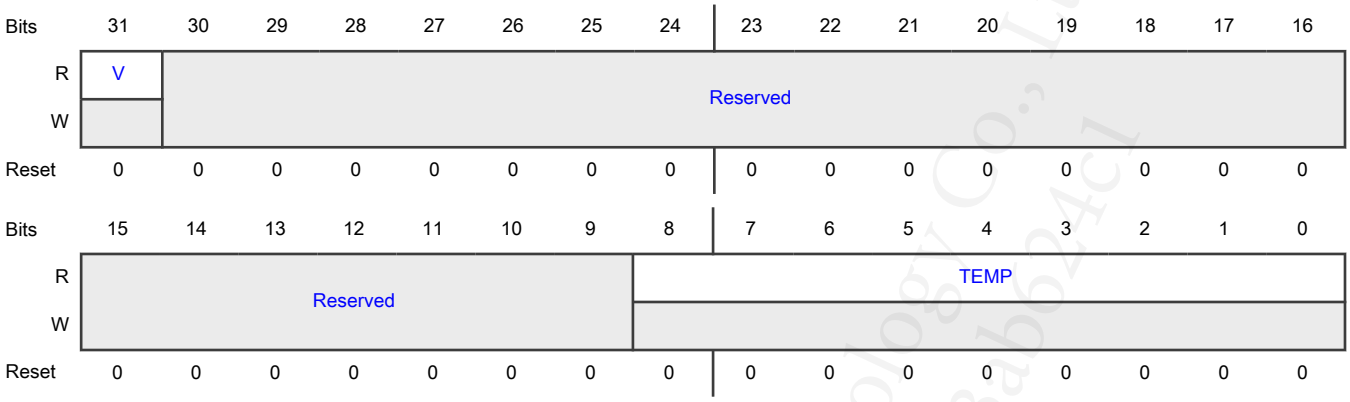
Register	Offset
TRATSR0	104h
TRATSR1	114h
TRATSR2	124h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHA Automotive Technology Co., Ltd.
ca462e3-4811-8a8-b251-390d3ab624c1

Function

This register returns the average measured temperature at the site, which must be a part of the list of enabled monitored sites as defined by TMSR[SITE].

Diagram



Fields

Field	Function
31 V	Valid measured temperature 0b - Not valid; temperature is out of sensor range or the first measurement is still pending 1b - Valid
30-9 —	Reserved
8-0 TEMP	Average temperature reading in Kelvin at the site Valid when V=1.

59.3.27 Central Module Configuration (TCMCFG)

Offset

Register	Offset
TCMCFG	F00h

Function

This register allows for control of the SAR algorithm and clock settings during a measurement. For proper operation, updates to the register should only occur when monitoring is disabled.

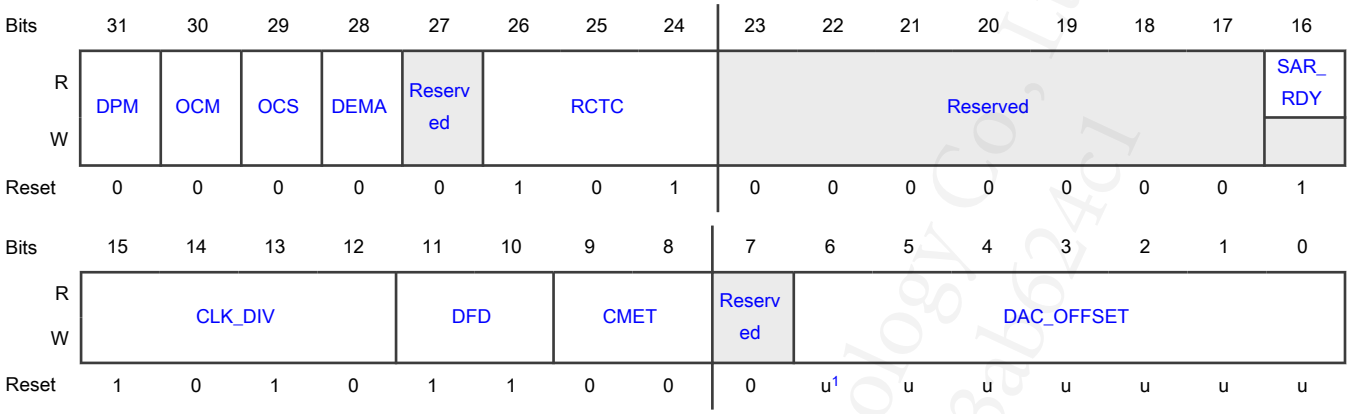
Writing to this register triggers an automatic clock speed change based on the new CLK_DIV and DFD settings. You must poll SAR_RDY to determine when it is safe to continue the monitoring operation.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-390d3ab624c1

NOTE

The values for OCS, OCM, DEMA, DFDRCTC, and CLKDIV are chip-specific. See the "Chip-specific TMU section" for the required settings.

Diagram



1. The DAC_OFFSET value comes from fuses. Also, the values depend on the DAC calibration.

Fields

Field	Function
31 DPM	Dynamic power management 0b - TMU is disabled from entering low-power mode. Circuit is permanently enabled. 1b - Reserved.
30 OCM	Offset cancellation mode Reverses the current path through the op amps. **NOTE** This field should be set to 1 for the normal TMU monitor operation. For details on initialization, see Initialization information.
29 OCS	Offset cancellation manual setting Applicable when OCM=0 for manual control of current path through the op amps. 0b - Current path 0 1b - Reserved
28 DEMA	Dynamic element match averaging mode The voltage bias to the sensor opamp is a N:M ratio of equally sized transistors. Because of process variations, we cannot guarantee the transistor ratio. Through dynamic element matching, M out of (M+N) transistors can be selected to provide the best current mirror ratio match for the sensor diode. **NOTE** This field should be set to 1 for the normal TMU monitor operation. For details on initialization, see Initialization information.

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 —	Reserved
26-24 RCTC	RC time constant setting Before SAR runs, the analog signals to the control site, offset swap, and DEM must settle. This setting allows for a configurable delay based on the SAR clock as defined by CLK_DIV. 000b - Reserved 001b - Reserved 010b - Reserved 011b - Reserved 100b - Reserved 101b - 256 SAR clocks 110b - Reserved 111b - Reserved
23-17 —	Reserved
16 SAR_RDY	SAR ready Indicates that the SAR module is ready to receive user commands. During normal functional mode, this field must be set for calibration or monitoring to work properly. During a clock switch, which occurs as a result of writing to this register, the SAR module may be temporarily unavailable. 0b - SAR not ready to receive command 1b - SAR ready to receive command
15-12 CLK_DIV	Clock divider Clock divider for SAR clock based on platform clock. The digital filter clock runs at a clock frequency of [DFD x SAR]. For example, if the main platform clock is 800 MHz, and the divider setting is 32, SAR runs at 25 MHz. If eight samples are taken by the digital filter, each sample is taken at 200 MHz. 0000b - Reserved 0001b - Reserved 0010b - Reserved 0011b - Reserved 0100b - Reserved 0101b - Reserved 0110b - Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0111b - Reserved 1000b - Reserved 1001b - Reserved 1010b - Divide by 16K 1011b - Reserved 1100b - Reserved 1101b - Reserved 1110b - Reserved 1111b - Reserved
11-10 DFD	Digital filter depth TMU takes multiple DAC code bit samples during successive approximation and uses the average. 00b - Reserved 01b - Reserved 10b - Reserved 11b - Eight samples
9-8 CMET	Central module enable time Indicates how long to wait after enabling the centralized TMU circuit before it is operational. This information is used when TMR[CMD] changes from 1 to 0, and during dynamic power management when DPM=1. 00b - 5 ms @ 800 MHz platform 01b - Reserved 10b - Reserved 11b - Reserved
7 —	Reserved
6-0 DAC_OFFSET	DAC offset Binary two's complement value between -64 and +63 Modifies the SAR DAC result from a measurement accordingly. The initial value of this field is sampled during POR from fuses and is part-specific, and should not be changed by you.

59.3.28 Temperature Range Control a (TTRCR0 - TTRCR15)

Offset

For a = 0 to 15:

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e-4821-48a8-b251-390d3ab624c1

Register	Offset
TTRCRa	F10h + (a × 4h)

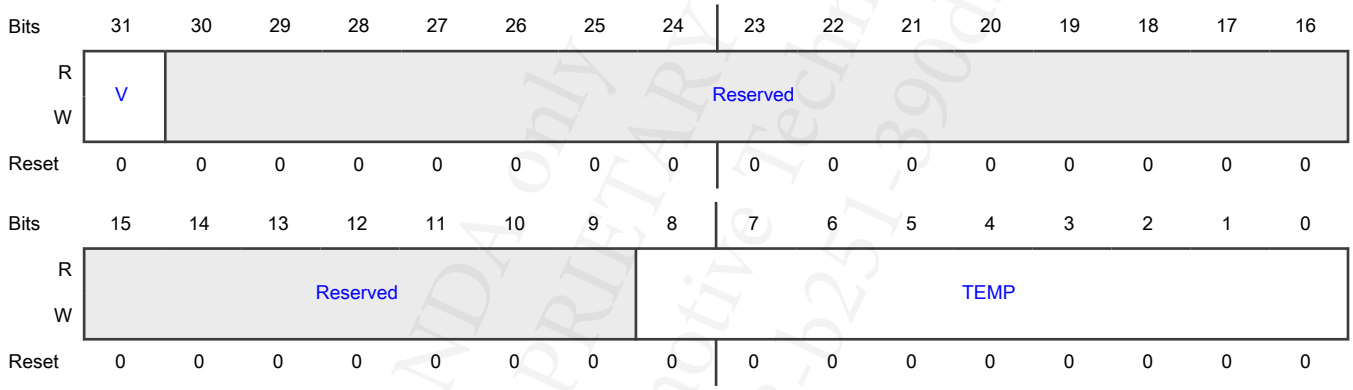
Function

These registers allow you to define the calibration points used to cover the temperature sensor range. TTRCR0 defines the start of the temperature range which, along with TTRCR1, must always be valid to account for at least two calibration points. We provide selected calibration points to account for any non-linear behavior in the sensor as a function of temperature. A perfectly linear sensor reading requires only two calibration points to find the temperature based on linear interpolation. You must enable the calibration points sequentially, starting with TTRCR0 through TTRCR15.

NOTE

The circuit is designed and calibrated for measuring temperatures in the range of 233 K to 398 K (-40 °C to +125 °C). Without prior circuit knowledge, it is not recommended to go beyond this range when setting up the calibration points. We cannot guarantee the margin of error outside these limits.

Diagram



Fields

Field	Function
31 V	Calibration point is valid 0b - Not valid 1b - Valid
30-9 —	Reserved
8-0 TEMP	Temperature in Kelvin for the calibration point In the case of TTRCR0, this is the starting temperature for the range.

59.4 Functional description

The following sections describe the functionality of the TMU in detail.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
cc4625-4821-48a8-b251-390d3ab624c1

59.4.1 Monitoring

Monitoring is the process of reading on-chip temperature sensor sites at regular intervals and taking appropriate actions, such as alarming the user when the temperature exceeds a programmed temperature threshold. The monitoring process only applies to those sites that are enabled.

During this process, TMU periodically measures all enabled temperature sensor sites for temperature, starting with site 0 and ending with site 2. In Monitor Temperature Measurement Interval (TMTMIR), you can set the interval at which TMU reads the sensors, and the setting should reflect the maximum interval time required to accurately capture temperature changes. If the measurement interval has not expired after the last active site is read, the sensor logic may enter the low-power mode. If the interval expires before the module reads the last active site, the module sets TSR[MIE] to 1 and reads the next active site immediately. If the interval has been exceeded, you may opt to reduce the number of sites monitored by TMU, or increase the interval, if possible. This is especially critical for capturing temperature rate change events accurately.

TMU records the current and average temperature for each site. It calculates the average temperature based on the low-pass filter function in the TMU mode register. If the module triggers any of the enabled threshold or rate event registers, it sets the corresponding interrupt detect field in Interrupt Detect (TIDR). You enable interrupts through Interrupt Enable (TIER).

The procedure to enable the monitoring mode is as follows:

1. Clear the interrupt detect register, Interrupt Detect (TIDR).
2. Clear the interrupt capture registers: Interrupt Immediate Site Capture (TIISCR), Interrupt Average Site Capture (TIAS CR), and Interrupt Critical Site Capture (TICSCR).
3. Enable interrupt handling by writing 1 to the appropriate fields in the interrupt enable register, Interrupt Enable (TIER).
4. Write the appropriate values to the following temperature threshold registers:
   - Monitor High Temperature Immediate Threshold (TMHTITR)
   - Monitor Low Temperature Immediate Threshold (TMLTITR)
   - Monitor High Temperature Average Threshold (TMHTATR)
   - Monitor Low Temperature Average Threshold (TMLTATR)
   - Monitor High Temperature Average Critical Threshold (TMHTACTR)
   - Monitor Low Temperature Average Critical Threshold (TMLTACTR)
5. Write the appropriate values to thsee temperature rate change registers: Monitor Rising Temperature Rate Critical Threshold (TMRTRCTR) and Monitor Falling Temperature Rate Critical Threshold (TMFTRCTR).
6. Write the appropriate value to the monitoring interval register, Monitor Temperature Measurement Interval (TMTMIR).
7. Enable the monitor mode by setting TMR[MODE] to b10. You control the monitored sites by writing an appropriate value to TMSR[SITE]. You need to enable at least one active site, and set the other mode control fields as needed.
8. If the monitoring interval is too short, as indicated by TSR[MIE], you may need to increase the interval or reduce the number of sites.

59.4.2 Reporting

The TMU can directly report the current and average temperature for a particular temperature sensor site during the monitoring mode by reading one of the site report registers. The report uses the last measurement from the monitoring process, and requires a site to be actively monitored to obtain an accurate reading. Reading a site, which last measured an invalid temperature outside the sensor range of 233 K to 398 K, will have the valid bit cleared.

If you disable monitoring, the last temperature measurement for the site(s) that the TMU previously monitored is retained by the module, and you can still read the report registers, with the knowledge that the temperature(s) reported is (are) no longer accurate. This method can be used to capture the temperatures at multiple sites in time, but does not allow for continuous monitoring.

The temperature capture registers (Monitor High Temperature Capture (TMHTCR) and Monitor Low Temperature Capture (TMLT CR)) record the history of the highest and lowest temperatures that they measure within the valid sensor range. The temperature rate change capture registers (Monitor Rising Temperature Rate Capture (TMRTRCR) and Monitor Falling Temperature

Rate Capture (TMFTRCR)) record how quickly the temperature rises or falls, and you can use this information to determine programmable values for the rate change threshold registers in real applications.

You can read out-of-range indications from Status (TSR). You write 1 to clear these indications during active monitoring.

59.4.3 Using temperature thresholds

The main purpose of the TMU is to autonomously measure temperature at select sites and report to the core when a potentially dangerous operating temperature is about to be reached. This may be to protect a circuit from overheating and causing permanent damage or affecting reliability.

The TMU has a two-level (normal/critical) threshold and interrupt scheme. This allows the software to take preventive actions at a first temperature threshold and additional drastic measures at a second critical threshold. For high temperatures, these measures may include:

- Downgrading the performance of a function or core to reduce power consumption (preventive)
- Reducing the voltage or frequency to a function or core

When TMU detects a high temperature (through interrupt), the software could enable a low or normal threshold to detect when to restore functions to normal. Similarly, preventive actions taken for low temperatures could enable a high or normal temperature to restore functions.

For proper operation, you must set the temperatures in the high threshold registers to be higher than those you set in the low threshold registers.

The rising and falling temperature rate threshold registers (Monitor Rising Temperature Rate Critical Threshold (TMRTRCTR) and Monitor Falling Temperature Rate Critical Threshold (TMFTRCTR)) trigger critical interrupts. A quickly rising temperature because of heatsink separation or cooling fan issues may occur sooner than a critically high temperature. For example, immediate preventive measures can be taken to protect the chip. You could determine a nominal value for the rising threshold value by using the Monitor Rising Temperature Rate Capture (TMRTRCR) or from lab simulations. A quickly falling temperature is less common but you could use this to warn the chip about stress caused by drastic changes in temperature that may lead to cracking.

59.5 Initialization information

TMU calculates temperature by reading one of the on-chip temperature sensor sites. The module calculates the temperature using a table of calibration points that you must program using the procedure outlined below.

You can manually program the sensor translation table by supplying the sensor readings for all temperature points as defined by the TMU temperature range control registers. Failure to initialize all the temperature/sensor configuration table entries correctly may lead to an undefined behavior. The TMU temperature range control registers, Temperature Range Control a (TTRCR0 - TTRCR15), define the starting temperature, the ending temperature, and a number of intervening calibration points. The number of configuration points used depends on the linearity of the sensor. If a sensor reading falls between two configuration points, TMU reports the temperature based on linear interpolation between the nearest upper/lower two points in the table.

Following is the procedure to program the TMU calibration table:

NOTE

For the last four calibration points (12-15), there are per-part trims that you need to apply. You must read these OCOTP fields (in the appropriate shadow register): CFG_DAC_TRIM_0, CFG_DAC_TRIM_1, CFG_DAC_TRIM_2, and CFG_DAC_TRIM_3.

1. Disable monitoring mode, TMR[MODE]=0 (default).
2. Perform a Read-Modified-Write on register Central Module Configuration (TCMCFG) to write 1 to TCMCFG[OCM] and TCMCFG[DEMA]. Do not modify the other fields in this register.
3. Write the temperature configuration register with TTCFGR[CAL_PT]=*n* (where *n* is the calibration point from the table).
4. Write to the TMU configuration register, Sensor Configuration (TSCFGR), with the sensor value for the first calibration point *n*.
5. Write to the TMU temperature range control registers (TTRCR*n*) associated with calibration point *n*.

You need to repeat steps 3—5 for all calibration points from the provided calibration table. You must write 1 to the DEMA and OCM fields of Central Module Configuration (TCMCFG) for the proper operation of TMU.

The values for TTCFGR and TTRCR are chip-specific. See the "Chip-specific TMU section" for details.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 60
Security

60.1 Introduction

The Hardware Security Engine (HSE_H) is a subsystem that implements the security functions for the device. It provides cryptographic services to host CPUs and the network accelerators, targeting current security specifications. For example, Secure Hardware Extension (SHE), Hardware Security Module (HSM), and E-safety Vehicle Intrusion Protected Application (EVITA) Full. The HSE_H subsystem is responsible for establishing the root of trust on the device during the boot process and includes the following features:

- Secure boot of customer code using asymmetric or symmetric keys
- Highly featured symmetric and asymmetric accelerators
- Support for the following cryptographic functions in hardware:
  - AES (up to 256)
  - SHA-1 and SHA-2
  - Wide support of Elliptic curves (ECC)
  - RSA (up to 4096).
- Arm Cortex-M7 CPU
- True Random Number Generator (TRNG)
- Pseudo Random Number Generator (PRNG)
- Side channel physical protection
- Glitch attack countermeasures
- Firmware Over-the-Air (FOTA) support.

Following is a simplified block diagram of the HSE_H:

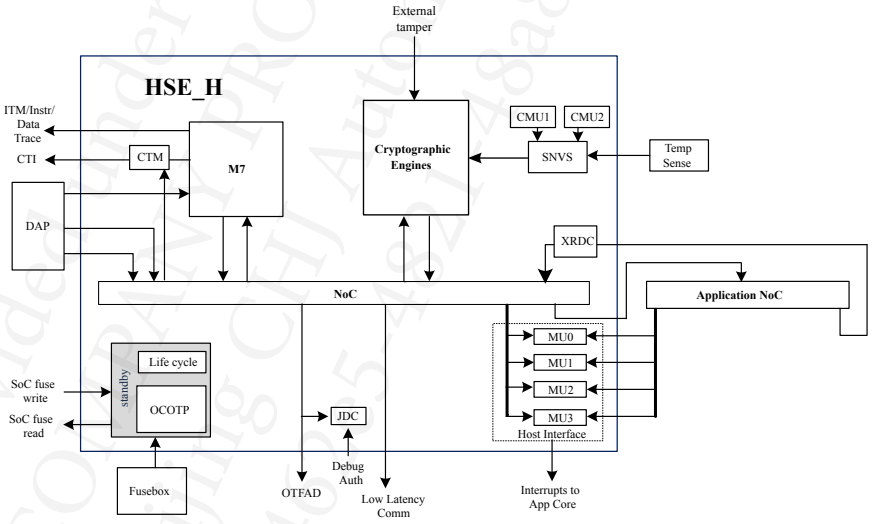


Figure 544. HSE_H simplified block diagram

NOTE

1. Factory-supplied firmware controls HSE_H function. See the documents in the following table for details.[11]
2. No confidentiality is supported for customer application code in devices subject to export controls.

Document Number	Document Title	Description
HSEFWRM	HSE Firmware Reference Manual	Contains architecture details of security firmware
HSESIRM	HSE Service Interface Reference Manual	Security firmware API reference

60.1.1 Detection and response to parametric intrusion attempts

Software and hardware designs incorporate known industry standard measures to prevent leakage of secrets and exposure to other exploitable vulnerabilities. The Security subsystem is designed to be resistant against certain fault and glitch attacks.

60.1.2 Secure software updates

To avoid costly and time consuming service bay updates, over the air (OTA) update is supported for situations where the vehicle MCU software must be refreshed. Updates may be needed to fix known problems or to provide additional features to the vehicle. The Security subsystem supports secure OTA updates for software that is executed on the device, such as HSE code, application code, operating systems and drivers, but not the Boot Loader in ROM.

60.1.3 Secure vehicle network and messaging

Communication between ECUs within a vehicle is necessary for safe and proper operation. Messages between ECUs must be guaranteed authentic and free of error. Attacks on the network can be through a direct connection to the network wires or indirectly through an alternative communication port. The Security subsystem supports both secret key (symmetric) and public key (asymmetric) authentication protocols. Because of the large number of network nodes, data rates and message sizes, a symmetric key protocol is the prevalent authentication choice.

60.1.4 Secure boot

The image stored in external memory may be encrypted and require authentication. The choices are selected by configuration values stored in IVT present in external flash. Authentication with CMAC, HMAC, RSA-2048, ECC-256, and ECC-384 is supported. For details, refer to the Boot chapter.

60.2 Security subsystem registers

60.2.1 Security subsystem register descriptions

The security subsystem registers are detailed in this section.

NOTE

1. Only word accesses are supported for these registers.
2. Any access made to address offsets Ch-1Bh will not generate a transfer error.

60.2.1.1 SEC memory map

Security_CC base address: 4007_C900h

[11] The distribution of each document listed is restricted. Contact your NXP Marketing or Sales representative for further information.

Offset	Register	Width (In bits)	Access	Reset value
8h	External Debugger Status (EXT_DBGSTAT)	32	RO	See description
1Ch	HSE GPR 0 (HSE_GPR0)	32	RW	0000_0000h
20h	HSE GPR n (HSE_GPR1)	32	RW	0000_0000h
24h	HSE GPR n (HSE_GPR2)	32	RW	0000_0000h
28h - 48h	HSE GPR n (HSE_GPR3 - HSE_GPR11)	32	RW	0000_0000h

60.2.1.2 External Debugger Status (EXT_DBGSTAT)

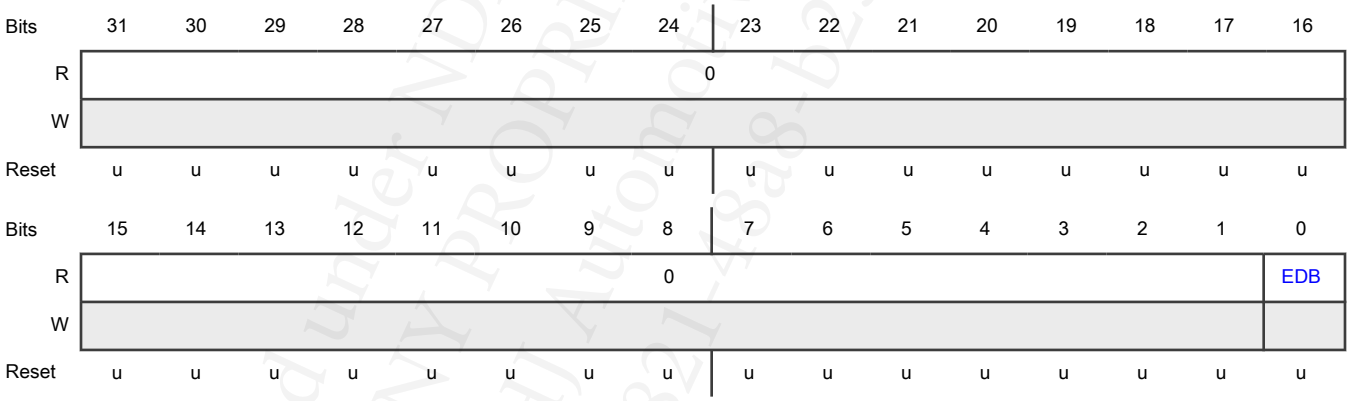
Offset

Register	Offset
EXT_DBGSTAT	8h

Function

Indicates external debugger connectivity

Diagram



Fields

Field	Function
31-1 —	Reserved
0 EDB	External debugger indication Indicates whether an external debugger is connected. After the debugger is connected, this field is set to '0' during the next reset (POR) cycle.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - External debugger is not connected. 1b - External debugger (JTAG) is connected.

60.2.1.3 HSE GPR 0 (HSE_GPR0)

Offset

Register	Offset
HSE_GPR0	1Ch

Function

This is a general purpose register that is reset on every reset.

NOTE

Access to this register is Read/Write for the HSE_H software and Read Only for the application software.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATA1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA1									Reserved	DATA0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-7 DATA1	General purpose data
6 —	Reserved
5-0 DATA0	General purpose data

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e54821484a8-025-399d3ab624c1

60.2.1.4 HSE GPR n (HSE_GPR1)

Offset

Register	Offset
HSE_GPR1	20h

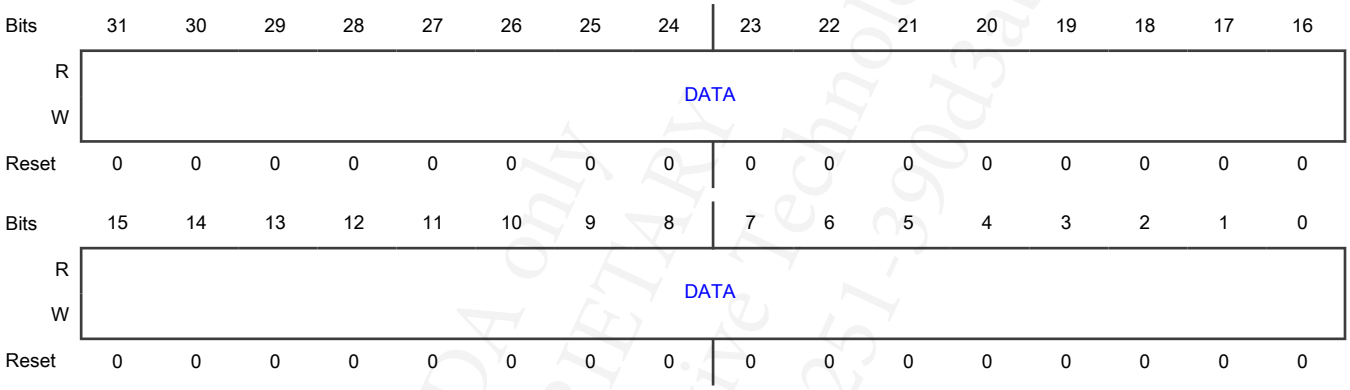
Function

This is a general purpose register that is reset on every reset.

NOTE

Access to this register is Read/Write for the HSE_H software and Read Only for the application software.

Diagram



Fields

Field	Function
31-0 DATA	General purpose data

60.2.1.5 HSE GPR n (HSE_GPR2)

Offset

Register	Offset
HSE_GPR2	24h

Function

This is a general purpose register that is reset on every reset.

NOTE

Access to this register is Read/Write for the HSE_H software and Read Only for the application software.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4821-48a8-b251-590d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATA	General purpose data

60.2.1.6 HSE GPR n (HSE_GPR3 - HSE_GPR11)

Offset

For a = 3 to 11:

Register	Offset
HSE_GPRa	1Ch + (a × 4h)

Function

This is a general purpose register that is reset on destructive reset and POR.

NOTE

Access to this register is Read/Write for the HSE_H software and Read Only for the application software.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
8c462e54-8214-48a8-b251-390d3ab624c1

Fields

Field	Function
31-0 DATA	General purpose data

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 61
Messaging Unit (MU)

61.1 Chip-specific MU information

This device includes four MU module instances:

- MU0
- MU1
- MU2
- MU3

Each instance has two sets of registers: MUA and MUB.

Table 535. Register sets of each MU instance

Register set	Mapped to	Accessible only by	Use
MUA	HSE module's internal address space	HSE's Cortex-M7 processor core	Information included for reference only
MUB	Host address space	Application processor core	Functions as the host interface

61.2 Overview

The Messaging Unit module enables two processors within the SoC to communicate and coordinate by passing messages (e.g. data, status and control) through the MU interface. The MU also provides the ability for one processor to signal the other processor using interrupts.

Because the MU manages the messaging between processors potentially using different clocks, the MU must synchronize the accesses from one side to the other. The MU accomplishes synchronization using two sets of matching registers (Processor A-facing, Processor B-facing).

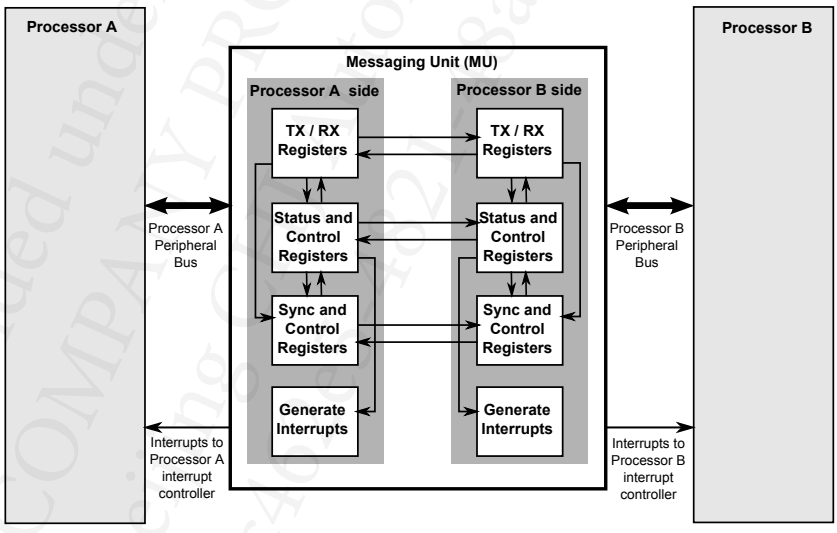


Figure 545. MU Block Diagram

Provided under NDA only
COMPANY PROPRIETARY
Beijing Chuangxin Technology Co., Ltd.
c8c4e4f42f8a8-2251-390a3ab624c1

61.2.1 Features

The MU includes the following features:

- Memory-Mapped Registers
  - The MU is connected as a peripheral under the Peripheral bus on Processor A-side and Processor B-side.
- Synchronized Message Transfers between Cores
  - For sending data or messages from one MU-side to the other MU-side, the MUA provides 16 transmit registers and 16 receive registers, the MUB provides 16 transmit registers and 16 receive registers.
  - The transfer of data messages between cores uses transmit empty and receive full flags provided on both sides of the MU.
  - The update of these transmit and receive flags is accomplished using a synchronization mechanism. There is inherent latency between updating the flag on one side and reflecting its status on other side.
  - The MU has a 32 bit Flag Data Register, which can be used to send flag data between two MU sides.
- Interprocessor Interrupts
  - The MU has 64 interrupt sources on each side (Processor A-side, Processor B-side) that are used for signaling the other processor. The interrupts can be used for notification of RX/TX events and general-purpose signaling between the processors. There are 32 general-purpose interrupt requests available and 32 RX/TX interrupt sources.
  - Processor A can initiate a Non-Maskable Interrupt to Processor B
  - Processor B can initiate a Non-Maskable Interrupt to Processor A
- MU Reset
  - Each Processor can issue a reset to the entire MU, using a control bit (MUR) in each Processor's Control Register (CR).
  - The MUR bit is a self-clearing bit.

61.3 Register Definition

The MU provides transmit and receive data registers (xTR0-n, xRR0-n) for the communication between Processor A and Processor B. Some control and status registers (xCR, xSR) to the Processor A and Processor B sides for control operations (such as interrupts and reset), and for status checking of the other MU-side. Figure 546 shows the MU registers schematic.

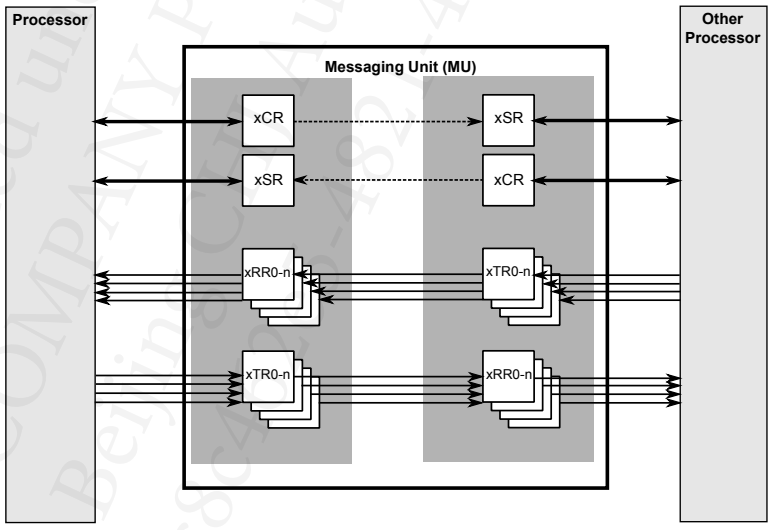


Figure 546. MU Registers

The detailed MU Register Definition can be found below.

61.3.1 MUA register descriptions

This section contains the detailed register descriptions for the MUA registers.

NOTE

A read/write access to any illegal location and a write to a read-only register on the Processor A-side or Processor B-side of the MU will generate a module transfer error acknowledge to the Processor A or Processor B. The only exception to this is at address offsets 0x1C and 0x14 - no transfer error will occur when accessing these locations.

61.3.1.1 MUA memory map

MU0.MUA base address: 2325_8000h

MU1.MUA base address: 2325_9000h

MU2.MUA base address: 2325_A000h

MU3.MUA base address: 2325_B000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VER)	32	RO	0300_000Fh
4h	Parameter Register (PAR)	32	RO	2020_1010h
8h	Control Register (CR)	32	RW	0000_0000h
Ch	Status Register (SR)	32	W1C	See description
10h	Core Control Register 0 (CCR0)	32	RW	0000_0000h
18h	Core Sticky Status Register 0 (CSSR0)	32	W1C	0000_0000h
100h	Flag Control Register (FCR)	32	RW	0000_0000h
104h	Flag Status Register (FSR)	32	RO	0000_0000h
110h	General Interrupt Enable Register (GIER)	32	RW	0000_0000h
114h	General Control Register (GCR)	32	RW	0000_0000h
118h	General Status Register (GSR)	32	W1C	0000_0000h
120h	Transmit Control Register (TCR)	32	RW	0000_0000h
124h	Transmit Status Register (TSR)	32	RO	0000_FFFFh
128h	Receive Control Register (RCR)	32	RW	0000_0000h
12Ch	Receive Status Register (RSR)	32	RO	0000_0000h
200h - 23Ch	Transmit Register (TR0 - TR15)	32	RW	0000_0000h
280h - 2BCh	Receive Register (RR0 - RR15)	32	RO	0000_0000h

NXP CONFIDENTIAL
Beijing ESWIN Computing Technology Co., Ltd.
c8462e5-4821-48a8-b251-3903ab624c1

61.3.1.2 Version ID Register (VER)

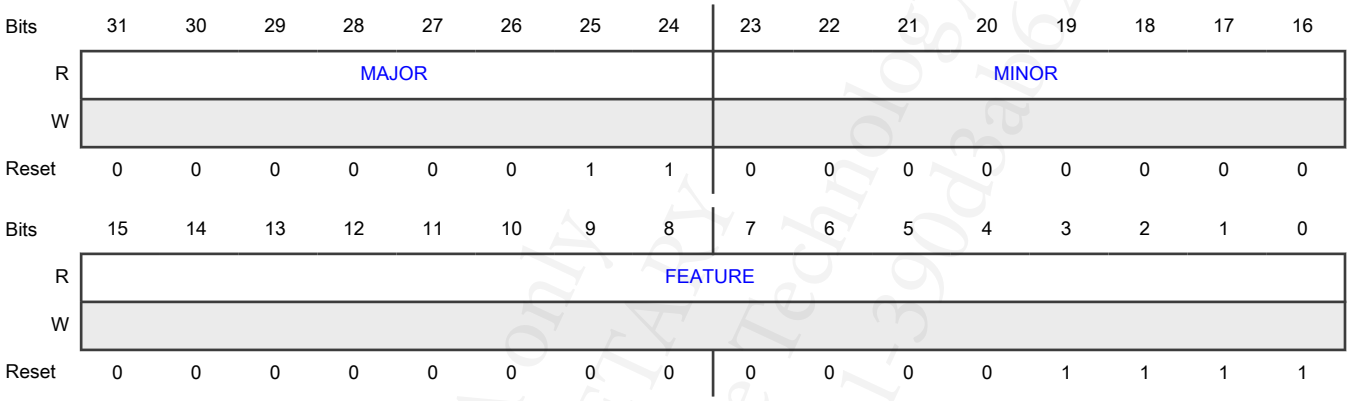
Offset

Register	Offset
VER	0h

Function

Use Version ID register to determine the version ID and feature set number of MUA.

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number
23-16 MINOR	Minor Version Number
15-0 FEATURE	Feature Set Number Each bit of the Feature Set Number bitfield specifies a different feature: • If bit0 = 1: Standard features are implemented • If bit1 = 1: RAIP/RAIE register bits are implemented • If bit2 = 1: Core Control and Status Registers are implemented in both MUA and MUB • If bit3 = 1: Expand TRn/RRn registers number

Provided under NDA only
COMPANY PROPRIETARY
Beijing CEC Automotive Technology Co., Ltd.
c8c46e21-48a8-b251-390d3a624c1

61.3.1.3 Parameter Register (PAR)

Offset

Register	Offset
PAR	4h

Function

The PAR register defines the number of flags, transmit registers, receive registers, and the number of general interrupt requests available for the MU.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FLAG_WIDTH								GIR_NUM							
W																
Reset	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RR_NUM								TR_NUM							
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-24 FLAG_WIDTH	Flag Width This bitfield specifies the number of flag bits in the Flag Control Register (FCR) and Flag Status Register (FSR).
23-16 GIR_NUM	General Interrupt Request Number This bitfield specifies the number of general interrrupt requests available.
15-8 RR_NUM	RR Number This bitfield specifies the number of receive registers.
7-0 TR_NUM	Transmit Register Number This bitfield specifies the number of transmit registers.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46215-4821-4ca8-b251-39003a062401

61.3.1.4 Control Register (CR)

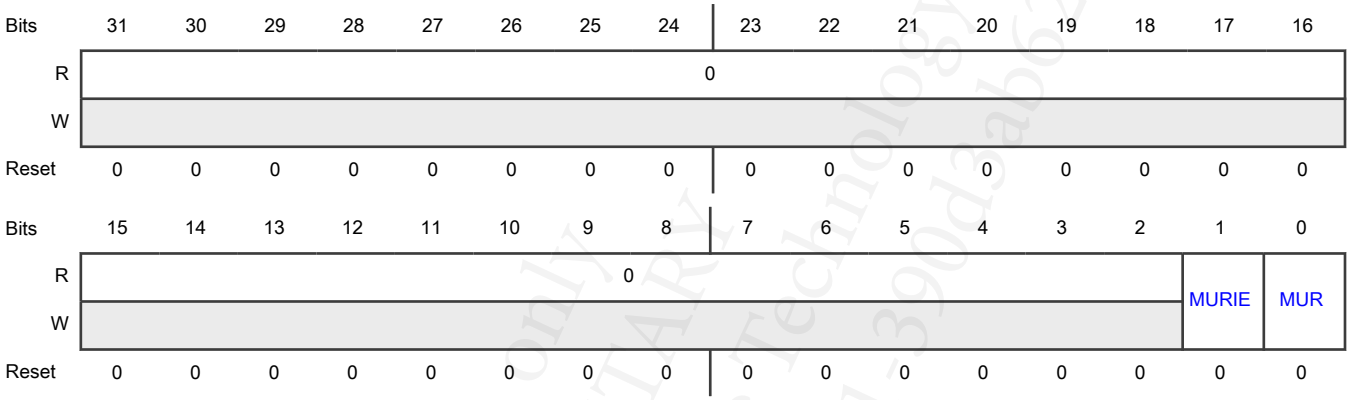
Offset

Register	Offset
CR	8h

Function

The CR register controls the MU reset and MU reset interrupt enable.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 MURIE	MUA Reset Interrupt Enable • If MURIE bit is set to "1", then MU reset interrupt request is issued to the Processor A when the MUA_SR[MURIP] bit is set to "1". • If MURIE is cleared, then the value of the MURIP bit is ignored and no MU reset interrupt request will be issued. • Only system reset can reset MURIE, MU reset(MUR) cannot reset this bit. 0b - Disables Processor A-side MU Reset Interrupt request due to MU reset issued by MUB. 1b - Enables Processor A-side MU Reset Interrupt request due to MU reset issued by MUB.
0 MUR	MU Reset • Setting MUR bit to "1" resets both the MUA-side and MUB-side, forcing all control and status registers to return to their default values (except in MUA/B_CCR0 registers, MURIE in MUA/B_CR registers, MURIP bit and MURS bit in MUA/B_SR registers), and all internal states to be cleared.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• Before setting the MUR bit to "1", it is advisable to interrupt the Processor B, because setting the MUR bit may affect the ongoing Processor B program. • After setting the MUR bit, monitor the value of the MUA_SR[MURS] bit to know when the reset sequence on the Processor B-side has ended. • MUR bit is always read as "0". • MUR bit is cleared during the MU reset sequence. 0b - Self clearing bit. 1b - Asserts the MU reset.

61.3.1.5 Status Register (SR)

Offset

Register	Offset
SR	Ch

Function

The SR register shows the status of MU resets and the status of pending events/requests described below.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	RFP	TEP	GIRP	FUP	EP	MURIP	MURS
W															W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	u

Fields

Field	Function
31-8 —	Reserved
7	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4824-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
6 RFP	MUA Receive Full Pending Flag • RFP bit is set to "1" when any TRn register is written by MUB to send data to MUA. • After the RFP bit is set to "1", checking the RSR[RFn] bit to decide the data in which RRn register is ready to be read by the MUA. • RFP bit is cleared when all MUA RRn registers are read, or when the MU is reset. 0b - No TRn register is written by MUB. 1b - Any TRn register is written by MUB.
5 TEP	MUA Transmit Empty Pending • TEP bit is set to "1" when TCR[TIEn] bit is set and the corresponding RRn register is read by MUB. • After the TEP bit is set to "1", checking the TSR[TEn] bit to decide the data in which TRn register is ready to be written by the MUA. • TEP bit is cleared when all MUA TRn register are written. • TEP bit is set to "0" when the MU resets. 0b - RRn register is not read by MUB. 1b - Any RRn register is read by MUB.
4 GIRP	MUA General Interrupt Pending • GIRP bit is set to "1" when any general interrupt request is sent from the MUB side to the MUA side. Reading the GSR[GIPn] bit can know which general interrupt request is received. • GIRP bit is cleared when all GIPn bits in MUA_GSR register are cleared. • GIRP bit is cleared when the MU resets. 0b - No general interrupt request is sent from MUB. 1b - Any general interrupt request is sent from MUB.
3 FUP	MUA Flags Update Pending • FUP bit is set to "1" when the MUA side sends a Flags Update request to the MUB side. • A Flags Update request is generated when there is a change to the Fn[31:0] bits of the MUA_FCR register. No flag update changes are allowed while the FUP bit is set to "1". Any write to the Fn[31:0] bits of the MUA_FCR register, while the FUP bit is set to "1", will not generate a Flags Update event, and the Fn[31:0] bits will stay unchanged. • Writing the FCR register doesn't set FUP bit immediately if there is any event pending(SR[EP] is "1"). • FUP bit is cleared when this Flags Update request is internally acknowledged (the flag is updated) from the MUB side, or during MU reset. 0b - No pending update flags(initiated by MUA) are in process 1b - Pending update flags(initiated by MUA) are in process

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 EP	MUA Side Event Pending • An "event" is any hardware message that is reflected in the status register on the MUB side (for example, "transmit register 0 written"). During normal operations, the update mechanism for the EP bit works automatically. • EP bit is set to "1" when the MUA side sends an event update request to the MUB side. • EP bit is cleared by hardware automatically when the event update acknowledge is received. • To ensure events have been posted to MUB, verify that the EP bit is cleared. If EP bit is set to "1", wait and continue to poll the EP bit. • The EP bit is cleared when the MU resets. 0b - The MUA side event is not pending. 1b - The MUA side event is pending.
1 MURIP	MU Reset Interrupt Pending • MURIP bit signals Processor A that Processor B initiated a MU reset by setting the MUB_CR[MUR] bit. • MURIP bit is set to "1" after Processor B initiated a MU reset. If the interrupt is enabled by the MURIE bit, the Processor A MU reset interrupt request is issued when the Processor B set MUR bit in MUB_CR. • Writing "1" to clear the MURIP bit, which also clears MU reset interrupt request. • Only system reset can reset MURIP. MU reset cannot reset this bit. 0b - Processor B did not issue MU reset. 1b - Processor B issued MU reset.
0 MURS	MUA and MUB Reset State • The MURS bit is set to "1" during any system reset or any MU reset from MUA-side or MUB-side. • The MURS bit is cleared when the reset sequence on both MUA and MUB side ends. After issuing any of the reset events mentioned previously, **verify that the MURS bit is cleared before starting any access.** 0b - MUA and MUB are out of reset. 1b - MUA or MUB is in reset state.

61.3.1.6 Core Control Register 0 (CCR0)

Offset

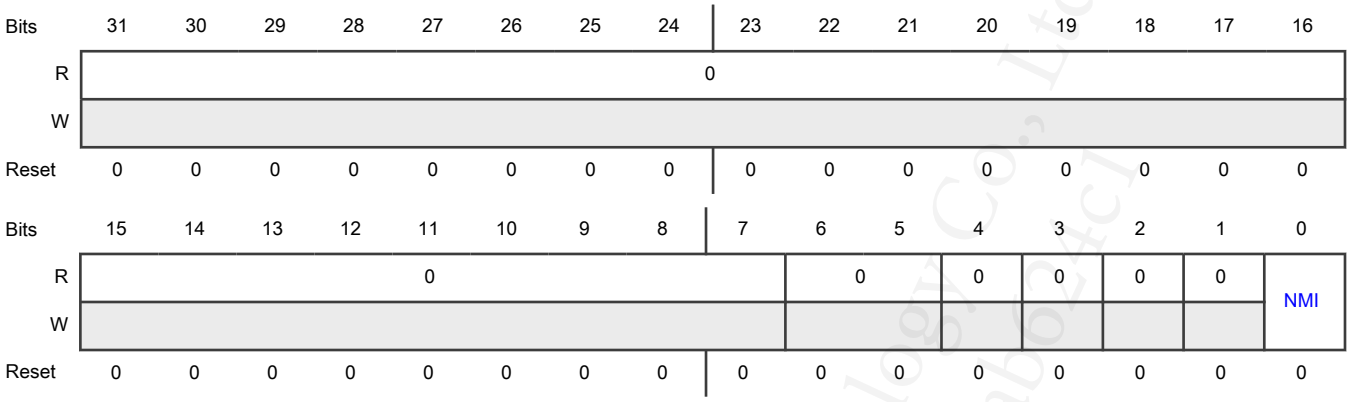
Register	Offset
CCR0	10h

provided under NDA only
COMPANY PROPRIETARY
Beijing CAT Automotive Technology Co., Ltd.
c8c462e5-4821-4a8-b21-90d3ab62

Function

The CCR0 enables the MUA to control the Processor on the MUB-side.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 NMI	MUB Non-maskable Interrupt Request • When NMI bit is set to "1", it initiates a Non-Maskable Interrupt to the Processor B. • NMI bit is cleared after the MUB_CSSR0[NMIC] bit is written to "1". After the NMI bit is cleared, the MUA can initiate another non-maskable interrupt to the MUB. • The NMI bit is cleared when the MU resets. 0b - Non-maskable interrupt is not issued to the Processor B by the Processor A. 1b - Non-maskable interrupt is issued to the Processor B by the Processor A.

Provided under NDA only
COMPANY PROPRIETARY
Renesas RCHJ Automotive Technology Co., Ltd.
c5100c_4821-48a8-b251-390d3ab624c1

61.3.1.7 Core Sticky Status Register 0 (CSSR0)

Offset

Register	Offset
CSSR0	18h

Function

The CSSR0 register shows the status of interrupts pending (W1C).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	0	0	0	0	0	0	NMIC
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390da3ab624c1

Table continued from the previous page...

Field	Function
—	
1 —	Reserved
0 NMIC	Processor A Non-Maskable-Interrupt Clear • NMIC bit is used by the MUA-side Non-Maskable Interrupt service routine to clear the non-maskable interrupt from MUB-side. • Writing NMIC bit as "1" signals the MUB to clear the MUB_CCR0[NMI] bit, thus de-asserting the interrupt and enabling the NMI bit to receive another interrupt. • The NMIC bit is always read as "0", therefore NMIC bit cannot be polled. The NMIC bit can only be used as part of the interrupt service routine, in which the NMIC bit should only be written as "1" once. • The NMIC bit is cleared when the MU resets. 0b - Default 1b - Writing "1" clears the MUB_CCR0[NMI] bit.

61.3.1.8 Flag Control Register (FCR)

Offset

Register	Offset
FCR	100h

Function

The FCR register contains read-write flags that reflect the MUB_FSR[Fn] bits.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	F31	F30	F29	F28	F27	F26	F25	F24	F23	F22	F21	F20	F19	F18	F17	F16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESI Automotive Technology Co., Ltd.
c84e2e5d-21-28a8-b251-390d3a6041

Fields

Field	Function
31-0 Fn	MUA to MUB Flag n • n = 0 to 31 • Fn bit is a read-write flag that reflects the MUB_FSR[Fn] bits. • Fn bits are cleared when the MU resets. • Write 0 to an Fn bit - Clears the Fn bit in the MUB_FSR register. • Write 1 to an Fn bit - Sets the Fn bit in the MUB_FSR register. 0b - Clears the Fn bit in the FSR register. 1b - Sets the Fn bit in the FSR register.

61.3.1.9 Flag Status Register (FSR)

Offset

Register	Offset
FSR	104h

Function

The FSR register contains flags that reflect the values written to the Fn bit of the MUB_FCR register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	F31	F30	F29	F28	F27	F26	F25	F24	F23	F22	F21	F20	F19	F18	F17	F16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 Fn	MUB to MUA Side Flag n • n = 0 to 31

Table continues on the next page...

Field	Function
	• Fn bit is the MUA side flag that reflects the values written to the Fn bit in the MUB_FCR register. • Every time that the Fn bit in the MUB_FCR register is written, the Fn bit in the MUB FCR register write event updates the Fn bit in the MUA_FSR register after the event update latency. • If a 0 is read from an Fn bit - Fn bit in the MUB_FCR is 0. • If a 1 is read from an Fn bit - Fn bit in the MUB_FCR is 1. 0b - Fn bit in the MUB FCR register is written 0. 1b - Fn bit in the MUB FCR register is written 1.

61.3.1.10 General Interrupt Enable Register (GIER)

Offset

Register	Offset
GIER	110h

Function

The GIER register contains the MUA general purpose interrupt enables.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GIE31	GIE30	GIE29	GIE28	GIE27	GIE26	GIE25	GIE24	GIE23	GIE22	GIE21	GIE20	GIE19	GIE18	GIE17	GIE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GIE15	GIE14	GIE13	GIE12	GIE11	GIE10	GIE9	GIE8	GIE7	GIE6	GIE5	GIE4	GIE3	GIE2	GIE1	GIE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 GIEn	MUA General Purpose Interrupt Enable n • There are 32 General Purpose Interrupts (n = 0 to 31) • If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor A when the GIPn bit in the MUA GSR register is set to "1".

Table continues on the next page...

Field	Function
	• If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request will be issued. • GIEn bit is cleared when the MU resets. 0b - Disables MUA General Interrupt n. 1b - Enables MUA General Interrupt n.

61.3.1.11 General Control Register (GCR)

Offset

Register	Offset
GCR	114h

Function

The GCR register contains the MUA general purpose interrupt requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GIR31	GIR30	GIR29	GIR28	GIR27	GIR26	GIR25	GIR24	GIR23	GIR22	GIR21	GIR20	GIR19	GIR18	GIR17	GIR16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GIR15	GIR14	GIR13	GIR12	GIR11	GIR10	GIR9	GIR8	GIR7	GIR6	GIR5	GIR4	GIR3	GIR2	GIR1	GIR0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 GIRn	MUA General Purpose Interrupt Request n • There are 32 General Purpose Interrupts (n = 0 to 31) • Writing "1" to the GIRn bit sets the MUB_GSR[GIPn] bit on the MUB-side. If the MUB_GIER[GIEn] bit is set to "1" on the MUB-side, a General Purpose Interrupt n request is triggered to Processor B. • The GIRn bit is cleared if the MUB_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUA that the interrupt was accepted (cleared by the software). • Writing "0" to the GIRn bit is ignored.

Table continues on the next page...

Field	Function
	• To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit). • GIRn bit is cleared when the MU resets. 0b - MUA General Interrupt n is not requested to the MUB. 1b - MUA General Interrupt n is requested to the MUB.

61.3.1.12 General Status Register (GSR)

Offset

Register	Offset
GSR	118h

Function

The GSR register contains the status of the MUA general interrupt pending requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GIP31	GIP30	GIP29	GIP28	GIP27	GIP26	GIP25	GIP24	GIP23	GIP22	GIP21	GIP20	GIP19	GIP18	GIP17	GIP16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GIP15	GIP14	GIP13	GIP12	GIP11	GIP10	GIP9	GIP8	GIP7	GIP6	GIP5	GIP4	GIP3	GIP2	GIP1	GIP0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 GIPn	MUA General Interrupt Request Pending n • There are 32 General Purpose Interrupts (n = 0 to 31) • GIPn bit signals the MUA that the MUB_GCR[GIRn] bit was set from "0" to "1". If the MUA_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor A. • The GIPn bit is cleared by writing "1". Writing "0" is ignored. • GIPn bit is cleared when the MU resets. • After GIPn bit is cleared, the General Interrupt n request(if the MUA_GIER[GIEn] bit is set to "1") is cleared on the MUA side.

Table continues on the next page...

Field	Function
	0b - MUA general purpose interrupt n is not pending. 1b - MUA general purpose interrupt n is pending.

61.3.1.13 Transmit Control Register (TCR)

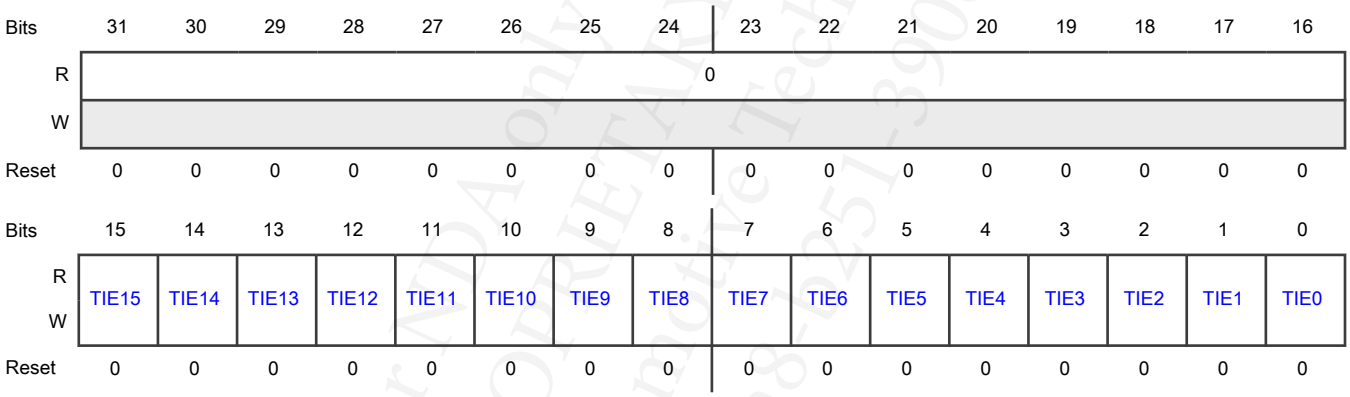
Offset

Register	Offset
TCR	120h

Function

The TCR register contains the MUA transmit interrupt enables.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TIEn	MUA Transmit Interrupt Enable n • There are 16 Transmit Interrupts (n = 0 to 15) • If TIEn bit is set to "1", then an MUA Transmit Interrupt n request is issued when the MUA_TSR[TEn] bit is set to "1". • If TIEn bit is cleared, then the value of the TEn bit is ignored and no MUA Transmit Interrupt n request will be issued. • TIEn bit is cleared when the MU resets. 0b - Disables MUA Transmit Interrupt n. (default) 1b - Enables MUA Transmit Interrupt n.

61.3.1.14 Transmit Status Register (TSR)

Offset

Register	Offset
TSR	124h

Function

The TSR register shows whether the MUA transmit registers are empty or not.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TE15	TE14	TE13	TE12	TE11	TE10	TE9	TE8	TE7	TE6	TE5	TE4	TE3	TE2	TE1	TE0
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-16 —	Reserved
15-0 TEn	MUA Transmit Register n Empty • There are 16 Transmit Registers (n = 0 to 15) • The TEn bit is set to "1" after the MUB RRn register is read on the MUB side. • After the TEn bit is set to "1", it signals the MUA side that the MUA TRn register is ready to be written on the MUA side, and a Transmit n interrupt is issued on the MUA side (if the MUA_TCR[TIEn] bit is set to "1"). • TEn bit is cleared after the MUA TRn register is written on the MUA side. • After TEn bit is cleared, the Transmit n interrupt(if the MUA_TCR[TIEn] bit is set to "1") is cleared on the MUA side. • TEn bit is set to "1" when the MU resets. 0b - MUA TRn register is not empty. 1b - MUA TRn register is empty.

Provided under NDA
COMPANY PROPRIETARY
Beijing ESWIN Automotive Technology Co., Ltd.
c8c26e54-4921-48a8-b251-30fdaab624c1

61.3.1.15 Receive Control Register (RCR)

Offset

Register	Offset
RCR	128h

Function

The RCR register contains the MUA receive interrupt enables.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RIE15	RIE14	RIE13	RIE12	RIE11	RIE10	RIE9	RIE8	RIE7	RIE6	RIE5	RIE4	RIE3	RIE2	RIE1	RIE0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 RIEn	MUA Receive Interrupt Enable n • There are 16 Transmit Interrupts (n = 0 to 15) • If RIEn bit is set to "1", then an MUA Receive Interrupt n request is issued when the MUA_RSR[RFn] bit is set to "1". • If RIEn bit is cleared, then the value of the RFn bit is ignored and no MUA Receive Interrupt n request will be issued. • RIEn bit is cleared when the MU resets. 0b - Disables MUA Receive Interrupt n. 1b - Enables MUA Receive Interrupt n.

Provided under NDA only
COPYRIGHT PROPRIETARY
Beijing ESAIL Automotive Technology Co., Ltd.
c8c46f2e-4c21-48a8-b251-39dda6b624c1

61.3.1.16 Receive Status Register (RSR)

Offset

Register	Offset
RSR	12Ch

Function

The RSR register shows whether the MUA receive registers are full or not.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RF15	RF14	RF13	RF12	RF11	RF10	RF9	RF8	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 RFn	MUA Receive Register n Full • There are 16 Receive Registers (n = 0 to 15) • The RFn bit is set to "1" when the MUB TRn register is written on the MUB side. • After the RFn bit is set to "1", the RFn bit signals the MUA side that new data in the MUA RRn register is ready to be read by the MUA, and a Receive n interrupt is issued on the MUA side (if the MUA_RCR[RIEn] has been set to "1"). • RFn bit is cleared when the MUA RRn register is read, or when the MU is reset. • After RFn bit is cleared, the Receive n interrupt(if the MUA_RCR[RIEn] has been set to "1") is cleared on the MUA side. 0b - MUA RRn register is not full. 1b - MUA RRn register has received data from MUB TRn register and is ready to be read by the MUA.

Provided under NDA
COPYRIGHT PROPRIETARY
Beijing Unisoc Automotive Technology Co., Ltd.
c8c46265-2e91-48a8-b251-30fdaab624c1

61.3.1.17 Transmit Register (TR0 - TR15)

Offset

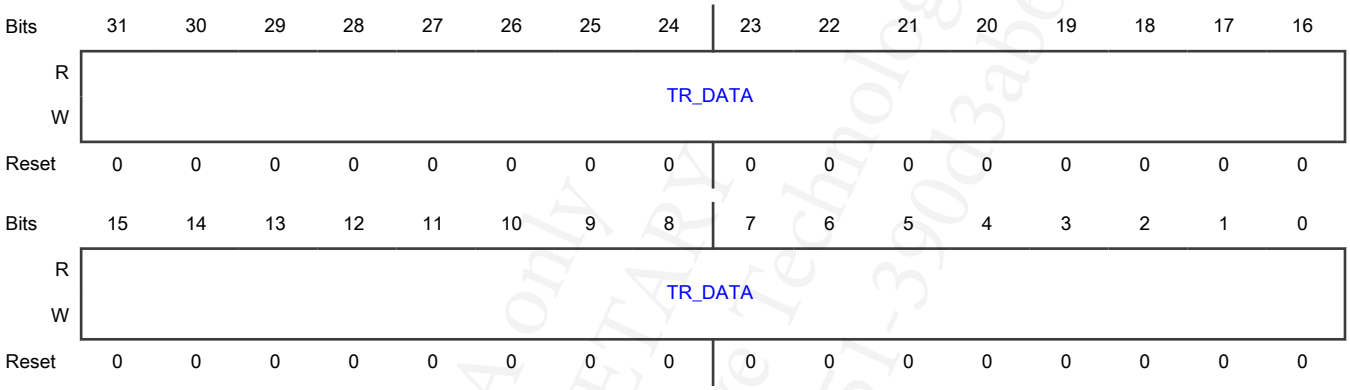
For n = 0 to 15:

Register	Offset
TRn	200h + (n × 4h)

Function

MUA Transmit Data.

Diagram



Fields

Field	Function
31-0 TR_DATA	MUA Transmit Data • Data written to the TRn register is reflected in the MUB Receive Register n (RRn). The TRn and RRn registers are not double-buffered, a write to the TRn register overrides the data readable at the RRn register. • A write to the transmit register clears the MUA_TSR[TEn] bit on the transmitter side, and sets the MUB_RSR[RFn] bit on the receiver side. • TRn register can be written only when the MUA_TSR[TEn] bit is set to "1". • Reading the TRn register returns all zeros.

61.3.1.18 Receive Register (RR0 - RR15)

Offset

For n = 0 to 15:

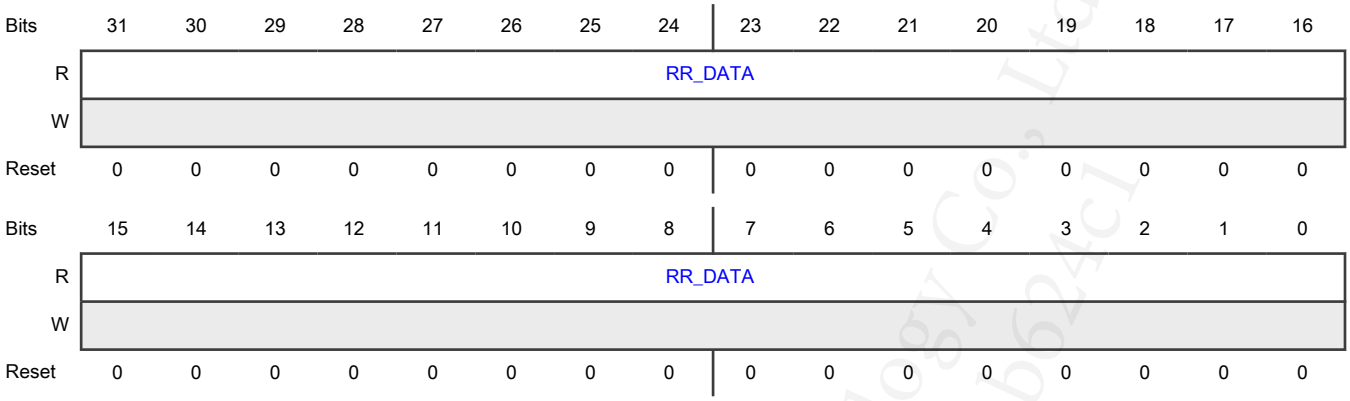
Register	Offset
RRn	280h + (n × 4h)

Preliminary NDA only
COMPANY PROPRIETARY
Beijing C-H Automotive Technology Co., Ltd.
c8c462e5-1a2-4a8-b251-3903ab624c1

Function

MUA Receive Data.

Diagram



Fields

Field	Function
31-0 RR_DATA	MUA Receive Data • Reflects the data written to MUB Transmit Register (TRn). • Reading the RRn register clears the MUA_RSR[RFn] bit on the receiver side, and sets the MUB_TSR[TEn] bit on the transmitter side. • RRn register can be read only when the MUB_RSR[RFn] bit is set to "1". Reading before MUB_RSR[RFn]=1 may result in reading incorrect data. Therefore, polling the MUB_RSR[RFn] bit to confirm it is set to "1" before reading RRn is required. • Writing to the RRn register generates an error response to the MUA.

61.3.2 MUB register descriptions

This section contains the detailed register descriptions for the MUB registers.

NOTE

A read/write access to any illegal location and a write to a read-only register on the Processor A-side or Processor B-side of the MU will generate a module transfer error acknowledge to the Processor A or Processor B. The only exception to this is at address offsets 0x1C and 0x14 - no transfer error will occur when accessing these locations.

61.3.2.1 MUB memory map

MU0.MUB base address: 4021_0000h

MU1.MUB base address: 4021_1000h

MU2.MUB base address: 4021_2000h

MU3.MUB base address: 4021_3000h

Preliminary PROPRIETARY COMPANY CONFIDENTIAL Beijing CHA Automotive Technology Co., Ltd. c8c462e5-45f4-418a-8351-390d3ab624c1

Offset	Register	Width (In bits)	Access	Reset value
0h	Version ID Register (VER)	32	RO	0300_000Fh
4h	Parameter Register (PAR)	32	RO	2020_1010h
8h	Control Register (CR)	32	RW	0000_0000h
Ch	Status Register (SR)	32	W1C	See description
10h	Core Control Register 0 (CCR0)	32	RW	0000_0000h
18h	Core Sticky Status Register 0 (CSSR0)	32	W1C	0000_0000h
100h	Flag Control Register (FCR)	32	RW	0000_0000h
104h	Flag Status Register (FSR)	32	RO	0000_0000h
110h	General Interrupt Enable Register (GIER)	32	RW	0000_0000h
114h	General Control Register (GCR)	32	RW	0000_0000h
118h	General Status Register (GSR)	32	W1C	0000_0000h
120h	Transmit Control Register (TCR)	32	RW	0000_0000h
124h	Transmit Status Register (TSR)	32	RO	0000_FFFFh
128h	Receive Control Register (RCR)	32	RW	0000_0000h
12Ch	Receive Status Register (RSR)	32	RO	0000_0000h
200h - 23Ch	Transmit Register (TR0 - TR15)	32	RW	0000_0000h
280h - 2BCh	Receive Register (RR0 - RR15)	32	RO	0000_0000h

61.3.2.2 Version ID Register (VER)

Offset

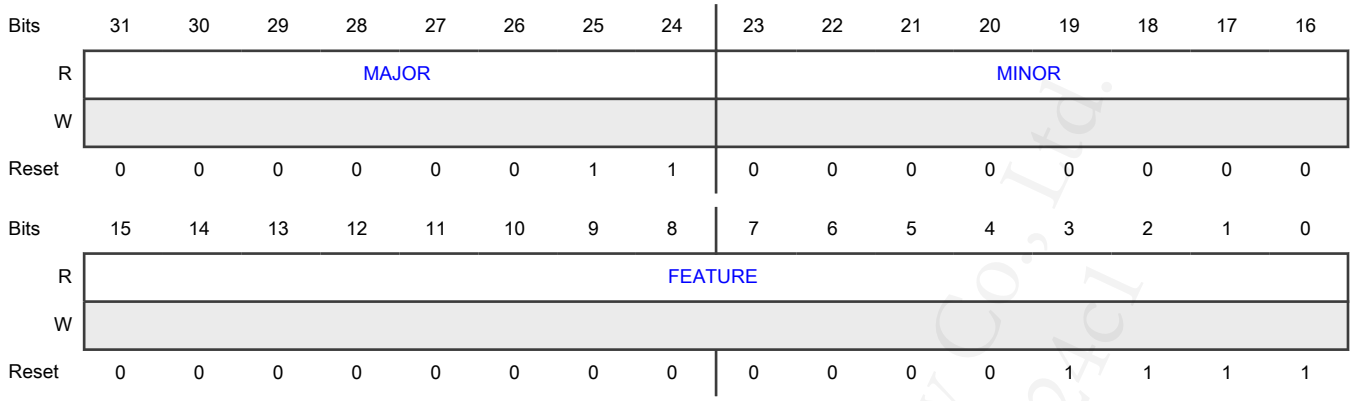
Register	Offset
VER	0h

Function

Use Version ID register to determine the version ID and feature set number of MUB.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e3-4821-48a8-b251-29d3ab6d4c1

Diagram



Fields

Field	Function
31-24 MAJOR	Major Version Number
23-16 MINOR	Minor Version Number
15-0 FEATURE	Feature Set Number Each bit of the Feature Set Number bitfield specifies a different feature: • If bit0 = 1: Standard features are implemented • If bit1 = 1: RAIP/RAIE register bits are implemented • If bit2 = 1: Core Control and Status Registers are implemented in both MUA and MUB • If bit3 = 1: Expand TRn/RRn registers number

61.3.2.3 Parameter Register (PAR)

Offset

Register	Offset
PAR	4h

Function

The PAR register defines the number of flags, transmit registers, receive registers, and the number of general interrupt requests available for the MU.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a2-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FLAG_WIDTH								GIR_NUM							
W																
Reset	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RR_NUM								TR_NUM							
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0

Fields

Field	Function
31-24 FLAG_WIDTH	Flag Width This bitfield specifies the number of flag bits in the Flag Control Register (FCR) and Flag Status Register (FSR).
23-16 GIR_NUM	General Interrupt Request Number This bitfield specifies the number of general interrrupt requests available.
15-8 RR_NUM	RR Number This bitfield specifies the number of receive registers.
7-0 TR_NUM	Transmit Register Number This bitfield specifies the number of transmit registers.

61.3.2.4 Control Register (CR)

Offset

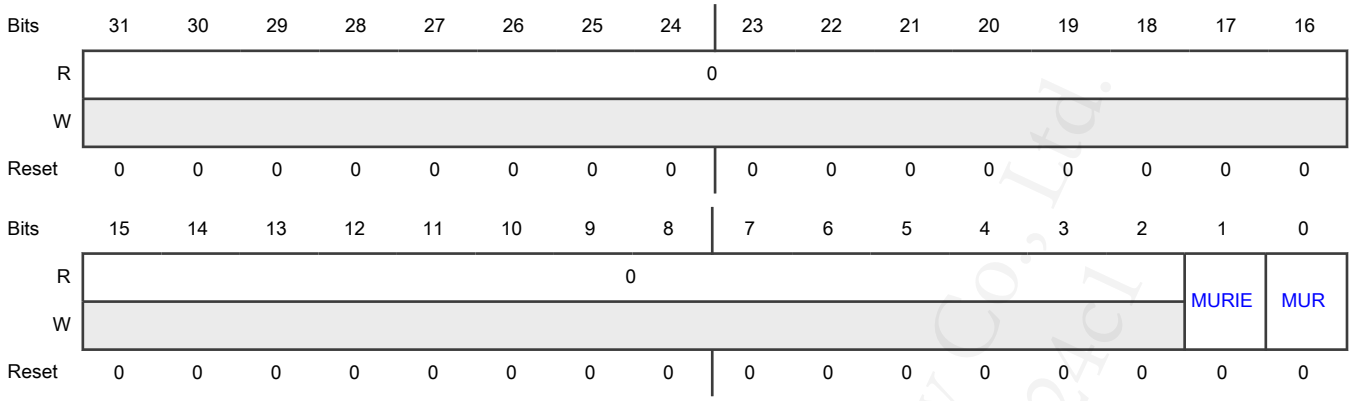
Register	Offset
CR	8h

Function

The CR register controls the MU reset and MU reset interrupt enable.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 —	Reserved
1 MURIE	MUB Reset Interrupt Enable • If MURIE bit is set to "1", then MU reset interrupt request is issued to the Processor B when the MUB_SR[MURIP] bit is set to "1". • If MURIE is cleared, then the value of the MURIP bit is ignored and no MU reset interrupt request will be issued. • Only system reset can reset MURIE, MU reset(MUR) cannot reset this bit. 0b - Disables Processor B-side MU Reset Interrupt request due to MU reset issued by MUA. 1b - Enables Processor B-side MU Reset Interrupt request due to MU reset issued by MUA.
0 MUR	MU Reset • Setting MUR bit to "1" resets both the MUB-side and MUA-side, forcing all control and status registers to return to their default values (except in MUB/A_CCR0 registers, MURIE in MUB/A_CR registers, MURIP bit and MURS bit in MUB/A_SR registers), and all internal states to be cleared. • Before setting the MUR bit to "1", it is advisable to interrupt the Processor A, because setting the MUR bit may affect the ongoing Processor A program. • After setting the MUR bit, monitor the value of the MUB_SR[MURS] bit to know when the reset sequence on the Processor A-side has ended. • MUR bit is always read as "0". • MUR bit is cleared during the MU reset sequence. 0b - Self clearing bit. 1b - Asserts the MU reset.

Provided under NDA only
COMPANY CONFIDENTIAL
Beijing ChipRoute Automotive Technology Co., Ltd.
c8c4625c-48e8-4a12-1390d8ab624c1

61.3.2.5 Status Register (SR)

Offset

Register	Offset
SR	Ch

Function

The SR register shows the status of MU resets and the status of pending events/requests described below.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	RFP	TEP	GIRP	FUP	EP	MURIP	MURS
W															W1C	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	u

Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6 RFP	MUB Receive Full Pending Flag • RFP bit is set to "1" when any TRn register is written by MUA to send data to MUB. • After the RFP bit is set to "1", checking the RSR[RFn] bit to decide the data in which RRn register is ready to be read by the MUB. • RFP bit is cleared when all MUB RRn registers are read, or when the MU is reset. 0b - No TRn register is written by MUA. 1b - Any TRn register is written by MUA.
5 TEP	MUB Transmit Empty Pending • TEP bit is set to "1" when TCR[TIEn] bit is set and the corresponding RRn register is read by MUA.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• After the TEP bit is set to "1", checking the TSR[TEn] bit to decide the data in which TRn register is ready to be written by the MUB. • TEP bit is cleared when all MUB TRn register are written. • TEP bit is set to "0" when the MU resets. 0b - RRn register is not read by MUA. 1b - Any RRn register is read by MUA.
4 GIRP	MUB General Interrupt Pending • GIRP bit is set to "1" when any general interrupt request is sent from the MUA side to the MUB side. Reading the GSR[GIPn] bit can know which general interrupt request is received. • GIRP bit is cleared when all GIPn bits in MUB_GSR register are cleared. • GIRP bit is cleared when the MU resets. 0b - No general interrupt request is sent from MUA. 1b - Any general interrupt request is sent from MUA.
3 FUP	MUB Flags Update Pending • FUP bit is set to "1" when the MUB side sends a Flags Update request to the MUA side. • A Flags Update request is generated when there is a change to the Fn[31:0] bits of the MUB_FCR register. No flag update changes are allowed while the FUP bit is set to "1". Any write to the Fn[31:0] bits of the MUB_FCR register, while the FUP bit is set to "1", will not generate a Flags Update event, and the Fn[31:0] bits will stay unchanged. • Writing the FCR register doesn't set FUP bit immediately if there is any event pending(SR[EP] is "1"). • FUP bit is cleared when this Flags Update request is internally acknowledged (the flag is updated) from the MUA side, or during MU reset. 0b - No pending update flags(initiated by MUB) are in process 1b - Pending update flags(initiated by MUB) are in process
2 EP	MUB Side Event Pending • An "event" is any hardware message that is reflected in the status register on the MUA side (for example, "transmit register 0 written"). During normal operations, the update mechanism for the EP bit works automatically. • EP bit is set to "1" when the MUB side sends an event update request to the MUA side. • EP bit is cleared by hardware automatically when the event update acknowledge is received. • To ensure events have been posted to MUA, verify that the EP bit is cleared. If EP bit is set to "1", wait and continue to poll the EP bit. • The EP bit is cleared when the MU resets. 0b - The MUB side event is not pending. 1b - The MUB side event is pending.

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 MURIP	MU Reset Interrupt Pending • MURIP bit signals Processor B that Processor A initiated a MU reset by setting the MUA_CR[MUR] bit. • MURIP bit is set to "1" after Processor A initiated a MU reset. If the interrupt is enabled by the MURIE bit, the Processor B MU reset interrupt request is issued when the Processor A set MUR bit in MUA_CR. • Writing "1" to clear the MURIP bit, which also clears MU reset interrupt request. • Only system reset can reset MURIP. MU reset cannot reset this bit. 0b - Processor A did not issue MU reset. 1b - Processor A issued MU reset.
0 MURS	MUA and MUB Reset State • The MURS bit is set to "1" during any system reset or any MU reset from MUA-side or MUB-side. • The MURS bit is cleared when the reset sequence on both MUA and MUB side ends. After issuing any of the reset events mentioned previously, **verify that the MURS bit is cleared before starting any access.** 0b - MUA and MUB are out of reset. 1b - MUA or MUB is in reset state.

61.3.2.6 Core Control Register 0 (CCR0)

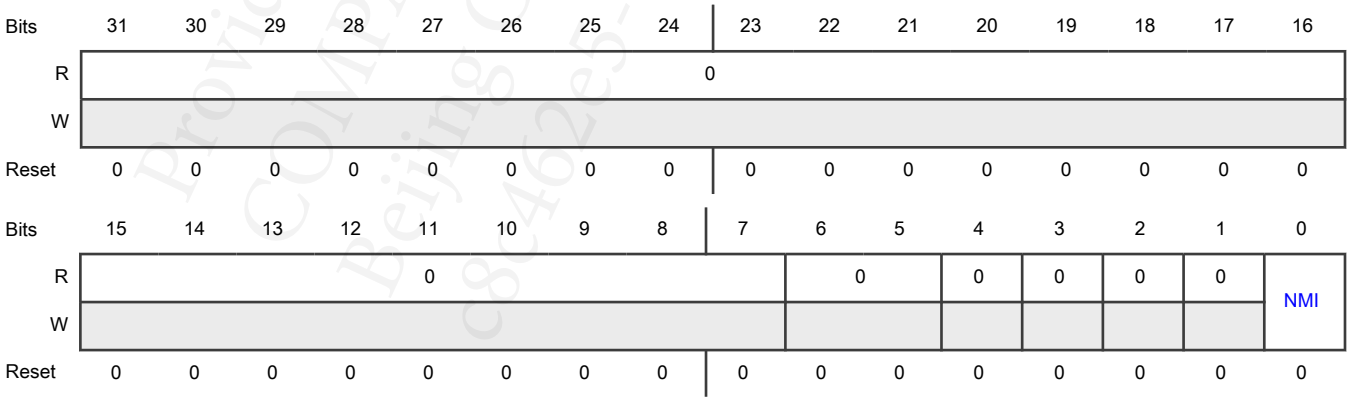
Offset

Register	Offset
CCR0	10h

Function

The CCR0 enables the MUB to control the Processor on the MUA-side.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 NMI	MUA Non-maskable Interrupt Request • When NMI bit is set to "1", it initiates a Non-Maskable Interrupt to the Processor A. • NMI bit is cleared after the MUA_CSSR0[NMIC] bit is written to "1". After the NMI bit is cleared, the MUB can initiate another non-maskable interrupt to the MUA. • The NMI bit is cleared when the MU resets. 0b - Non-maskable interrupt is not issued to the Processor A by the Processor B. 1b - Non-maskable interrupt is issued to the Processor A by the Processor B.

61.3.2.7 Core Sticky Status Register 0 (CSSR0)

Offset

Register	Offset
CSSR0	18h

Function

The CSSR0 register shows the status of interrupts pending (W1C).

Provided under nonDISCLOSURE AGREEMENT
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48c8-b651-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0	0	0	0	0	0	0	NMIC
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 NMIC	Processor B Non-Maskable-Interrupt Clear • NMIC bit is used by the MUB-side Non-Maskable Interrupt service routine to clear the non-maskable interrupt from MUA-side.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cc46e5-4821-48a8-b251-390d8ab624d1

Table continued from the previous page...

Field	Function
	• Writing NMIC bit as "1" signals the MUA to clear the MUA_CCR0[NMI] bit, thus de-asserting the interrupt and enabling the NMI bit to receive another interrupt. • The NMIC bit is always read as "0", therefore NMIC bit cannot be polled. The NMIC bit can only be used as part of the interrupt service routine, in which the NMIC bit should only be written as "1" once. • The NMIC bit is cleared when the MU resets. 0b - Default 1b - Writing "1" clears the MUA_CCR0[NMI] bit.

61.3.2.8 Flag Control Register (FCR)

Offset

Register	Offset
FCR	100h

Function

The FCR register contains read-write flags that reflect the MUA_FSR[Fn] bits.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	F31	F30	F29	F28	F27	F26	F25	F24	F23	F22	F21	F20	F19	F18	F17	F16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 Fn	MUB to MUA Flag n • n = 0 to 31 • Fn bit is a read-write flag that reflects the MUA_FSR[Fn] bits.

Table continues on the next page...

Field	Function
	• Fn bits are cleared when the MU resets. • Write 0 to an Fn bit - Clears the Fn bit in the MUA_FSR register. • Write 1 to an Fn bit - Sets the Fn bit in the MUA_FSR register. 0b - Clears the Fn bit in the FSR register. 1b - Sets the Fn bit in the FSR register.

61.3.2.9 Flag Status Register (FSR)

Offset

Register	Offset
FSR	104h

Function

The FSR register contains flags that reflect the values written to the Fn bit of the MUA_FCR register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	F31	F30	F29	F28	F27	F26	F25	F24	F23	F22	F21	F20	F19	F18	F17	F16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	F15	F14	F13	F12	F11	F10	F9	F8	F7	F6	F5	F4	F3	F2	F1	F0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 Fn	MUA to MUB Side Flag n • n = 0 to 31 • Fn bit is the MUB side flag that reflects the values written to the Fn bit in the MUA_FCR register. • Every time that the Fn bit in the MUA_FCR register is written, the Fn bit in the MUA FCR register write event updates the Fn bit in the MUB_FSR register after the event update latency. • If a 0 is read from an Fn bit - Fn bit in the MUA_FCR is 0.

Table continues on the next page...

Field	Function
	• If a 1 is read from an Fn bit - Fn bit in the MUA_FCR is 1. 0b - Fn bit in the MUA FCR register is written 0. 1b - Fn bit in the MUA FCR register is written 1.

61.3.2.10 General Interrupt Enable Register (GIER)

Offset

Register	Offset
GIER	110h

Function

The GIER register contains the MUB general purpose interrupt enables.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GIE31	GIE30	GIE29	GIE28	GIE27	GIE26	GIE25	GIE24	GIE23	GIE22	GIE21	GIE20	GIE19	GIE18	GIE17	GIE16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GIE15	GIE14	GIE13	GIE12	GIE11	GIE10	GIE9	GIE8	GIE7	GIE6	GIE5	GIE4	GIE3	GIE2	GIE1	GIE0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 GIEn	MUB General Purpose Interrupt Enable n • There are 32 General Purpose Interrupts (n = 0 to 31) • If a GIEn bit is set to "1", then a General Interrupt n request is issued to Processor B when the GIPn bit in the MUB GSR register is set to "1". • If GIEn is cleared, then the value of the GIPn bit is ignored and no General Interrupt n request will be issued. • GIEn bit is cleared when the MU resets. 0b - Disables MUB General Interrupt n. 1b - Enables MUB General Interrupt n.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c6242a821-48a9-9023-1990d3ab624c1

61.3.2.11 General Control Register (GCR)

Offset

Register	Offset
GCR	114h

Function

The GCR register contains the MUB general purpose interrupt requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	GIR31	GIR30	GIR29	GIR28	GIR27	GIR26	GIR25	GIR24	GIR23	GIR22	GIR21	GIR20	GIR19	GIR18	GIR17	GIR16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	GIR15	GIR14	GIR13	GIR12	GIR11	GIR10	GIR9	GIR8	GIR7	GIR6	GIR5	GIR4	GIR3	GIR2	GIR1	GIR0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 GIRn	MUB General Purpose Interrupt Request n • There are 32 General Purpose Interrupts (n = 0 to 31) • Writing "1" to the GIRn bit sets the MUA_GSR[GIPn] bit on the MUA-side. If the MUA_GIER[GIEn] bit is set to "1" on the MUA-side, a General Purpose Interrupt n request is triggered to Processor A. • The GIRn bit is cleared if the MUA_GSR[GIPn] bit is cleared by writing "1", thereby signalling the MUB that the interrupt was accepted (cleared by the software). • Writing "0" to the GIRn bit is ignored. • To ensure proper operations, you must verify that the GIRn bit is cleared (meaning that there is no pending interrupt) before setting it (GIRn bit). • GIRn bit is cleared when the MU resets. 0b - MUB General Interrupt n is not requested to the MUA. 1b - MUB General Interrupt n is requested to the MUA.

61.3.2.12 General Status Register (GSR)

Offset

Register	Offset
GSR	118h

Function

The GSR register contains the status of the MUB general interrupt pending requests.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GIP31	GIP30	GIP29	GIP28	GIP27	GIP26	GIP25	GIP24	GIP23	GIP22	GIP21	GIP20	GIP19	GIP18	GIP17	GIP16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GIP15	GIP14	GIP13	GIP12	GIP11	GIP10	GIP9	GIP8	GIP7	GIP6	GIP5	GIP4	GIP3	GIP2	GIP1	GIP0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 GIPn	MUB General Interrupt Request Pending n • There are 32 General Purpose Interrupts (n = 0 to 31) • GIPn bit signals the MUB that the MUA_GCR[GIRn] bit was set from "0" to "1". If the MUB_GIER[GIEn] bit is set to "1", a General Interrupt n request is issued to Processor B. • The GIPn bit is cleared by writing "1". Writing "0" is ignored. • GIPn bit is cleared when the MU resets. • After GIPn bit is cleared, the General Interrupt n request(if the MUB_GIER[GIEn] bit is set to "1") is cleared on the MUB side. 0b - MUB general purpose interrupt n is not pending. 1b - MUB general purpose interrupt n is pending.

61.3.2.13 Transmit Control Register (TCR)

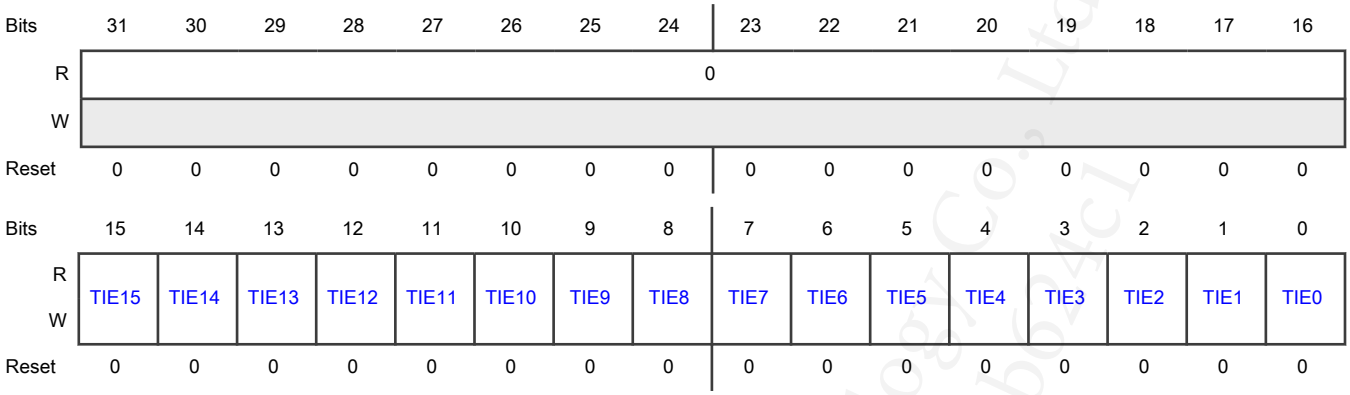
Offset

Register	Offset
TCR	120h

Function

The TCR register contains the MUB transmit interrupt enables.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TIEn	MUB Transmit Interrupt Enable n • There are 16 Transmit Interrupts (n = 0 to 15) • If TIEn bit is set to "1", then an MUB Transmit Interrupt n request is issued when the MUB_TSR[TEn] bit is set to "1". • If TIEn bit is cleared, then the value of the TEn bit is ignored and no MUB Transmit Interrupt n request will be issued. • TIEn bit is cleared when the MU resets. 0b - Disables MUB Transmit Interrupt n. (default) 1b - Enables MUB Transmit Interrupt n.

61.3.2.14 Transmit Status Register (TSR)

Offset

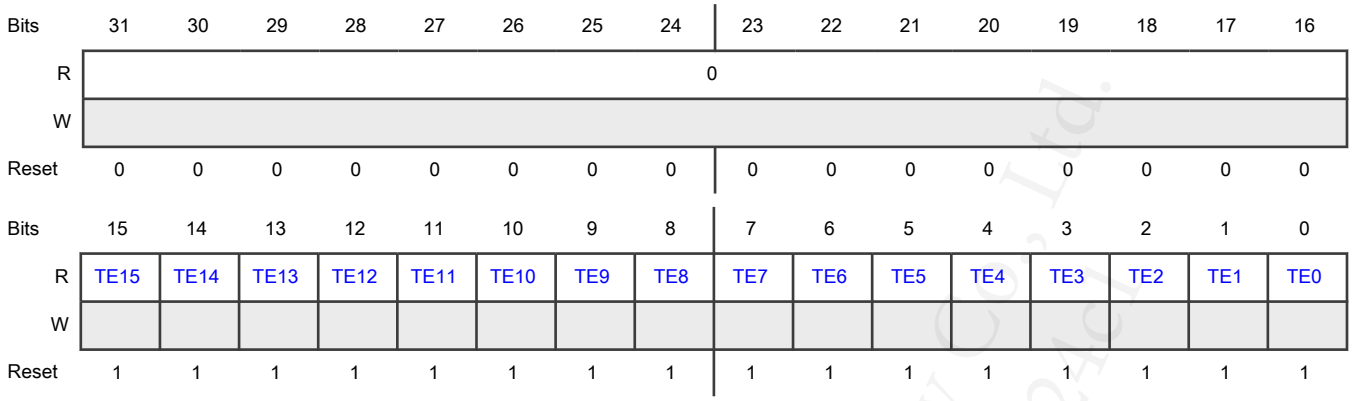
Register	Offset
TSR	124h

Function

The TSR register shows whether the MUB transmit registers are empty or not.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 TEn	MUB Transmit Register n Empty • There are 16 Transmit Registers (n = 0 to 15) • The TEn bit is set to "1" after the MUA RRn register is read on the MUA side. • After the TEn bit is set to "1", it signals the MUB side that the MUB TRn register is ready to be written on the MUB side, and a Transmit n interrupt is issued on the MUB side (if the MUB_TCR[TIEn] bit is set to "1"). • TEn bit is cleared after the MUB TRn register is written on the MUB side. • After TEn bit is cleared, the Transmit n interrupt(if the MUB_TCR[TIEn] bit is set to "1") is cleared on the MUB side. • TEn bit is set to "1" when the MU resets. 0b - MUB TRn register is not empty. 1b - MUB TRn register is empty.

61.3.2.15 Receive Control Register (RCR)

Offset

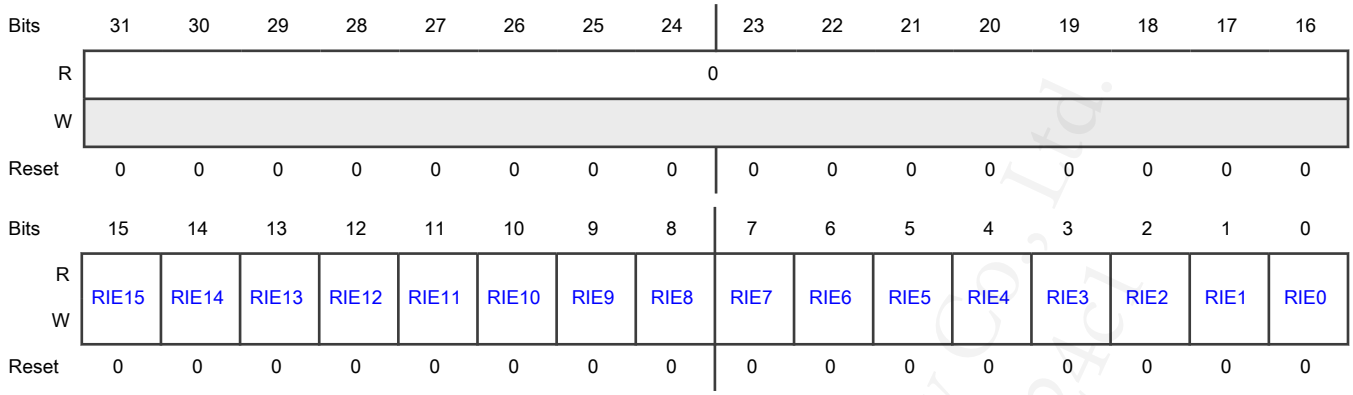
Register	Offset
RCR	128h

Function

The RCR register contains the MUB receive interrupt enables.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b25-390d8ab624c

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 RIEn	MUB Receive Interrupt Enable n • There are 16 Transmit Interrupts (n = 0 to 15) • If RIEn bit is set to "1", then an MUB Receive Interrupt n request is issued when the MUB_RSR[RFn] bit is set to "1". • If RIEn bit is cleared, then the value of the RFn bit is ignored and no MUB Receive Interrupt n request will be issued. • RIEn bit is cleared when the MU resets. 0b - Disables MUB Receive Interrupt n. 1b - Enables MUB Receive Interrupt n.

61.3.2.16 Receive Status Register (RSR)

Offset

Register	Offset
RSR	12Ch

Function

The RSR register shows whether the MUB receive registers are full or not.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RF15	RF14	RF13	RF12	RF11	RF10	RF9	RF8	RF7	RF6	RF5	RF4	RF3	RF2	RF1	RF0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 RFn	MUB Receive Register n Full • There are 16 Receive Registers (n = 0 to 15) • The RFn bit is set to "1" when the MUA TRn register is written on the MUA side. • After the RFn bit is set to "1", the RFn bit signals the MUB side that new data in the MUB RRn register is ready to be read by the MUB, and a Receive n interrupt is issued on the MUB side (if the MUB_RCR[RIEn] has been set to "1"). • RFn bit is cleared when the MUB RRn register is read, or when the MU is reset. • After RFn bit is cleared, the Receive n interrupt(if the MUB_RCR[RIEn] has been set to "1") is cleared on the MUB side. 0b - MUB RRn register is not full. 1b - MUB RRn register has received data from MUA TRn register and is ready to be read by the MUB.

61.3.2.17 Transmit Register (TR0 - TR15)

Offset

For n = 0 to 15:

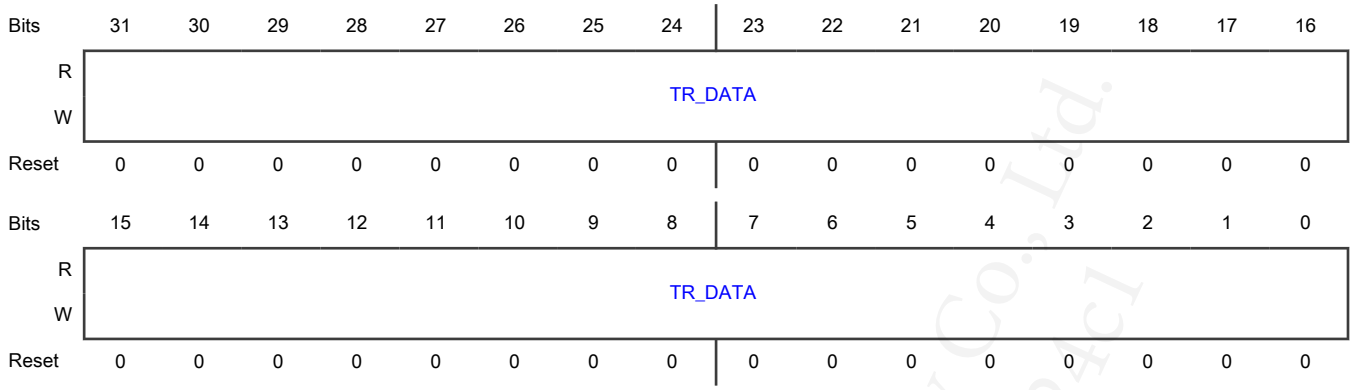
Register	Offset
TRn	200h + (n × 4h)

Function

MUB Transmit Data.

Provided by IHS Markit
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a2-b2c5-390d3ab624c7

Diagram



Fields

Field	Function
31-0 TR_DATA	MUB Transmit Data • Data written to the TRn register is reflected in the MUA Receive Register n (RRn). The TRn and RRn registers are not double-buffered, a write to the TRn register overrides the data readable at the RRn register. • A write to the transmit register clears the MUB_TSR[TEn] bit on the transmitter side, and sets the MUA_RSR[RFn] bit on the receiver side. • TRn register can be written only when the MUB_TSR[TEn] bit is set to "1". • Reading the TRn register returns all zeros.

61.3.2.18 Receive Register (RR0 - RR15)

Offset

For n = 0 to 15:

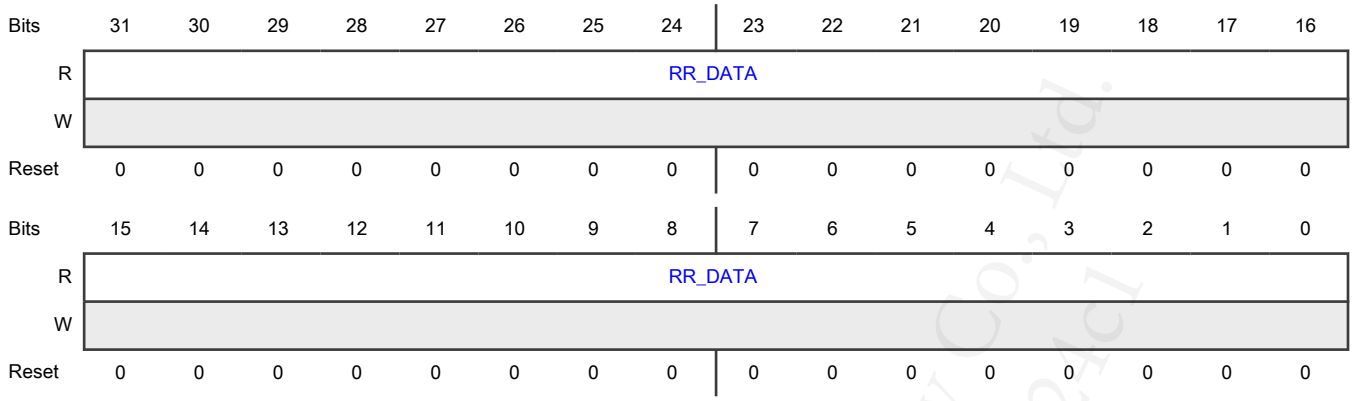
Register	Offset
RRn	280h + (n × 4h)

Function

MUB Receive Data.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-39d3ab624c1

Diagram



Fields

Field	Function
31-0 RR_DATA	MUB Receive Data • Reflects the data written to MUA Transmit Register (TRn). • Reading the RRn register clears the MUB_RSR[RFn] bit on the receiver side, and sets the MUA_TSR[TEn] bit on the transmitter side. • RRn register can be read only when the MUA_RSR[RFn] bit is set to "1". Reading before MUA_RSR[RFn]=1 may result in reading incorrect data. Therefore, polling the MUA_RSR[RFn] bit to confirm it is set to "1" before reading RRn is required. • Writing to the RRn register generates an error response to the MUB.

61.4 Functional Description

The Messaging Unit (MU) enables two cores (Processor A or Processor B) to communicate with each other, by passing message/data information to each other, and by enabling one core to wake up the other core using interrupts.

The messaging, control, and status registers of the Processor A and Processor B-side for the MU are mapped to the Processor A and Processor B memory as a regular peripheral. The Peripheral data bus is 32 bits wide inside the MU module.

The messaging logic is used in conjunction with external memory. There are various messaging methods that can be used to implement a messaging protocol. Some of these messages could mean “A message of N words has been written starting at offset X in the memory,” or "The previous data block that was sent has been read." Having the messaging logic independent from the memory array is not restricted to a predefined hardware protocol. On the other hand, the software needed to manage the messaging is short and straightforward.

Most of the messaging mechanisms are symmetric. They are duplicated and are available on both the Processor A-side and the Processor B-side. The messaging mechanisms are:

- 16 32-bit transmit registers, which are each reflected in 16 read-only receive registers in the other processor’s side. You can use these registers to transfer 32-bit word messages or frame information of messages written to the shared memory (number of words, initial address, and message type code).
- A write to a transmit register on the transmitter side clears a “transmitter empty” bit in the Status Register on the transmitter side, and sets a “receiver full” bit in the Status Register on the receiver side. The setting of the bit at the receiver side can optionally trigger an interrupt at the receiver side (maskable receive interrupt).

- A read of one of the receive registers at the receiver side clears the “receiver full” bit in the Status Register at the receiver side, and sets the “transmitter empty” bit in the Status Register on the transmitter side. The setting of the “transmitter empty” bit can optionally trigger an interrupt at the transmitter side (maskable transmit interrupt).
- 32 general purpose interrupt request flags are reflected in the General Status Register(GSR) on the receiver side.
- 32-bit flag data is transmitted from FCR register to xFSR register with the SR[FUP] indicator bit.

A write to a transmit register signals the receiver side that data is ready for retrieval.

- Writing to the transmit register again without verifying that the data was retrieved is prohibited, because the transmitter side has no way of knowing the exact time that the receiver will attempt to retrieve the data.
- Before attempting to write the transmit register again, the transmitter side should wait for a “Transmitter Empty” interrupt, or should poll the “Transmitter Empty” bit in the Transmit Status Register.

A read of a receive register signals the transmitter side that data can be written to that register. In the same way, the receiver processor should not read a receive register before receiving a “Receiver Full” interrupt or polling the “Receiver Full” bit in the Receive Status Register.

- Reading the receive register again without verifying that the data was written is prohibited, because the receiver side has no way of knowing the exact time that the transmitter will attempt to write the data.
- Before attempting to read the receive register again, the receiver side should wait for a “Receiver Full” interrupt, or should poll the “Receiver Full” bit in the Receive Status Register.

61.4.1 Messaging Examples

The following are some messaging examples:

- **Passing short messages:** Transmit register(s) can be used to pass short messages from one to 16 words in length for Processor A and 16 for Processor B. For example, when a four-word message is desired, only one of the registers needs to have its corresponding interrupt enable bit set at the receiver side; the message’s first three words are written to the registers whose interrupt is masked, and the fourth word is written to the other register (which triggers an interrupt at the receiver side).
- **Passing frame information:** Transmit registers can be used to pass frame information for long messages written to the shared system (for example, SRAM and FLASH). Such frame information normally includes a start address, number of words, and perhaps a message type code.
- **Passing event notices and requests:** Events and requests that do not include data words can be signaled from the Processor B to the Processor A using the general interrupts, such as acknowledging that a long message was read from the shared system memory.
- **Passing fixed length data:** Formatted data with a fixed length can be written in predetermined locations in the shared memory. A processor can use a general interrupt to signal the other processor that the data is ready.
- **Passing announcements:** The 32 flags can be used by a processor to announce its current program state or other billboard messages to the other processor.

61.4.1.1 Messaging Protocols using Interrupts

The example below describes a four-word messaging sequence sent by the Processor to the other Processor.

In this example, the first, second, and third receive interrupts are disabled, and the fourth receive interrupt is enabled. We write registers sequentially for n = 0, 1, 2, 3. For n = 0, 1, 2, the interrupts are disabled, therefore no interrupt will go to the other core (although interrupt conditions occur). For n = 3, the interrupt is enabled, and the last Receive Interrupt request is generated.

1. Write Sequence
   - The Processor writes the message information sequentially to its Transmit Registers 0, 1, 2.
   - When the write to the Transmit Register 3 occurs, the RF3 bit of the xRSR is set after synchronization, and it immediately trigger the Receive Full 3 interrupt to the other Processor.
2. Read Sequence

- The other Processor receives the Receive Full 3 interrupt and starts reading the message transferred from the receive registers.
- After Receive Register 3 is read, the interrupt bit is cleared.

The following table and diagram describe the messaging model using transmit/receive registers and interrupt messaging protocol.

Table 536. Interrupt Messaging Protocol (Generalized)

Sequence	Action	Description
1	Processor A Data write	A data write to the TRn register by Processor A is immediately reflected in the Processor B RRn register.
2	Clear Tx Empty bit and Set Rx Full bit	The data write to the TRn register • Clears the transmitter empty bit (TEn) in the Processor A Transmit Status Register • Sets the receiver full bit (RFn) in the Processor B Receive Status Register
3	Generate Receive Interrupt request	The setting of the receiver full bit (RFn) in the Receive Status Register generates a Receive Interrupt request to Processor B.
4	Processor B Data read	After receiving the Receive Interrupt request, Processor B performs a data read of the RRn register.
5	Clear Rx Full bit and Set Tx Empty bit	Reading the data out of the RRn register • Clears the receiver full bit (RFn) in the Processor B Receive Status Register • Sets the transmitter empty bit (TEn) in the Processor A Transmit Status Register
6	Generate Transmit Interrupt request	The setting of the transmitter empty bit (TEn) in the Transmit Status Register generates a Transmit Interrupt request to Processor A.

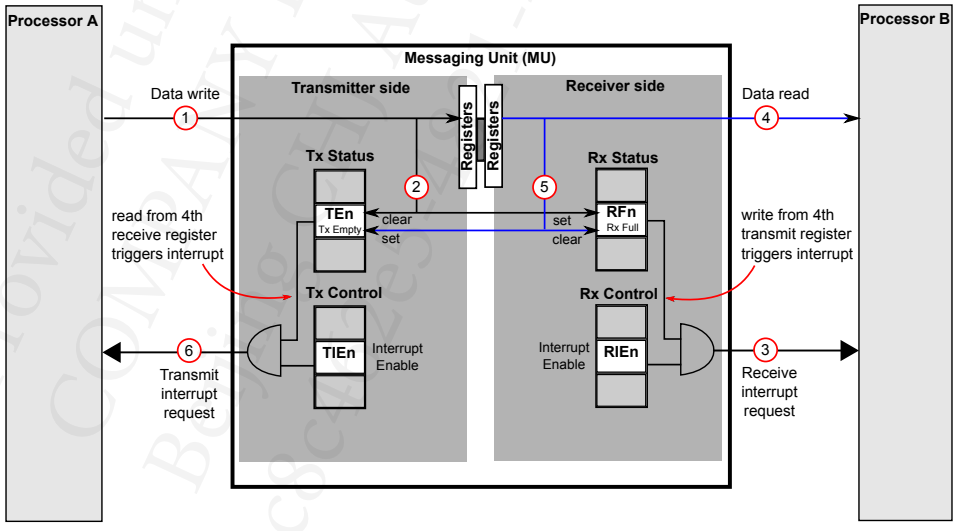


Figure 547. Messaging Model Using Transmit and Receive Registers

The messaging hardware can be used by software to implement messaging protocols for a wide array of message types. Full support is given for both interrupt and polling management schemes.

61.4.1.2 Messaging Protocols using Event Interrupts

Events and requests that do not include data words can be signaled from the Processor B to the Processor A using the two general interrupts.

Formatted data with a fixed length can be written in predetermined locations in the shared memory. A processor can use a general purpose interrupt to signal the other processor that the data is ready.

The 32 flags can be used by a processor to announce to the other processor the program state it is currently in, or to announce similar messages.

The following table and diagram describe the event sequence when the Processor triggers an interrupt.

Table 537. General Interrupt Messaging Protocol (Generalized)

Sequence	Action	Description
1	Processor A sets General Interrupt request bit	Processor A sets its associated General Interrupt request bit (GIRn = 1) in the General Control Register (GCR).
2	General Interrupt Request Pending status bit is set	The General Interrupt Request Pending status bit (GIPn) in the General Status Register (GSR) is set to "1".
3	General Interrupt request to Processor B is generated	Setting the GIPn bit generates the General Interrupt request to Processor B (Interrupt Request Enable bit, GIEn, must be set for Processor B).
4	Processor B reads status register	The Processor B reads the GIPn bit in the GSR register.
5	Processor B services the interrupt	-
6	Processor B sets GIPn bit to clear interrupt	The Processor B writes "1" to the corresponding GIPn bit to clear the interrupt.
7	GIRn bit is cleared	Setting the GIPn bit to "1" clears the General Interrupt request bit (GIRn) in the Processor A General Control Register (GCR).

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4ab2-b251-9d5b624c

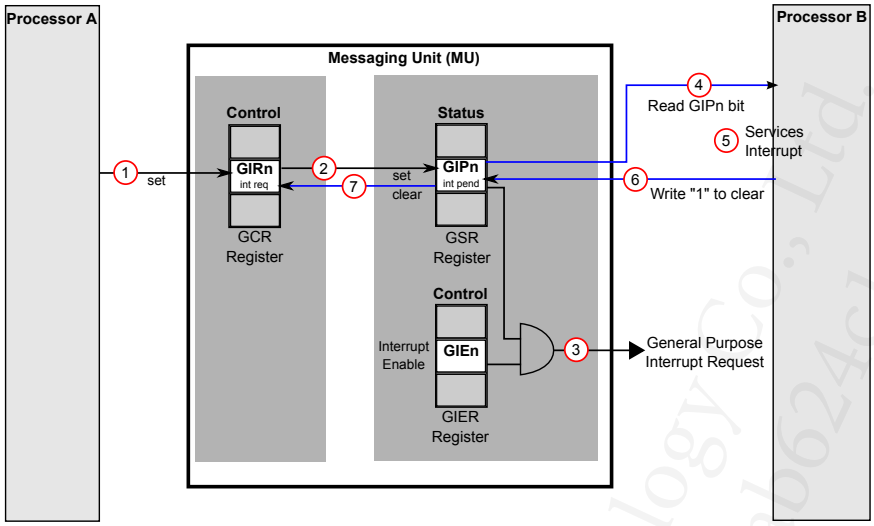


Figure 548. Messaging Model Using a General Purpose Interrupt

61.4.1.3 Exclusive Access to Shared Memory

MU can be used to signal one processor about its current access to the shared memory, so that the data is not overwritten by the other processor during the exclusive memory access period.

The following tables describe the signaling protocol that the Processor A uses to inform the Processor B about its current access (write) to the shared memory, assuming that the set of bits and registers (GIRn bit, RRn register, TRn register) are reserved to support exclusive access to the shared memory protocol.

Table 538. How the Processor A Performs an Exclusive Access to Shared Memory

Sequence	Action	Description
1	Processor A sends GIRn request to Processor B using Processor A control register	When the Processor A wants to perform an exclusive access to the shared memory, the Processor A sends an GIR0 request to the Processor B.
2	Processor A sends an exclusive-access request using a transmit data register (TRn)	The Processor A will send an exclusive-access request (command, location, and length of target access) to Processor B using a selected transmit data register (TR0).
3	Processor A waits for a dedicated interrupt from Processor B	The Processor A waits for a dedicated interrupt (as an acknowledgement) triggered by the Processor B before proceeding.
4	Processor A accesses shared memory	After receiving a dedicated interrupt from the Processor B, Processor A proceeds.

Table 539. How the Processor B Scans for Transaction Information

Sequence	Action	Description
1	Processor B receives an interrupt from a receive data register (RRn)	-
2	Processor B reads the receive data register (RRn)	-
3	Processor B scans the receive data register contents	For transaction information (whether Processor A has requested an exclusive-access).

Table 540. How the Processor B Accepts Exclusive Access by Processor A

Sequence	Action	Description
1	Processor B triggers a dedicated interrupt	Processor B acknowledges the Processor A request by triggering a dedicated interrupt (ack) to the Processor A.
2	Processor B sends a code message to Processor A	Along with the acknowledge interrupt, the Processor B sends a code message to the Processor A through the selected transmit register (TRn). The message informs the Processor A that it can exclusively access the shared memory.

Table 541. How the Processor B Rejects Exclusive Access by Processor A

Sequence	Action	Description
1	Processor B ignores Processor A request for exclusive access	If the Processor B does not want to give go-ahead permission to the Processor A, Processor B ignores the exclusive access request.

61.4.1.4 Packet Data Transfers

The following example describes the packet transfer sequence between the Processor B and Processor A subsystems:

Table 542. Packet Data Transfer Sequence

Action	Sequence	Description
Processor B requests DMA	1	The Processor B sends a DMA request to initiate the packet data transfer.
DMA data transfer	2	DMA acknowledges.
	3	DMA starts transferring data from the specified Processor B location to the specified shared memory
	4	DMA interrupts the Processor B to signal that the packet transfer has finished.

Table continues on the next page...

Table 542. Packet Data Transfer Sequence (continued)

Action	Sequence	Description
Processor B informs Processor A that data is in shared memory	5	Using an MU Processor B-side transmit register, the Processor B sends a packet information message to the Processor A to inform the Processor A of the arrival of new packet data that is stored in shared memory. The message contains the command, location, and length of packet data information.
Processor A receives interrupt	6	The Processor A receives an interrupt (assuming its corresponding Processor A MU-side receive interrupt is enabled), and the pending processing task becomes active and processes packet data from memory.
Processor A reads data, writes data	7	The Processor A reads or processes packet data from shared memory.
	8	The Processor A writes the result from packet processing to a separate buffer.
Processor A informs Processor B that transfer is finished	9	After the processing of the packet data finishes, the Processor A informs the Processor B (using the MU Processor A-side transmit register, ATRn).
Processor A sends interrupt to Processor B (request for more data)	10	The Processor B receives the next interrupt from the Processor A, in which the Processor A requests more packet data.

61.4.1.5 Free the Processor from Deadlock

During normal operations, one processor may conclude that the other processor is not working or is deadlocked. Using the SR register, the following methods are available for the processor to use to identify and correct the problem.

Table 543. How to Free the Processor A/B from Deadlock

Processor Mode	Technique	Description
-	Processor issues an interrupt	The other Processor can interrupt the Processor by issuing any one of the 64 (general purpose, receive, and transmit) interrupts
Normal	Processor issues a non-maskable interrupt	If the Processor is in Normal mode, the other Processor can issue an Non-Maskable Interrupt (NMI). The NMI routine can be made to execute a diagnostic routine that is predefined for such an event.
-	Processor issues a hardware reset	If the above is not helpful, the Processor can issue a hardware reset strobe to the other Processor.

61.4.2 Event Update Timing

Each processor's MU messaging side (Processor A or Processor B) has a hardware mechanism to send "event update requests" to the other processor's side. An "event" is considered when any information change should be reflected at the Status Register of the receiving processor. The event update latency is the delay between the event being ready at one processor and the resulting update at the Status Register of the other processor.

- The minimum event latency is “1 clock of the sending side” + “2 1/2 clocks of the receiving side”. The minimum case is if there is no event pending when the new event occurs.
- The maximum event latency is “6 clocks of the sending side” + “6 1/2 clocks of the receiving side.” The maximum case is if the event occurred just after a previous event was sent to the other side. The event update latency will vary between the above-mentioned minimum and maximum latencies, depending on the time at which the subsequent event is triggered.

61.4.3 Interrupts

The MU controls one processor interrupt requests to the other processor. This section describes all the interrupts that the module generates.

Below interrupt sources can be generated individually from the MU to the Processors.

- 16 receive interrupts (asserted when the Processors receive full bits are set and enabled in the xRCR register) for each of the receive registers
- 16 transmit interrupts (asserted when the Processor transmit empty bits are set and enabled in the xTCR register) for each of the transmit registers
- 32 general purpose interrupts (asserted when the GIP bits are set and enabled in the xGIER register)
- One non-maskable interrupt(asserted when the NMI bit is set in the CCR0 register)
- Core Reset Assertion interrupts(asserted when Processor goes into of Reset state and enabled in RAIE register bit)
- MU Software/Hardware Reset assertion(asserted when MUR and HR bit are written and enabled in MURIE and HRIE register bit)

All the interrupts except NMI are maskable in the Processor Control Register (xTCR, xRCR, xGIER, xCR and xCIER). The MU does not assume any internal priority of these interrupts. Multiple interrupts (for example, Receive 0 and Receive 1 interrupts or any of the transmit and general purpose interrupts) can be asserted at one time. The priority of these interrupts should be resolved by the interrupt controller at the chip level.

Triggering any enabled interrupt will "wake up" the Processor before servicing the interrupt.

The General Purpose Interrupt Pending GIPn bits should be cleared by the software (as part of the interrupt service routine) to de-assert the request to the interrupt controller.

The non-maskable interrupt should also be cleared by the software, by writing the NIMC bit in the CSSR0 register (as a part of the interrupt service routine) in order to de-assert the non-maskable interrupt request.

When a Processor writes to the General Interrupt bit (GIRn), the write event is synchronized to the other Processor to set the general interrupt request pending bit (GIPn). When the GIPn bit is set, and if the general purpose interrupt is enabled on the receiving Processor side (GIEn bit is set), then the transmitting Processor general purpose interrupt is issued to the receiving Processor. The receiving Processor clears this interrupt by writing a “1” on the GIPn bit. The interrupt is de-asserted as soon as the GIPn bit is written. The write event of the GIPn bit is synchronized to the other Processor. The synchronized signal clears the GIR bit. The software should not write the GIRn bit again until the GIRn bit is cleared.

Before setting a GIRn bit, you must verify that the GIRn bit is cleared, which means that a general interrupt is not pending. Generally, setting the GIRn bit while the bit has been set to “1” will be ignored, but in some cases it may issue a second interrupt.

61.4.4 Resets

The MU has below reset sources, and each reset has a different function from the MU or system perspective.

- Asynchronous system reset:
  - The asynchronous system reset on one MU-side returns all registers on this MU side to their default state.
  - When the asynchronous system reset on any MU-side is asserted, the MURS bit in CR register is set to "1" until the asynchronous system reset ends. Check MURS bit is cleared before starting any access to MU.
- MU software reset:

— Writing the CR[MUR] bit to "1" causes most of control and status registers to return to their default values and all internal states to be cleared.

— The instruction immediately following assertion of the MUR bit should not write to MU registers. Such a write may be overwritten by the reset sequence and the register will remain with the reset value. You should monitor the value of the SR[MURS] bit to know when the reset sequence on both processors has ended. After the reset sequence on both processors has ended, you can attemp a write to MU registers.

NOTE

MUR bit assertion is a delicate operation because it affects the other side's registers asynchronously. MUR bit assertion may cause unpredictable behavior if, for example, the other Processor is concurrently testing an MU register bit (TE bit in the other Processor TSR register). Before asserting the MUR bit, you should verify that the other Processor is not presently engaged in an MU signalling activity.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 62
On-Chip One Time Programmable (OCOTP) Controller

62.1 Introduction

The OCOTP controller is the primary interface with the on-chip fuses. You can use these fuses for several purposes, including:

- Unique chip identifiers
- Mask revision numbers
- Cryptographic keys
- Boot configurations
- Various control signals requiring permanent non-volatility

In this document, the terms "eFuse" and "OTP" (one-time programmable) are interchangeable. OCOTP refers to the hardware block itself.

62.2 Features

The OCOTP controller provides the following features:

- Shadow cache of eFuse loaded at reset prior to system boot
- Ability to read and override fuse values in shadow cache (does not affect fuse element)
- Ability to read fuses directly (ignoring shadow cache)
- Ability to write (program) fuses by the software
- Using eFuses to enable or disable program protection and read protection
- Using eFuses to enable or disable the overriding and read protection of shadow registers
- CRC (Polynomial CRC-32/MPEG-2) test for read-lock fuse content
- Redundancy and ECC protection of each fuse

62.3 Block diagram

The figure below shows the OCOTP system-level diagram.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-590d3ab624c1

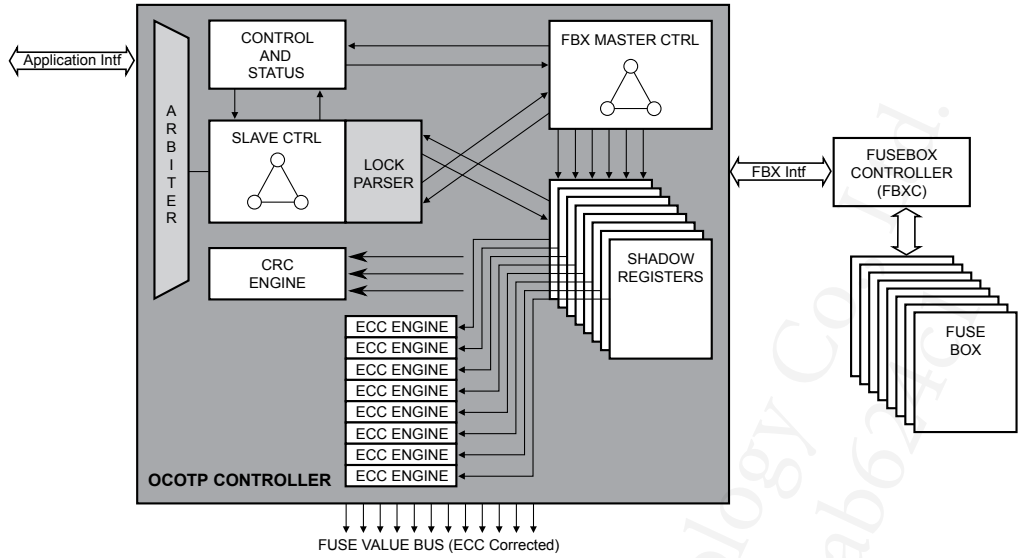


Figure 549. OCOTP system-level diagram

62.4 Operation

The OCOTP provides the following functions:

- Programming and reading of 32-bit eFuse word via the register interface
- ECC-corrected read of shadow registers
- Override of shadow register's value without programming of eFuses
- Calculation of CRC (Polynomial CRC-32/MPEG-2) on eFuse words
- 32-bit read and write accesses to eFuse and shadow registers

An eFuse word comprises 32 bits. Each bit in an eFuse is 0 by default.

The OCOTP reads all 32 bits of an eFuse at a time.

The OCOTP programs an eFuse one bit at a time. After the OCOTP programs an eFuse bit to 1, that bit permanently acquires a value of 1.

The OCOTP includes several groups of registers with slightly-different names, but identical functions. For brevity, the rest of this chapter uses the suffix _x to notate these groups. So, for example, CTRL_x refers to CTRL_SYS .

62.4.1 Shadow register reload

The OCOTP shadows the value of all eFuse words. Therefore, eFuse information is available through memory-mapped shadow registers. If you subsequently program the fuses, you must reload the shadow registers to keep them coherent with the eFuses.

The following operations reload the shadow registers:

- After the MC_RGM triggers a destructive reset

62.4.2 Fuse values

After the chip deasserts its reset, the OCOTP reads and stores all eFuse values in its shadow registers.

You can read eFuse values through shadow registers or by initiating a read sequence on the fuse location, if not read-protected.

All shadow register read accesses are single-error correction and double-error detection (SECDED) protected, if ECC check is enabled.

You can write or override the shadow register values depending on the protection that the corresponding fuse locks enable for those values.

62.4.3 Fuse protection and overrides

Typically the first few words of the eFuses define the eFuse word locks. These locks define the property of the eFuses they are associated with.

The OCOTP allows you to restrict access to eFuse words and shadow registers. You do this by programming the corresponding eFuse's lock bit, if applicable. See the attached chip fuse-map spreadsheet for details.

The type of locks and its effect on the associated fuse word are defined in the following table.

NOTE

eFuse word can have any number of these locks, and can also have no locks as mentioned in the chip fuse map.

NOTE

eFuse/shadow protection is chip-specific implementation dependent. See the chip fuse map for the protection specific to this chip.

Table 544. eFuse/shadow protection

Protection	Access	eFuse-word	Shadow register
Read Protect (RP)	Read	Prohibited *sets STATUS_x[ERROR]*	Prohibited *Reads back 0xBADABADA*
	Write	Allowed	Allowed
Write Protect (WP)	Read	Allowed	Allowed
	Write	Prohibited *sets STATUS_x[ERROR]*	Allowed
Override Protect (OP)	Read	Allowed	Allowed
	Write	Allowed	Prohibited *Will generate transaction error on the peripheral bus*

62.4.4 eFuse and shadow register read

All shadow registers are always readable except when read-protected. The read-protect lock corresponding to a eFuse word also applies read protection to the corresponding shadow register. Any attempt to read from a read-protected eFuse word returns 0xBADABADA. Before initiating any new access of fuse word or shadow register, you must write a 0 to STATUS_x[ERROR].

The shadow registers are memory-mapped. See Shadow (SHADOWS0 - SHADOWS127) for details.

To read a fuse word directly and correctly from the fusebox, complete the following steps:

1. Check that STATUS_x[BUSY] and STATUS_x[ERROR] are 0. You must do this because the OCOTP does not support overlapping access requests. It must complete any pending write, or read request before you can request a new read access.
2. Write the eFuse address to ADDR_x[ADDR].
3. Write the authorization code into CTRL_x[AUTH_KEY]. The OCOTP reads the data and transfers it to RDATA_x[DATA].
4. Write 0001b to CTRL_x[RD_WR]

5. Wait for the read request to finish. If the request finishes successfully, the OCOTP writes a 0 to STATUS_x[BUSY]. If the request was for a read-protected eFuse word or other master region, the OCOTP writes a 1 to STATUS_x[ERROR] and does not write anything to STATUS_x[BUSY].
6. Read RDATA_x[DATA] to get the eFuse word value.

62.4.5 eFuse and shadow register writes

Your software can override shadow register bits unless the bits are protected against overrides.

The shadow registers are memory-mapped. See Shadow (SHADOWS0 - SHADOWS127) for details.

NOTE

You cannot write to the ECC shadow registers.

Here are recommended best practices for you to perform before you program the eFuses:

1. Program the shadow register with the intended eFuse value.
2. Examine how step 1 affected the chip's functionality. If the functionality is as expected, then proceed to the next step.
3. Program the eFuse with the value from step 1.
4. Program the eFuses lock so that eFuses are protected from any spurious writes. If there are other eFuse words with same lock, then the fuse lock should be programmed after the last word.
5. Program the calculated ECC in the ECC eFuse.
6. Program the ECC eFuse enable as 1.

CAUTION

You cannot change the ECC eFuses after performing step 6.

Perform these steps to program the fuse bank correctly::

1. Check that STATUS_*x*[BUSY] and STATUS_*x*[ERROR] are 0. You must do this because the OCOTP does not support overlapping access requests. It must complete any pending write,or read request before you can request a new read access.
2. Write the requested fuse address to ADDR_*x*[ADDR].
3. Write the data to WRDATA_*x*[DATA].
4. Write the following value to CTRL_x[RD_WR], depending on your need:
   - 0010b
5. Write the authorization code to CTRL_x[AUTH_KEY]. You must do this step to avoid rogue code performing erroneus writes to eFuses.

NOTE

Step 4 and 5 must be performed together. You must write to both the register fields on a single write.

After you complete these steps, the OCOTP checks whether your request is to a protected or locked region. If it is, the OCOTP does not perform any eFuse access and writes a 1 to STATUS_x[ERROR]. Your software must then clear this error before submitting any new write request. If the OCOTP determines that it can fulfill your request, it does the following:

1. The OCOTP initiates the write sequence.
2. The OCOTP writes a 1 to STATUS_x[BUSY].
3. The OCOTP programs the eFuse as described in Operation. During the programming, the OCOTP shifts WRDATA_*x*[DATA] to the right with zero fill. This shifting is required to program the eFuse serially.
4. During the write operation, the OCOTP prohibits you from modifying WRDATA_x[DATA] and ADDR_x[ADDR].

5. After the operation completes, the OCOTP writes a 0 to STATUS_x[BUSY].

NOTE

If the ECC also has a lock, you must first program the ECC and only then program the lock register. The ECC must be calculated with assumption that the lock bit has a value of 1.

CAUTION

eFuse write can occur anytime the chip operates within specified operating conditions. However, if eFuse write occurs when the chip's operating conditions are outside of specifications (for example, when losing power), or if something interrupted the eFuse write (for example, a transaction or a chip reset), the update to the eFuse is not guaranteed. The best-practice recommendation is for your software to read the eFuse value after any eFuse write operation to verify the update has occurred as expected.

62.4.6 eFuse powerdown

Your software can powerdown the eFuse by writing 1 to CTRL_FBX[FBX_PD].

Following are the steps that you must perform before powering down eFuses.

1. Write 1 to CTRL_FBX[IPG_STOP].
2. Wait until OCOTP writes 1 to CTRL_FBX[IPG_STOP_ACK].
3. Powerdown eFuse by writing 1 to CTRL_FBX[FBX_PD].

CAUTION

You must not powerdown the eFuses while they are being accessed. This can corrupt the eFuses.

62.4.7 Write postamble

Because of internal electrical characteristics of the eFuse during writes, all eFuse operations following a write must :

- Be separated by a minimum duration defined in the chip data sheet
- Occur after STATUS_x[BUSY] becomes 0

This guarantees that the on-chip programming voltages reach a steady state after the OCOTP finishes a write sequence. A recommended software sequence, to meet the least time required between two subsequent writes, is as follows:

1. Request the writing of data.
2. Wait until STATUS_x[BUSY] becomes 0.
3. Wait for the defined duration.
4. Perform the next eFuse operation.

62.5 Behavior during reset

The OCOTP is always active, post destructive reset. The shadow registers automatically load the appropriate OTP contents after he chip deasserts its reset. During this loading time, the OCOTP keeps STATUS_x[BUSY]=1.

Fuses can support a finite number of reload cycles. See the chip data sheet.

62.6 Power control of OTP memory

Your software is responsible for shutting off power to OTP memory in idle status to save power.

After system power-up, your software must write a 1 to CTRL_FBX[FBX_PD] to shut off power to the eFuses and FBXC if none of the masters requested a fuse operation.

After completing eFuse programming or reading, your software must write a 1 to CTRL_FBX[FBX_PD] to shut off power to eFuses if none of the masters requested a eFuse operation.

Your software must wake up the OTP after power-down by writing a 0 to CTRL_FBX[FBX_PD]. You must meet the OTP access constraint after wakeup. See the chip data sheet.

The OCOTP and its shadow registers are available even after CTRL_FBX[FBX_PD]=1.

62.7 CRC (Polynomial CRC-32/MPEG-2) test

You can use the CRC test feature to validate read-protected eFuse content, though it's not restricted to only read-protected eFuses. A few eFuses are dedicated to store the CRC value of pre-defined eFuse section. See the chip fuse map for details.

When a master executes CRC tests by writing a 1 to CTRL_*x*[CRC_TEST], the OCOTP:

- Automatically calculates the CRC value of the user-defined fuse region
- Compares it to the value in the specified CRC eFuse word

Here is the procedure of the CRC test flow:

1. Program the CRC value calculated by software code into one of the CRC fuse words.
2. Program the CRC test region's start address in CRC_RGN_*x*[STRT_ADDR].
3. Program the CRC test region's end address in CRC_RGN_*x*[END_ADDR].
4. Program the eFuse address of pre-programmed CRC value in CRC_ADDR_*x*[ADDR]. The OCOTP uses this address to compare the calculated CRC for the region defined in steps 2 and 3.
5. Write a 1 to CTRL_*x*[CRC_TEST] to start the CRC check.
6. Check that STATUS_*x*[BUSY] and STATUS_*x*[ERROR] are 0. You must do this because the OCOTP sets STATUS_*x*[BUSY] = 1 until the OCOTP calculates and compares CRC for the eFuses.
7. Check the STATUS_*x*[CRC_FAIL] value. if STATUS_*x*[CRC_FAIL]=0 then the CRC test passed. STATUS_*x*[CRC_FAIL] must be read in an independent read after the STATUS_*x*[BUSY] is 0.

62.8 Fuse ECC

Some of the eFuses are protected with SECDED ECC. See the chip fuse map for details. The OCOTP applies ECC protection to the shadow registers's value.

The OCOTP implements 7-bit ECC for every 32-bit ECC-protected eFuse word.

The eight bit after every 7-bit ECC is the ECC efuse enable.

See Figure 550 for the ECC equation and Table 545 for the terms used in that equation. In Figure 550, the ^ denotes an XOR operation.

$$ECC[i] = \hat{} \left(\text{Fuse Word}[31:0] \,\&\, G[i] \right)$$

Figure 550. Equation for ECC

Table 545. G[i] values

i	G[i]
0	C148_40FFh
1	2124_FF90h
2	6CFF_0808h
3	FF01_A444h
4	16F0_92A6h

Table continues on the next page...

Table 545. G[i] values (continued)

i	G[i]
5	101F_7161h
6	8A82_0F1Bh

To write ECC in eFuses, follow the programming steps in eFuse and shadow register writes.

See the chip fuse map for the mapping of ECC eFuses to functional eFuses.

Figure 551 shows the circuit of the ECC calculation in OCOTP.

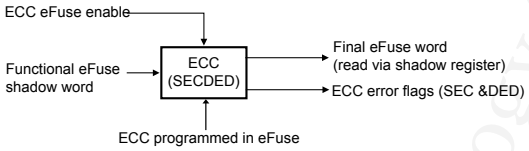


Figure 551. Shadow register ECC scheme

62.9 Memory maps/register definitions

The OCOTP register address space is assigned as per the following table:

Table 546. OCOTP memory space assignment

Memory map (Offset)	Access region
0h–FFh	Access to OCOTP registers (0h–FFh)
200h–3FCh	Access to OCOTP shadow registers for 4K eFuses[1]

1. See the attached chip fuse-map spreadsheet for details.

62.9.1 OCOTP register descriptions

62.9.1.1 OCOTP memory map

OCOTP base address: 400A_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	System master's control (CTRL_SYS)	32	ROWO	0000_0000h
4h	System master's eFuse address for read or write operation (ADDR_SYS)	32	RW	0000_0000h
8h	System master's eFuse write data (WRDATA_SYS)	32	RW	0000_0000h
Ch	System master's eFuse read data (RDATA_SYS)	32	RO	0000_0000h
30h	FBXC control (CTRL_FBX)	32	RW	0000_0000h
34h	eFuse region for CRC validation (CRC_RGN_SYS)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
38h	Pre-calculated CRC eFuse address for comparision (CRC_ADDR_SYS)	32	RW	0000_0000h
3Ch	CRC calculated from the eFuses (CRC_VALUE_SYS)	32	RO	0000_0000h
40h - 44h	General purpose (GPR0 - GPR1)	32	RW	0000_0000h
50h	System master's access status (STATUS_SYS)	32	W1C	0000_0000h
5Ch	OCOTP design version (VERSION)	32	RO	0000_0000h
70h	ECC status for single-bit ECC error (SEC0)	32	RO	0000_0000h
74h	ECC status for single-bit ECC error (SEC1)	32	RO	0000_0000h
78h	ECC status for single-bit ECC error (SEC2)	32	RO	0000_0000h
7Ch	ECC status for single-bit ECC error (SEC3)	32	RO	0000_0000h
B0h	ECC status for double-bit ECC error (DED0)	32	RO	0000_0000h
B4h	ECC status for double-bit ECC error (DED1)	32	RO	0000_0000h
B8h	ECC status for double-bit ECC error (DED2)	32	RO	0000_0000h
BCh	ECC status for double-bit ECC error (DED3)	32	RO	0000_0000h
F0h	ECC error injector (ERR_INJCTR)	32	RW	0000_0000h
200h - 3FCh	Shadow (SHADOWS0 - SHADOWS127)	32	RW	0000_0000h

62.9.1.2 System master's control (CTRL_SYS)

Offset

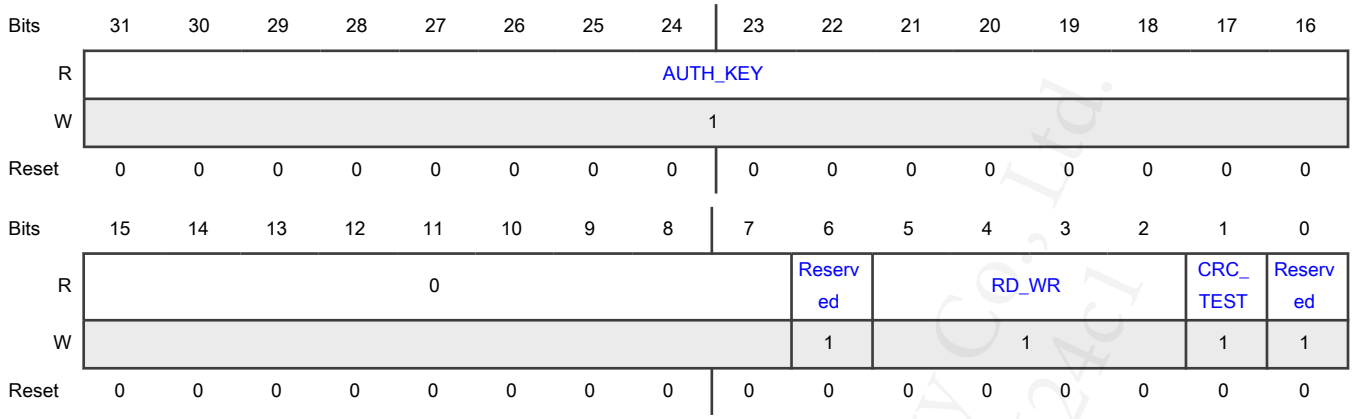
Register	Offset
CTRL_SYS	0h

Function

This register provides the necessary software interface for performing read and write operations to the eFuse. All the functions mentioned are mutually exclusive. You must perform only one operation at a time.

Provided under NXP COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-821-48a8-b251-29d6ca62d

Diagram



Fields

Field	Function
31-16 AUTH_KEY	Key to unlock read and write operation Must be written with CTRL_SYS[RD_WR]. The authorization key is 12h.
15-7 —	Reserved
6 —	Reserved
5-2 RD_WR	eFuse read and write control for the system master Read and write control for system master-controlled eFuses. See eFuse and shadow register read and eFuse and shadow register writes for more information. 0000, 0011, 0100, 0101, 0111, 1000, 1011, 1100, 1101, 1111b - Invalid combination, no action 0001b - Read eFuse 0010b - Write eFuse 0110b - Reserved 1001b - Reserved 1010b - Reserved 1110b - Reserved
1 CRC_TEST	CRC (Polynomial CRC-32/MPEG-2) test for system master Write a 1 to this field to calculate the CRC value according to the start address (CRC_RGN_SYS[START_ADDR]) and end address (CRC_RGN_SYS[END_ADDR]). Compare with CRC eFuse word according to CRC_ADDR_SYS[ADDR] to generate the STATUS_SYS[CRC_FAIL] flag. The OCOTP requires at least one cycle to update the status flags. 1b - CRC test enable for eFuses

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 —	Reserved

62.9.1.3 System master's eFuse address for read or write operation (ADDR_SYS)

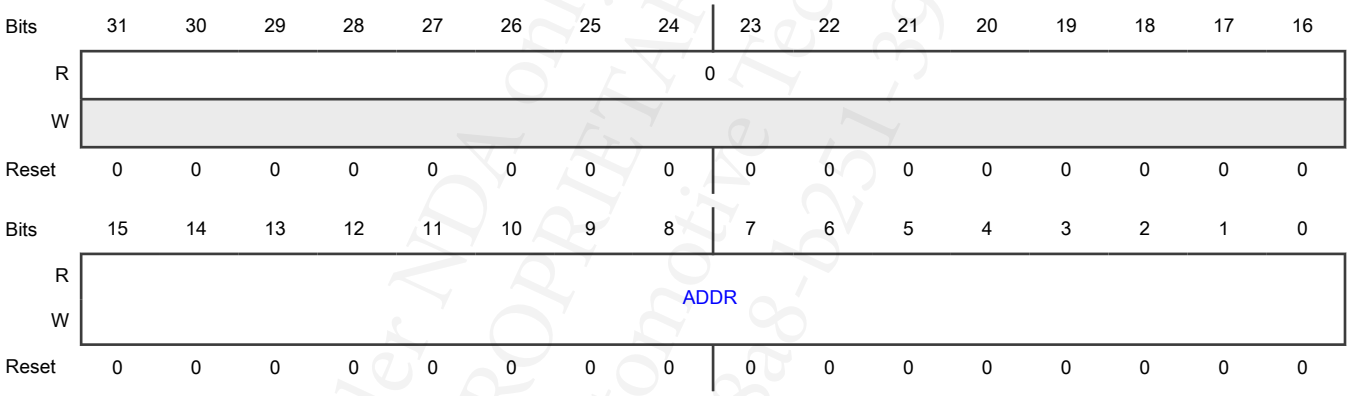
Offset

Register	Offset
ADDR_SYS	4h

Function

eFuse Address to be used in conjunction with CTRL_SYS

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 ADDR	eFuse read or write address If the eFuse address lies outside the master's region then the OCOTP generates a peripheral bus error. You must write to this field only when STATUS_SYS[BUSY]=0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e5-4821-4848-b251-590d3ab624c1

62.9.1.4 System master's eFuse write data (WRDATA_SYS)

Offset

Register	Offset
WRDATA_SYS	8h

Function

This register contains the write data to eFuse, to be used in conjunction with CTRL_SYS. You must write to this register only when STATUS_SYS[BUSY]=0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	DATA															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATA	eFuse data for eFuse write operation

62.9.1.5 System master's eFuse read data (RDATA_SYS)

Offset

Register	Offset
RDATA_SYS	Ch

Function

This register contains the data that the OCOTP read from the eFuse for the address specified in ADDR_SYS[ADDR].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c402e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DATA	Read data The read data from the eFuse for the address specified in ADDR_SYS[ADDR]

62.9.1.6 FBXC control (CTRL_FBX)

Offset

Register	Offset
CTRL_FBX	30h

Function

This register is accessible to all the masters and controls the FBXC directly. To avoid unintended operation, only one master must access this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													IPG_STO...	IPG_STOP	FBX_PD
W														W1C	1	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							SEC_MODE	0							Reserved
W																1
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4827-48a8-b251-39cd3ab624c1

Fields

Field	Function
31-19 —	Reserved
18 IPG_STOP_ACK	Low-power mode acknowledgment OCOTP acknowledges that no transaction is underway for eFuse, and OCOTP can be put into low-power mode. OCOTP then writes a 0 to CTRL_FBX[IPG_STOP].
17 IPG_STOP	Low Power entry Indicates that the Fusebox will be put in Low power mode.
16 FBX_PD	Fusebox Power Down After you write a 1 to this field, the FBXC drives the fusebox power-down signal to active-high state. If your chip is accessing the eFuse, the chip must not power down the fusebox during the access to avoid any data loss. See the chip data sheet for the required setup time (between power-down and fusebox access). You must respect this constraint. No accesses are possible to FBXC and eFuse during this time. You must ensure that you have used the CTRL_FBX[IPG_STOP] and CTRL_FBX[IPG_STOP_ACK] handshake mechanism to ensure all the pending transaction of FBXC are complete. See eFuse powerdown for the steps to powerdown eFuses. 1b - Enable fusebox power down mode
15-9 —	Reserved
8 SEC_MODE	Indicates that the FBXC is in Secure mode. If a write error occurs in this mode, OCOTP flags the error as an irrepairable error, even if repairs are available. 1b - FBXC is in secure mode
7-1 —	Reserved
0 —	Reserved

62.9.1.7 eFuse region for CRC validation (CRC_RGN_SYS)

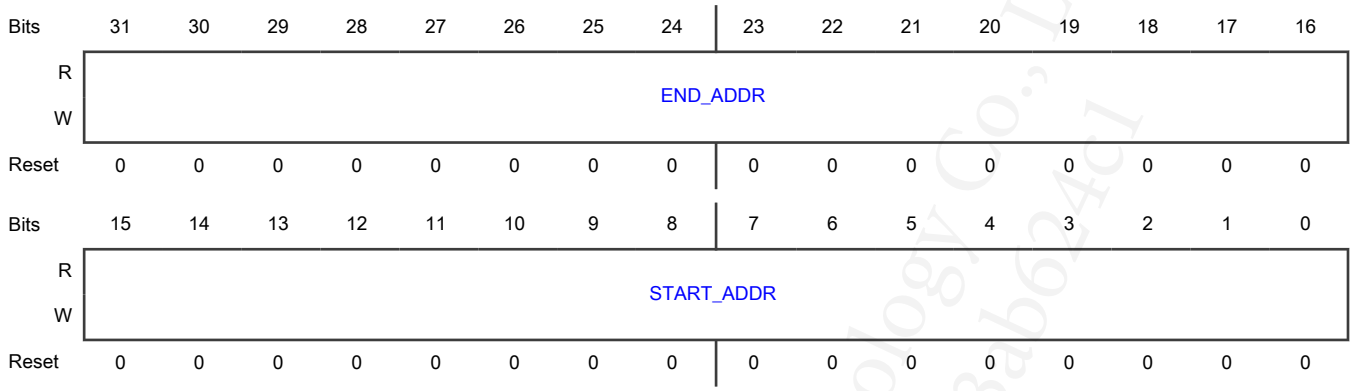
Offset

Register	Offset
CRC_RGN_SYS	34h

Function

Program the CRC (Polynomial CRC-32/MPEG-2) test region's start and end address in this register. A master must address only its own region. Otherwise, OCOTP generates a peripheral bus error. You must write to this register only when STATUS_SYS[BUSY]=0.

Diagram



Fields

Field	Function
31-16 END_ADDR	End eFuse address for CRC calculation
15-0 START_ADDR	Start eFuse address for CRC calculation

62.9.1.8 Pre-calculated CRC eFuse address for comparision (CRC_ADDR_SYS)

Offset

Register	Offset
CRC_ADDR_SYS	38h

Function

Pre-calculated CRC value's (Polynomial CRC-32/MPEG-2) eFuse address used for comparing with OCOTP computed CRC value. Used in conjunction with CTRL_SYS[CRC_TEST]. You must write to this register only when STATUS_SYS[BUSY]=0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 ADDR	CRC eFuse address used for comparison in CRC test

62.9.1.9 CRC calculated from the eFuses (CRC_VALUE_SYS)

Offset

Register	Offset
CRC_VALUE_SYS	3Ch

Function

CRC value (Polynomial CRC-32/MPEG-2) computed by the OCOTP.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CRC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CRC															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-0 CRC	CRC value The final 32-bit CRC signature calculated for eFuses within the CRC_RGN_SYS[START_ADDR] and CRC_RGN_SYS[END_ADDR].

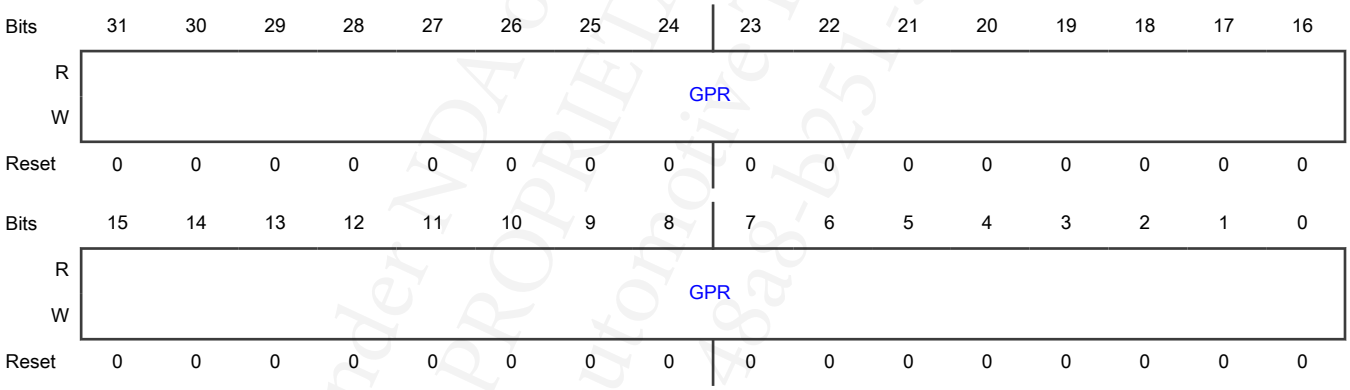
62.9.1.10 General purpose (GPR0 - GPR1)

Offset

Register	Offset
GPR0	40h
GPR1	44h

Function
General-purpose registers that you can use to store information for your application.

Diagram



Fields

Field	Function
31-0 GPR	General-purpose field that you can use to store information for your application

62.9.1.11 System master's access status (STATUS_SYS)

Offset

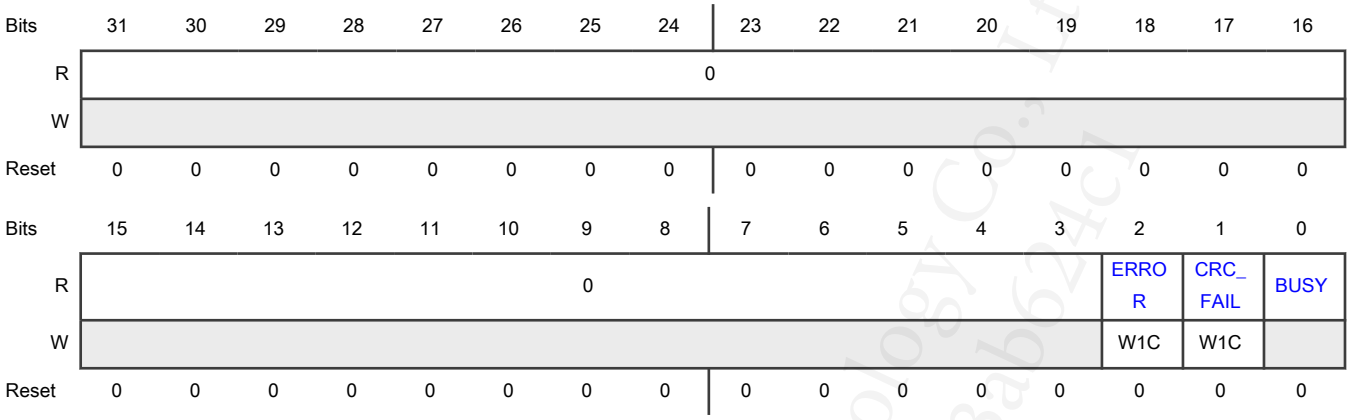
Register	Offset
STATUS_SYS	50h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8ab624c1

Function

This register provides you with OCOTP and eFuse status. The OCOTP updates this register two OCOTP clock cycles after you write to CTRL_SYS.

Diagram



Fields

Field	Function
31-3 —	Reserved
2 ERROR	Error status flag The OCOTP writes a 1 to this field whenever the OCOTP detects an illegal access. This occurs under the following circumstances: • You wrote an incorrect key to CTRL_SYS[AUTH_KEY] • You attempted to write to write-protected eFuse words • You attempted to read from read-protected eFuse words • The FBXC signaled a peripheral bus error during an access • The FBXC signaled a repair error after the OCOTP tried to program a fuse word • You attempted an access when the OCOTP already signaled a peripheral bus error You must clear this error by writing a 1 to this field before attempting any further access.
1 CRC_FAIL	CRC compare status Indicates that the OCOTP-computed CRC value did not match the CRC eFuse word at CRC_FUSE_ADDR_SYS[ADDR]. You must read CRC_FAIL only when STATUS_SYS[BUSY]=0.
0 BUSY	OCOTP busy indication The OCOTP writes a 1 to this field when any of the following eFuse accesses are in progress: • Write to eFuse • Read from eFuse

Table continues on the next page...

Table continued from the previous page...

Field	Function
	• CRC test You must not initiate any eFuse access when this field is 1. The OCOTP writes a 0 to this field after the OCOTP completes access.

62.9.1.12 OCOTP design version (VERSION)

Offset

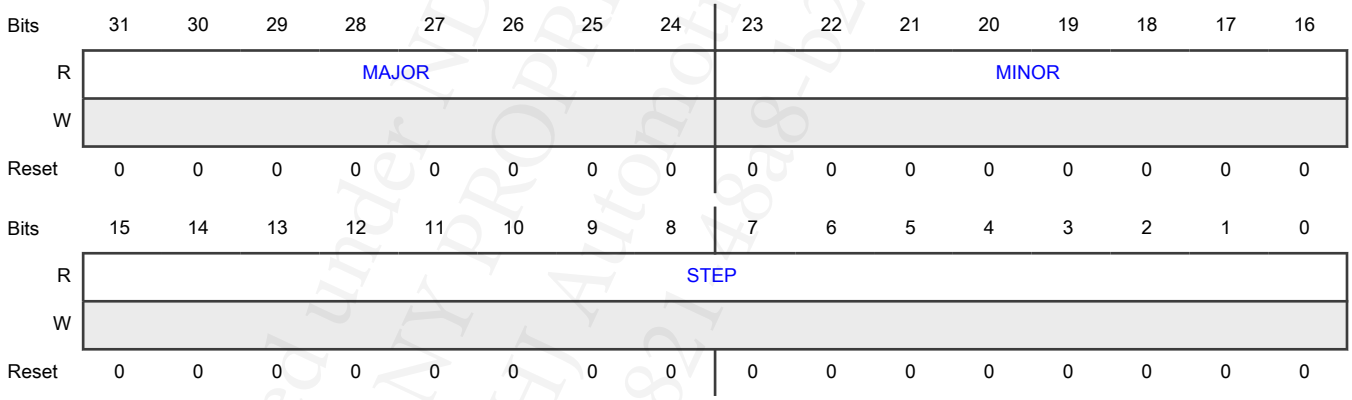
Register	Offset
VERSION	5Ch

Function

Indicates the OCOTP design version. This version consists of three parts:

- Major
- Minor
- Step

Diagram



Fields

Field	Function
31-24 MAJOR	The major part of the RTL version
23-16 MINOR	The minor part of the RTL version

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 STEP	The step part of the RTL version

62.9.1.13 ECC status for single-bit ECC error (SEC0)

Offset

Register	Offset
SEC0	70h

Function

Indicates that a single-bit ECC error occurred in the respective eFuse word and shadow register (n-1).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SEC32	SEC31	SEC30	SEC29	SEC28	SEC27	SEC26	SEC25	SEC24	SEC23	SEC22	SEC21	SEC20	SEC19	SEC18	SEC17
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SEC16	SEC15	SEC14	SEC13	SEC12	SEC11	SEC10	SEC9	SEC8	SEC7	SEC6	SEC5	SEC4	SEC3	SEC2	SEC1
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SECn	SEC error status Defines SEC error for the eFuse word n-1. 0b - No single-bit ECC error detected 1b - Single-bit ECC error detected

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462c4821-48a8-b251-90a3ab624c1

62.9.1.14 ECC status for single-bit ECC error (SEC1)

Offset

Register	Offset
SEC1	74h

Function

Indicates that a single-bit ECC error occurred in the respective eFuse word and shadow register (n-1).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SEC64	SEC63	SEC62	SEC61	SEC60	SEC59	SEC58	SEC57	SEC56	SEC55	SEC54	SEC53	SEC52	SEC51	SEC50	SEC49
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SEC48	SEC47	SEC46	SEC45	SEC44	SEC43	SEC42	SEC41	SEC40	SEC39	SEC38	SEC37	SEC36	SEC35	SEC34	SEC33
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SECn	SEC error status Defines SEC error for the eFuse word n-1. 0b - No single-bit ECC error detected 1b - Single-bit ECC error detected

62.9.1.15 ECC status for single-bit ECC error (SEC2)

Offset

Register	Offset
SEC2	78h

Function

Indicates that a single-bit ECC error occurred in the respective eFuse word and shadow register (n-1).

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-30d0da624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SEC96	SEC95	SEC94	SEC93	SEC92	SEC91	SEC90	SEC89	SEC88	SEC87	SEC86	SEC85	SEC84	SEC83	SEC82	SEC81
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SEC80	SEC79	SEC78	SEC77	SEC76	SEC75	SEC74	SEC73	SEC72	SEC71	SEC70	SEC69	SEC68	SEC67	SEC66	SEC65
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SECn	SEC error status Defines SEC error for the eFuse word n-1. 0b - No single-bit ECC error detected 1b - Single-bit ECC error detected

62.9.1.16 ECC status for single-bit ECC error (SEC3)

Offset

Register	Offset
SEC3	7Ch

Function

Indicates that a single-bit ECC error occurred in the respective eFuse word and shadow register (n-1).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CTU Automotive Technology Co., Ltd.
c8c4b2e5-4821-48a8-b251-390d8ab624c

Fields

Field	Function
31-0 —	Reserved

62.9.1.17 ECC status for double-bit ECC error (DED0)

Offset

Register	Offset
DED0	B0h

Function
Indicates a double ECC error in the respective eFuse word (n-1).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DED32	DED31	DED30	DED29	DED28	DED27	DED26	DED25	DED24	DED23	DED22	DED21	DED20	DED19	DED18	DED17
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DED16	DED15	DED14	DED13	DED12	DED11	DED10	DED9	DED8	DED7	DED6	DED5	DED4	DED3	DED2	DED1
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DEDn	DED error status Defines DED error for the eFuse word n-1. 0b - No double-bit ECC error detected 1b - Double-bit ECC error detected

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-4821-43a6-b254-90d3ab624c1

62.9.1.18 ECC status for double-bit ECC error (DED1)

Offset

Register	Offset
DED1	B4h

Function
Indicates a double ECC error in the respective eFuse word (n-1).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DED64	DED63	DED62	DED61	DED60	DED59	DED58	DED57	DED56	DED55	DED54	DED53	DED52	DED51	DED50	DED49
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DED48	DED47	DED46	DED45	DED44	DED43	DED42	DED41	DED40	DED39	DED38	DED37	DED36	DED35	DED34	DED33
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DEDn	DED error status Defines DED error for the eFuse word n-1. 0b - No double-bit ECC error detected 1b - Double-bit ECC error detected

62.9.1.19 ECC status for double-bit ECC error (DED2)

Offset

Register	Offset
DED2	B8h

Function
Indicates a double ECC error in the respective eFuse word (n-1).

Provided under NDA
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8462e5-4321-48a8-b251-50d8a624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DED96	DED95	DED94	DED93	DED92	DED91	DED90	DED89	DED88	DED87	DED86	DED85	DED84	DED83	DED82	DED81
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DED80	DED79	DED78	DED77	DED76	DED75	DED74	DED73	DED72	DED71	DED70	DED69	DED68	DED67	DED66	DED65
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DEDn	DED error status Defines DED error for the eFuse word n-1. 0b - No double-bit ECC error detected 1b - Double-bit ECC error detected

62.9.1.20 ECC status for double-bit ECC error (DED3)

Offset

Register	Offset
DED3	BCh

Function

Indicates a double ECC error in the respective eFuse word (n-1).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab6240

Fields

Field	Function
31-0 —	Reserved

62.9.1.21 ECC error injector (ERR_INJCTR)

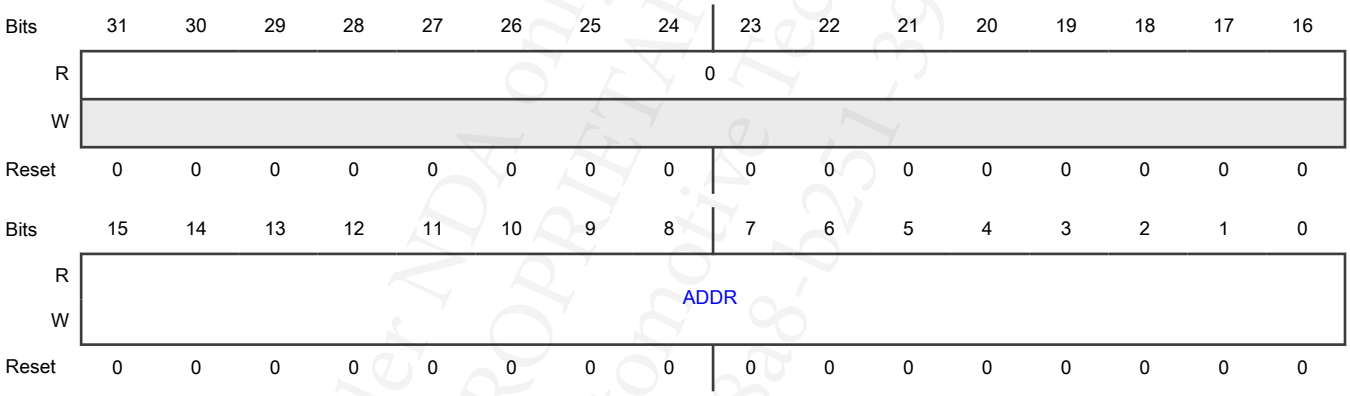
Offset

Register	Offset
ERR_INJCTR	F0h

Function

Injects an ECC error at a bit in the shadow register that you specify.

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 ADDR	The address of the bit in the shadow register where you want to inject the single-bit ECC error. The OCOTP inverts that bit. ADDR has the following structure: • Bits 15:5 represent the shadow register number minus one. Thus, ADDR[15:5]=0 represents shadow register 1, ADDR[15:5]=1 represents shadow register 2, and so on. • Bits 4:0 represent the shadow-register bit that you want to invert. Thus, ADDR[4:0]=0 represents bit 0 in the shadow register, ADDR[4:0]=1 represents bit 1 in the shadow register, and so on.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca42e2e-4821-4848-b251-590d3ab624c1

Table continued from the previous page...

Field	Function
	ECC error is not introduced when you attempt to inject error on a shadow register that has override protection. OCOTP generates SEC/DED based on the inverted value and on the ECC calculation for ECC-enabled fuse-words.

62.9.1.22 Shadow (SHADOWS0 - SHADOWS127)

Offset

For a = 0 to 127:

Register	Offset
SHADOWSa	200h + (a × 4h)

Function

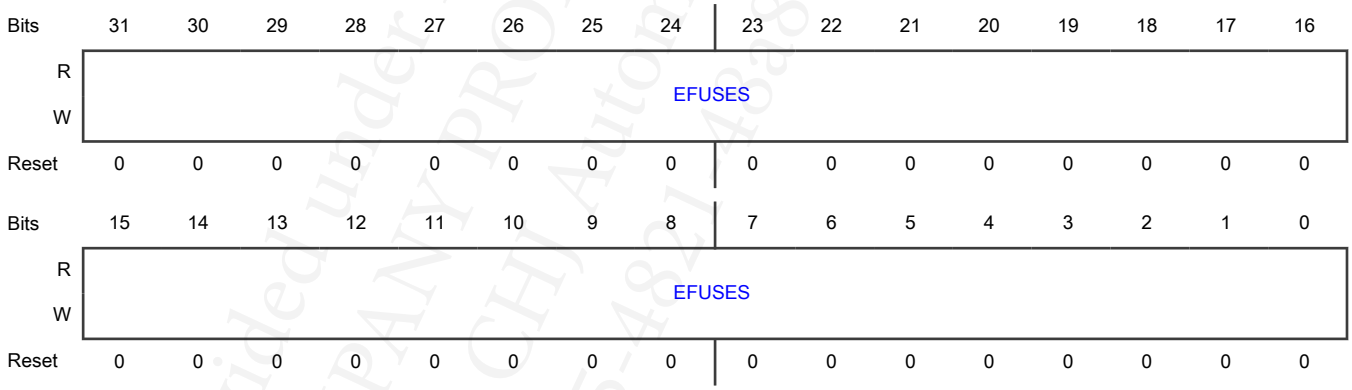
The accesses to shadow register is governed by the access lock as defined in the chip fuse map.

If you write 0000_0000h to this region, the OCOTP disallows it and generates a peripheral bus error. This safeguards against unintended writes.

NOTE

You cannot write to the ECC shadow registers.

Diagram



Fields

Field	Function
31-0 EFUSES	OTP data

Chapter 63
Fusebox Controller (FBXC)

63.1 Introduction

The FBXC (fusebox solution) is designed to serve all the applications that require electrically programmable fuses (eFuses). The eFuse controller can be accessed by any master.

The FBXC gasket is aimed to be placed between the eFuse block and a block with a peripheral interface. This way, it provides normal read/write access and controls the eFuse block access to the peripheral interface master block.

63.1.1 Overview

The FBXC gasket supports the peripheral interface and fusebox-specific control signals. The data sheet describes the third-party eFuse macros. The fusebox solution is illustrated in the following block diagram.

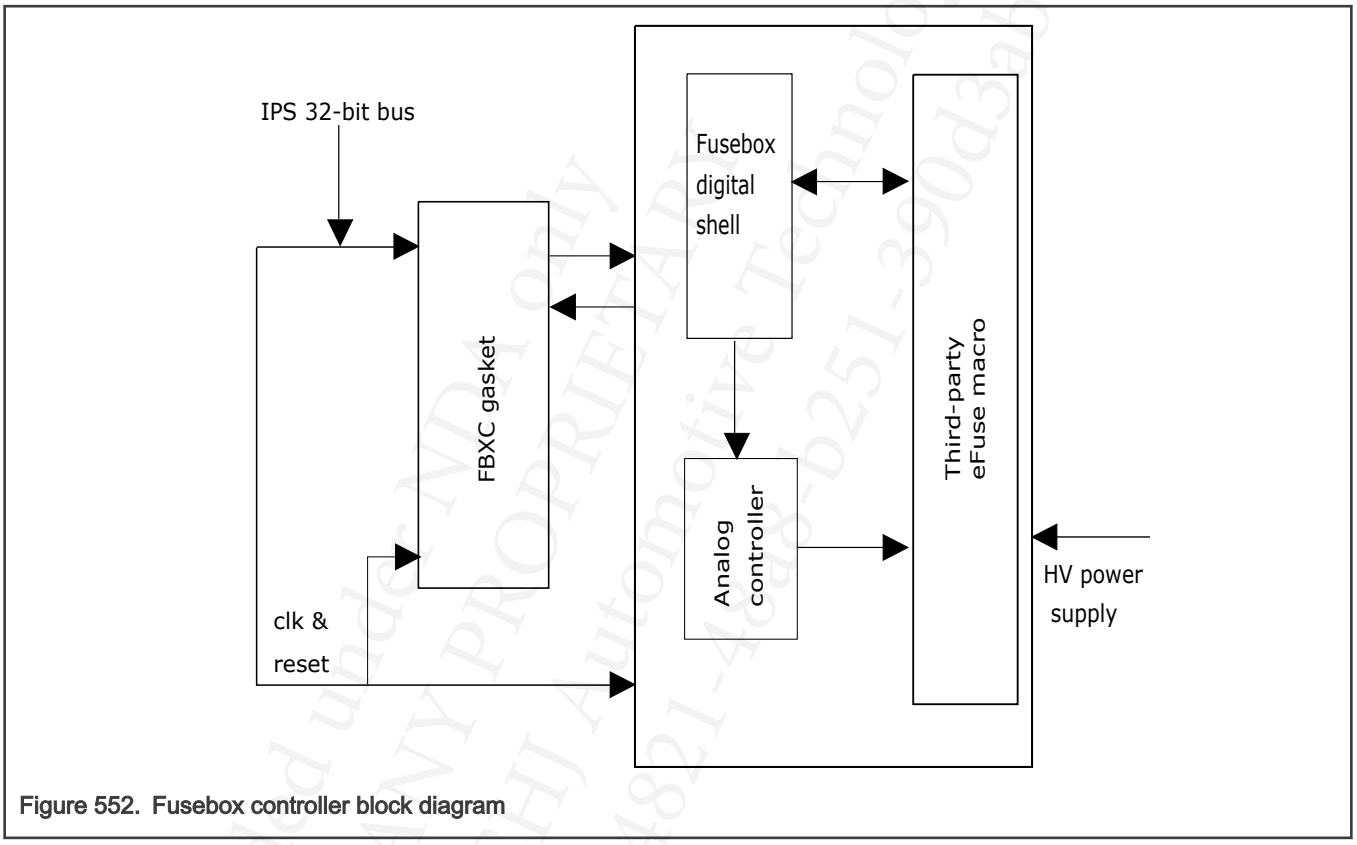


Figure 552. Fusebox controller block diagram

63.1.2 Features

This FBXC gasket includes the following features:

- Synchronous access to one-time electrically programmable fuses (eFuses) with a 32-bit-wide databus
- Synchronous access to all hard macro input/output ports
- Control signals to the eFuse block can be driven by the platform VDD-level control signals (unlike the third-party macros)
- Read / write control signals to the eFuse block that control all the analog eFuse block timing requirements
- Complete control or view of the eFuse block by control/status registers

Chapter 64
Safety

64.1 Introduction

This chip family is developed according to ISO 26262 and has an integrated safety concept targeting safety-related systems requiring high-safety integrity levels. The following documents support integration of this chip into safety-related systems:

- Reference Manual - Describes chip functionality
- Data Sheet - Describes chip operating conditions
- Safety Manual - Describes safety concept, possible safety mechanisms, and measures to reduce dependent failures
- Failure Modes Effects and Diagnostic Analysis (FMEDA) - Inductive analysis enabling customization of system-level safety mechanisms, including the resulting safety metrics for the given ISO standard
- FMEDA Report - Describes the FMEDA methodology and safety mechanisms supported in the FMEDA, including source of failure rates, failure modes, and assumptions made during the analysis
- The Dynamic FMEDA and FMEDA reports are available on request.

This device is a SafeAssure solution. For further information, see www.nxp.com/safeassure.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39013a1621c1

Chapter 65
Error Reporting Module (ERM)

65.1 Chip-specific ERM information

65.1.1 Instances and numbers of channels

The number of supported channels varies by ERM instance. The following table lists and describes the ERM instances and identifies the number of channels for each instance.

Table 547. ERM instances and numbers of channels

Instance	Reports errors for...	Number of used memory channels
ERM_PER	Memories for: • FlexRay and FlexCAN peripherals • QuadSPI RX and TX • SRAMC	9
ERM_eDMA0	eDMA0	1
ERM_eDMA1	eDMA1	1
ERM_CPU0	• Cortex-M7 core 0 DTCM • data tag RAM • data cache data RAM • instruction tag RAM • instruction cache data RAM	6
ERM_CPU1		6
ERM_CPU2		6

65.1.2 Memory error event sources and captured information

Each ERM instance channel *n* corresponds to a source of potential memory error events. For each instance channel, the following table shows:

- The corresponding memory
- What event information is captured

Table 548. ERM instance channel event sources and captured information

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
ERM PER	0	FlexRay_0 LRAM	No	Yes	Yes	402F8800	402F88AC	Absolute	No

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	1	FlexRay_0 DRAM	Yes	Yes	Yes	402F8000	402F807F	Absolute	No
	2	FlexCan_0	Yes	Yes	Yes	0	0000210C	32-bit aligned	No
	3	Flexcan_1	Yes	Yes	Yes	0	0000210C	32-bit aligned	No
	4	FlexCan_2	Yes	Yes	Yes	0	0000210C	32-bit aligned	No
	5	FlexCan_3	Yes	Yes	Yes	0	0000210C	32-bit aligned	No
	6	QuadSPI RX	Yes	No	No	NA	NA	—	No
	7	QuadSPI TX	Yes	No	No	NA	NA	—	No
	8	SRAMC0	Yes	No	Yes	0	—	Refer to PRAMECCA[24:0] of SRAMC for details.	Yes
	9	SRAMC1	Yes	No	Yes	0	—	Refer to PRAMECCA[24:0] of SRAMC for details.	Yes
	10	SRAMC2	Yes	No	Yes	0	—	Refer to PRAMECCA[24:0] of SRAMC for details.	Yes
	11	SRAMC3	Yes	No	Yes	0	—	Refer to PRAMECCA[24:0] of SRAMC for details.	Yes
ERM_eDMA_0	0	DMA0	Yes	Yes	Yes	00000000	000003F8	64-bit aligned	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
ERM_eDMA_1	0	DMA1	Yes	Yes	Yes	00000000	000003F8	64-bit aligned	Yes
ERM_CPU0	0	Cortex-M7 core 0 DTCM_0	Yes	Yes	Yes	00000000	0000FFF8	32-bit aligned	Yes
	1	Cortex-M7 core 0 DTCM_1	Yes	Yes	Yes	00000000	0000FFFC	32-bit aligned	Yes
	2	Cortex-M7 core 0 data cache data RAM	Yes	No	No	Not applicable	Not applicable	—	No
	3	Cortex-M7 core 0 data tag RAM	Yes	No	No	Not applicable	Not applicable	—	No
	4	Cortex-M7 core 0 instruction cache data RAM	Yes	No	No	Not applicable	Not applicable	—	No
	5	Cortex-M7 core 0 instruction tag RAM	Yes	No	No	Not applicable	Not applicable	—	No
ERM_CPU1	0	Cortex-M7 core 1 DTCM_0	Yes	Yes	Yes	00000000	0000FFF8	32-bit aligned	Yes
	1	Cortex-M7 core 1 DTCM_1	Yes	Yes	Yes	00000000	0000FFFC	32-bit aligned	Yes
	2	Cortex-M7 core 1 data cache data RAM	Yes	No	No	Not applicable	Not applicable	—	No
	3	Cortex-M7 core 1 data tag RAM	Yes	No	No	Not applicable	Not applicable	—	No

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	4	Cortex-M7 core 1 instruction cache data RAM	Yes	No	No	Not applicable	Not applicable	—	No
	5	Cortex-M7 core 1 instruction tag RAM	Yes	No	No	Not applicable	Not applicable	—	No
ERM_CPU2	0	Cortex-M7 core 2 DTCM_0	Yes	Yes	Yes	00000000	0000FFF8	32-bit aligned	Yes
	1	Cortex-M7 core 2 DTCM_1	Yes	Yes	Yes	00000000	0000FFFC	32-bit aligned	Yes
	2	Cortex-M7 core 2 data cache data RAM	Yes	No	No	Not applicable	Not applicable	—	No
	3	Cortex-M7 core 2 data tag RAM	Yes	No	No	Not applicable	Not applicable	—	No
	4	Cortex-M7 core 2 instruction cache data RAM	Yes	No	No	Not applicable	Not applicable	—	No
	5	Cortex-M7 core 2 instruction tag RAM	Yes	No	No	Not applicable	Not applicable	—	No
ERM_CPU3	0	Cortex-M7 core 3 DTCM_0	Yes	Yes	Yes	00000000	0000FFF8	32-bit aligned	Yes
	1	Cortex-M7 core 3 DTCM_1	Yes	Yes	Yes	00000000	0000FFFC	32-bit aligned	Yes

Table continues on the next page...

Preliminary Draft under NDA only
NXP COMPANY PROPRIETARY
Beijing CIT Automotive Technology Co., Ltd.
8c4b2e5-4821-48a8-b241-590d3a6624d1

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	2	Cortex-M7 core 3 data cache data RAM	Yes	No	No	Not applicable	Not applicable	—	No
	3	Cortex-M7 core 3 data tag RAM	Yes	No	No	Not applicable	Not applicable	—	No
	4	Cortex-M7 core 3 instruction cache data RAM	Yes	No	No	Not applicable	Not applicable	—	No
	5	Cortex-M7 core 3 instruction tag RAM	Yes	No	No	Not applicable	Not applicable	—	No
ERM_PFE0	0	egpix_rxf_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	egpix_rxf_rd1	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	egpix_rxf_rd2	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	egpix_txf_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	egpix_txf_rd1	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	egpix_txf_rd2	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	etgpix_rxf_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	etgpix_rxf_rd1	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	etgpix_rxf_rd2	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	hgpi1_txf_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	hgpi1_txf_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_PFE1	0	emac2_iq_class_entry	Yes	Yes	Yes	0h	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	1	emac2_iq_class_entry1	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	emac2_iq_class_entry2	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	emac2_iq_class_entry3	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	emac2_iq_lru_count	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	emac2_iq_lmem1_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	emac2_iq_lmem2_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	emac2_iq_lmem3_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	emac2_iq_lmem4_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	emac2_iq_lmem5_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	emac2_iq_lmem6_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	11	emac2_iq_lmem7_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	emac2_iq_lmem8_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_PFE2	0	pe0_iram	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	pe0_dram	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	pe0_dram1	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	pe0_dram2	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	pe0_dram3	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	pe0_dram4	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	pe0_dram5	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	pe0_dram6	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	pe0_dram7	Yes	Yes	Yes	0h	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	9	pe0_dram_lpe0	Yes	Yes	Yes	—	0h	Absolute	Yes
	10	pe0_dram_lpe01	Yes	Yes	Yes	0h	—	Absolute	Yes
	11	pe0_dram_lpe02	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	pe0_dram_lpe03	Yes	Yes	Yes	0h	—	Absolute	Yes
	13	pe0_dram_lpe1	Yes	Yes	Yes	0h	—	Absolute	Yes
	14	pe0_dram_lpe11	Yes	Yes	Yes	0h	—	Absolute	Yes
	15	pe0_dram_lpe12	Yes	Yes	Yes	0h	—	Absolute	Yes
	16	pe0_dram_lpe13	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_PFE3	0	pe1_iram	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	pe1_dram	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	pe1_dram1	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	pe1_dram2	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	pe1_dram3	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	pe1_dram4	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	pe1_dram5	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	pe1_dram6	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	pe1_dram7	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	pe1_dram_lpe0	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	pe1_dram_lpe01	Yes	Yes	Yes	0h	—	Absolute	Yes
	11	pe1_dram_lpe02	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	pe1_dram_lpe03	Yes	Yes	Yes	0h	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	13	pe1_dram_lpe1	Yes	Yes	Yes	0h	—	Absolute	Yes
	14	pe1_dram_lpe11	Yes	Yes	Yes	0h	—	Absolute	Yes
	15	pe1_dram_lpe12	Yes	Yes	Yes	0h	—	Absolute	Yes
	16	pe1_dram_lpe13	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_PFE4	0	pe2_iram	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	pe2_dram	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	pe2_dram1	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	pe2_dram2	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	pe2_dram3	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	pe2_dram4	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	pe2_dram5	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	pe2_dram6	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	pe2_dram7	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	pe2_dram_lpe0	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	pe2_dram_lpe01	Yes	Yes	Yes	0h	—	Absolute	Yes
	11	pe2_dram_lpe02	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	pe2_dram_lpe03	Yes	Yes	Yes	0h	—	Absolute	Yes
	13	pe2_dram_lpe1	Yes	Yes	Yes	0h	—	Absolute	Yes
	14	pe2_dram_lpe11	Yes	Yes	Yes	0h	—	Absolute	Yes
	15	pe2_dram_lpe12	Yes	Yes	Yes	0h	—	Absolute	Yes
	16	pe2_dram_lpe13	Yes	Yes	Yes	0h	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
ERM_PFE5	0	pe3_iram	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	pe3_dram	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	pe3_dram1	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	pe3_dram2	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	pe3_dram3	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	pe3_dram4	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	pe3_dram5	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	pe3_dram6	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	pe3_dram7	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	pe3_dram_lpe0	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	pe3_dram_lpe01	Yes	Yes	Yes	0h	—	Absolute	Yes
	11	pe3_dram_lpe02	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	pe3_dram_lpe03	Yes	Yes	Yes	0h	—	Absolute	Yes
	13	pe3_dram_lpe1	Yes	Yes	Yes	0h	—	Absolute	Yes
	14	pe3_dram_lpe11	Yes	Yes	Yes	0h	—	Absolute	Yes
	15	pe3_dram_lpe12	Yes	Yes	Yes	0h	—	Absolute	Yes
	16	pe3_dram_lpe13	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_PFE6	0	pe4_iram	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	pe4_dram	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	pe4_dram1	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	pe4_dram2	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	pe4_dram3	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	pe4_dram4	Yes	Yes	Yes	0h	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	6	pe4_dram5	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	pe4_dram6	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	pe4_dram7	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	pe4_dram_lpe0	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	pe4_dram_lpe01	Yes	Yes	Yes	0h	—	Absolute	Yes
	11	pe4_dram_lpe02	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	pe4_dram_lpe03	Yes	Yes	Yes	0h	—	Absolute	Yes
	13	pe4_dram_lpe1	Yes	Yes	Yes	0h	—	Absolute	Yes
	14	pe4_dram_lpe11	Yes	Yes	Yes	0h	—	Absolute	Yes
	15	pe4_dram_lpe12	Yes	Yes	Yes	0h	—	Absolute	Yes
	16	pe4_dram_lpe13	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_PFE7	0	pe5_iram	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	pe5_dram	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	pe5_dram1	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	pe5_dram2	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	pe5_dram3	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	pe5_dram4	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	pe5_dram5	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	pe5_dram6	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	pe5_dram7	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	pe5_dram_lpe0	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	pe5_dram_lpe01	Yes	Yes	Yes	0h	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	11	pe5_dram_lpe02	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	pe5_dram_lpe03	Yes	Yes	Yes	0h	—	Absolute	Yes
	13	pe5_dram_lpe1	Yes	Yes	Yes	0h	—	Absolute	Yes
	14	pe5_dram_lpe11	Yes	Yes	Yes	0h	—	Absolute	Yes
	15	pe5_dram_lpe12	Yes	Yes	Yes	0h	—	Absolute	Yes
	16	pe5_dram_lpe13	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_PFE8	0	pe6_iram	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	pe6_dram	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	pe6_dram1	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	pe6_dram2	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	pe6_dram3	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	pe6_dram4	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	pe6_dram5	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	pe6_dram6	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	pe6_dram7	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	pe6_dram_lpe0	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	pe6_dram_lpe01	Yes	Yes	Yes	0h	—	Absolute	Yes
	11	pe6_dram_lpe02	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	pe6_dram_lpe03	Yes	Yes	Yes	0h	—	Absolute	Yes
	13	pe6_dram_lpe1	Yes	Yes	Yes	0h	—	Absolute	Yes
	14	pe6_dram_lpe11	Yes	Yes	Yes	0h	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	15	pe6_dram_lpe12	Yes	Yes	Yes	0h	—	Absolute	Yes
	16	pe6_dram_lpe13	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_PFE9	0	pe7_iram	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	pe7_dram	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	pe7_dram1	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	pe7_dram2	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	pe7_dram3	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	pe7_dram4	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	pe7_dram5	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	pe7_dram6	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	pe7_dram7	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	pe7_dram_lpe0	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	pe7_dram_lpe01	Yes	Yes	Yes	0h	—	Absolute	Yes
	11	pe7_dram_lpe02	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	pe7_dram_lpe03	Yes	Yes	Yes	0h	—	Absolute	Yes
	13	pe7_dram_lpe1	Yes	Yes	Yes	0h	—	Absolute	Yes
	14	pe7_dram_lpe11	Yes	Yes	Yes	0h	—	Absolute	Yes
	15	pe7_dram_lpe12	Yes	Yes	Yes	0h	—	Absolute	Yes
	16	pe7_dram_lpe13	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_PFE10	0	pe_lmem	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	pe_lmem1	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	pe_lmem2	Yes	Yes	Yes	0h	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	3	pe_lmem3	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	mac_hash	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	mac_hash1	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	vlan_hash	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	vlan_hash1	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	smem2arb_mem_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	smem2arb_mem_rd1	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	ro_buf	Yes	Yes	Yes	0h	—	Absolute	Yes
	11	lmem	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	class_inq_mem	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_PFE11	0	tlite_cntx	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	tlite_cntx1	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	tlite_cntx2	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	tlite_cntx3	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	tlite_cntx4	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	tlite_cntx5	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	teq_qfifo	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	teq_qfifo1	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	teq_qfifo2	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	teq_qfifo3	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	teq_qfifo4	Yes	Yes	Yes	0h	—	Absolute	Yes
	11	teq_qfifo5	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	teq_qfifo6	Yes	Yes	Yes	0h	—	Absolute	Yes
	13	teq_qfifo7	Yes	Yes	Yes	0h	—	Absolute	Yes
	14	teq_qfifo8	Yes	Yes	Yes	0h	—	Absolute	Yes
	15	teq_qfifo9	Yes	Yes	Yes	0h	—	Absolute	Yes
	16	teq_qfifo10	Yes	Yes	Yes	0h	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	17	teq_qfifo11	Yes	Yes	Yes	0h	—	Absolute	Yes
	18	reclaim_buf_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	19	bmu2	Yes	Yes	Yes	0h	—	Absolute	Yes
	20	bcnt2	Yes	Yes	Yes	0h	—	Absolute	Yes
	21	tlite_inq_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	22	bmu	Yes	Yes	Yes	0h	—	Absolute	Yes
	23	bcnt	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_PFE12	0	hif1_dma_dxr_drx_ctrl_word_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	hif1_dma_dxr_dtx_ctrl_word_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	hif1_rx_dma_bdp_wr_burst_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	hif1_tx_dma_bdp_wr_burst_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	hif1_dma_dxr1_drx_rd	Yes	Yes	Yes	0h	Absolute	—	Yes
	5	hif1_dma_dxr1_dtx_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	hif1_dma_dxr2_drx_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	hif1_dma_dxr2_dtx_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	hif1_dma_dxr3_drx_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	hif1_dma_dxr3_dtx_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	hif1_dma_dxr4_drx_rd	Yes	Yes	Yes	0h	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	11	hif1_dma_dxr4_dtx_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	hif1_rx_dma_bdp_rd_burst_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	13	hif1_tx_dma_bdp_rd_burst_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	14	hif1_tx_bd_ctrl_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	15	hif1_tx_bd_ctrl_rd1	Yes	Yes	Yes	0h	—	Absolute	Yes
	16	hif1_tx_bd_ctrl_rd2	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_PFE13	0	upe_dram_lpe1	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	upe_dram_lpe11	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	upe_dram_lpe12	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	upe_dram_lpe13	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	upe_iram	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	upe_dram	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	upe_dram1	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	upe_dram2	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	upe_dram3	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	upe_dram4	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	upe_dram5	Yes	Yes	Yes	0h	—	Absolute	Yes
	11	upe_dram6	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	upe_dram7	Yes	Yes	Yes	0h	—	Absolute	Yes
	13	upe_dram_lpe0	Yes	Yes	Yes	0h	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	14	upe_dram_lpe01	Yes	Yes	Yes	0h	—	Absolute	Yes
	15	upe_dram_lpe02	Yes	Yes	Yes	0h	—	Absolute	Yes
	16	upe_dram_lpe03	Yes	Yes	Yes	0h	—	Absolute	Yes
	17	upe_inq_mem	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_PFE14	0	emac0_iq_class_entry	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	emac0_iq_class_entry1	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	emac0_iq_class_entry2	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	emac0_iq_class_entry3	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	emac0_iq_lru_count	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	emac0_iq_lmem1_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	emac0_iq_lmem2_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	emac0_iq_lmem3_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	emac0_iq_lmem4_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	emac0_iq_lmem5_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	emac0_iq_lmem6_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	11	emac0_iq_lmem7_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	emac0_iq_lmem8_rd	Yes	Yes	Yes	0h	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
ERM_PFE15	0	emac1_iq_class_entry	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	emac1_iq_class_entry1	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	emac1_iq_class_entry2	Yes	Yes	Yes	0h	—	Absolute	Yes
	3	emac1_iq_class_entry3	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	emac1_iq_lru_count	Yes	Yes	Yes	0h	—	Absolute	Yes
	5	emac1_iq_lmem1_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	emac1_iq_lmem2_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	emac1_iq_lmem3_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	emac1_iq_lmem4_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	9	emac1_iq_lmem5_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	10	emac1_iq_lmem6_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	11	emac1_iq_lmem7_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
	12	emac1_iq_lmem8_rd	Yes	Yes	Yes	0h	—	Absolute	Yes
ERM_LLCE	0	Data RAM_A	Yes	Yes	Yes	0h	—	Absolute	Yes
	1	Data RAM_B	Yes	Yes	Yes	0h	—	Absolute	Yes
	2	Reserved	NA	NA	NA	NA	NA	NA	NA
	3	FRPE RAM	Yes	Yes	Yes	0h	—	Absolute	Yes
	4	TxPPE RAM	Yes	Yes	Yes	0h	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	5	RxPPE RAM	Yes	Yes	Yes	0h	—	Absolute	Yes
	6	DTE RAM	Yes	Yes	Yes	0h	—	Absolute	Yes
	7	RxLUT RAM	Yes	Yes	Yes	0h	—	Absolute	Yes
	8	Reserved	NA	NA	NA	NA	NA	NA	NA
ERM_Standby_RAM	0	Standby SRAM (SRAMC)	Yes	No	Yes	0h	—	Absolute	Yes
ACP_ERM	0	Data RAM_A	Yes	Yes	Yes	0x0	—	Absolute	Yes
	1	Data RAM_B	Yes	Yes	Yes	0x0	—	Absolute	Yes
	2	Reserved	NA	NA	NA	NA	—	NA	NA
	3	FRPE RAM	Yes	Yes	Yes	0x0	—	Absolute	Yes
	4	TxPPE RAM	Yes	Yes	Yes	0x0	—	Absolute	Yes
	5	RxPPE RAM	Yes	Yes	Yes	0x0	—	Absolute	Yes
	6	DTE RAM	Yes	Yes	Yes	0x0	—	Absolute	Yes
	7	RxLUT0 RAM0	Yes	Yes	Yes	0x0	—	Absolute	Yes
	8	RxLUT0 RAM1	Yes	Yes	Yes	0x0	—	Absolute	Yes
	9	RxLUT0 RAM2	Yes	Yes	Yes	0x0	—	Absolute	Yes
	10	RxLUT0 RAM3	Yes	Yes	Yes	0x0	—	Absolute	Yes
	11	RxLUT1 RAM0	Yes	Yes	Yes	0x0	—	Absolute	Yes
	12	RxLUT1 RAM1	Yes	Yes	Yes	0x0	—	Absolute	Yes
	13	RxLUT1 RAM2	Yes	Yes	Yes	0x0	—	Absolute	Yes

Table continues on the next page...

Table 548. ERM instance channel event sources and captured information (continued)

Instance	Channel	Memory event source	Single-bit errors	Multi-bit errors	Address details event				Syndrome(bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (Hex)	End address (Hex)	Address alignment	
	14	RxLUT1 RAM3	Yes	Yes	Yes	0x0	—	Absolute	Yes
	15	RxLUT1 RAM4	Yes	Yes	Yes	0x0	—	Absolute	Yes
	16	RxLUT1 RAM5	Yes	Yes	Yes	0x0	—	Absolute	Yes
	17	RxLUT1 RAM6	Yes	Yes	Yes	0x0	—	Absolute	Yes
	18	RxLUT1 RAM7	Yes	Yes	Yes	0x0	—	Absolute	Yes
	19	RxLUT1 RAM8	Yes	Yes	Yes	0x0	—	Absolute	Yes
	20	RxLUT1 RAM9	Yes	Yes	Yes	0x0	—	Absolute	Yes
	21	RxLUT1 RAM10	Yes	Yes	Yes	0x0	—	Absolute	Yes
	22	RxLUT1 RAM11	Yes	Yes	Yes	0x0	—	Absolute	Yes

NOTE

The non-correctable error interrupt of each of the ERM instance is covered in the respective FCCU NCF. They are not controlled by the interrupt controller. See the FCCU NCF sheet for details.

65.1.3 Over-reported single-bit ECC errors for QuadSPI accesses

The ECC errors are generated based on the number of words available in the Tx buffer. Since the QuadSPI module writes data on the flash, it prefetches the data from Tx buffer. Up to 8 words are prefetched.

65.2 ERM_PFE, ERM_LLCE, and ERM_Standby_RAM instances

65.2.1 Instances and numbers of channels

The number of supported channels varies by ERM instance. The following table lists and describes the ERM instances and identifies the number of channels for each instance.

Beijing Qualitative Technology Co., Ltd.
c8c46e54-2e2-48a8-b251-390d3ab624c1

Table 549. ERM instances and numbers of channels

Instance	Reports errors for	Number of channels
ERM_PFE0	egpix_rxf_rd, egpix_txf_rd, etgpix_rxf_rd,hgpi1_txf_rd, hgpi1_txf_rd	11
ERM_PFE1	emac2_iq_class_entry, emac2_iq_lru_count, emac2_iq_lmem*_rd	13
ERM_PFE2	pe0_iram, pe0_dram*	17
ERM_PFE3	pe1_iram, pe1_dram*	17
ERM_PFE4	pe2_iram, pe2_dram*	17
ERM_PFE5	pe3_iram, pe3_dram*	17
ERM_PFE6	pe4_iram, pe4_dram*	17
ERM_PFE7	pe5_iram, pe5_dram*	17
ERM_PFE8	pe6_iram, pe6_dram*	17
ERM_PFE9	pe7_iram, pe7_dram*	17
ERM_PFE10	pe_lmem*, mac_hash*, vlan_hash*, smem2arb_mem_rd*, ro_buf, lmem, class_inq_mem	13
ERM_PFE11	tlite_cntx*, teq_qfifo*, reclaim_buf_rd, bmu2, bcnt2, tlite_inq_rd, bmu, bcnt	24
ERM_PFE12	hif1*	17
ERM_PFE13	upe*	18
ERM_PFE14	emac0*	13
ERM_PFE15	emac1*	13
ERM_LLCE	LLCE Data RAM_A, LLCE Data RAM_B, LLCE FRPE RAM, LLCE TxPPE RAM, LLCE RxPPE RAM, LLCE DTE RAM	9
ERM_STDBY_SRAM	only used to count single bit errors from standby SRAMC	1

65.2.2 Memory error event sources and captured information

Each ERM instance's channel *n* corresponds to a source of potential memory error events. For each instance channel, the following table shows:

- The corresponding memory
- For the corresponding memory's error events, whether the ERM captures single-bit errors, multi-bit errors, the error event's address, event syndrome information, or a combination of this information.

NOTE

For ERM to work properly, follow the sequence below:

1. Initialize all memories inside PFE after power on.
2. Inject a software reset.

Table 550. ERM instance channel event sources and captured information

Instance	Instance channel *n*	Memory *n* event source	Single-bit errors	Multi-bit errors	Address details			Event syndrome (bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (hex)	Address alignment	
ERM_PFE0	0	egpix_rxf_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	1	egpix_rxf_rd1	Yes	Yes	Yes	0x0	Absolute	Yes
	2	egpix_rxf_rd2	Yes	Yes	Yes	0x0	Absolute	Yes
	3	egpix_txf_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	4	egpix_txf_rd1	Yes	Yes	Yes	0x0	Absolute	Yes
	5	egpix_txf_rd2	Yes	Yes	Yes	0x0	Absolute	Yes
	6	etgpix_rxf_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	7	etgpix_rxf_rd1	Yes	Yes	Yes	0x0	Absolute	Yes
	8	etgpix_rxf_rd2	Yes	Yes	Yes	0x0	Absolute	Yes
	9	hgpi1_txf_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	10	hgpi1_txf_rd	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE1	0	emac2_iq_class_entry	Yes	Yes	Yes	0x0	Absolute	Yes
	1	emac2_iq_class_entry1	Yes	Yes	Yes	0x0	Absolute	Yes
	2	emac2_iq_class_entry2	Yes	Yes	Yes	0x0	Absolute	Yes
	3	emac2_iq_class_entry3	Yes	Yes	Yes	0x0	Absolute	Yes
	4	emac2_iq_lru_count	Yes	Yes	Yes	0x0	Absolute	Yes
	5	emac2_iq_lmem1_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	6	emac2_iq_lmem2_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	7	emac2_iq_lmem3_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	8	emac2_iq_lmem4_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	9	emac2_iq_lmem5_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	10	emac2_iq_lmem6_rd	Yes	Yes	Yes	0x0	Absolute	Yes

Table continues on the next page...

Table 550. ERM instance channel event sources and captured information (continued)

Instance	Instance channel *n*	Memory *n* event source	Single-bit errors	Multi-bit errors	Address details			Event syndrome (bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (hex)	Address alignment	
	11	emac2_iq_lmem7_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	12	emac2_iq_lmem8_rd	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE2	0	pe0_iram	Yes	Yes	Yes	0x0	Absolute	Yes
	1	pe0_dram	Yes	Yes	Yes	0x0	Absolute	Yes
	2	pe0_dram1	Yes	Yes	Yes	0x0	Absolute	Yes
	3	pe0_dram2	Yes	Yes	Yes	0x0	Absolute	Yes
	4	pe0_dram3	Yes	Yes	Yes	0x0	Absolute	Yes
	5	pe0_dram4	Yes	Yes	Yes	0x0	Absolute	Yes
	6	pe0_dram5	Yes	Yes	Yes	0x0	Absolute	Yes
	7	pe0_dram6	Yes	Yes	Yes	0x0	Absolute	Yes
	8	pe0_dram7	Yes	Yes	Yes	0x0	Absolute	Yes
	9	pe0_dram_lpe0	Yes	Yes	Yes	0x0	Absolute	Yes
	10	pe0_dram_lpe01	Yes	Yes	Yes	0x0	Absolute	Yes
	11	pe0_dram_lpe02	Yes	Yes	Yes	0x0	Absolute	Yes
	12	pe0_dram_lpe03	Yes	Yes	Yes	0x0	Absolute	Yes
	13	pe0_dram_lpe1	Yes	Yes	Yes	0x0	Absolute	Yes
	14	pe0_dram_lpe11	Yes	Yes	Yes	0x0	Absolute	Yes
	15	pe0_dram_lpe12	Yes	Yes	Yes	0x0	Absolute	Yes
	16	pe0_dram_lpe13	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE3	0	pe1_iram	Yes	Yes	Yes	0x0	Absolute	Yes
	1	pe1_dram	Yes	Yes	Yes	0x0	Absolute	Yes
	2	pe1_dram1	Yes	Yes	Yes	0x0	Absolute	Yes
	3	pe1_dram2	Yes	Yes	Yes	0x0	Absolute	Yes

Table continues on the next page...

Table 550. ERM instance channel event sources and captured information (continued)

Instance	Instance channel *n*	Memory *n* event source	Single-bit errors	Multi-bit errors	Address details			Event syndrome (bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (hex)	Address alignment	
	4	pe1_dram3	Yes	Yes	Yes	0x0	Absolute	Yes
	5	pe1_dram4	Yes	Yes	Yes	0x0	Absolute	Yes
	6	pe1_dram5	Yes	Yes	Yes	0x0	Absolute	Yes
	7	pe1_dram6	Yes	Yes	Yes	0x0	Absolute	Yes
	8	pe1_dram7	Yes	Yes	Yes	0x0	Absolute	Yes
	9	pe1_dram_lpe0	Yes	Yes	Yes	0x0	Absolute	Yes
	10	pe1_dram_lpe01	Yes	Yes	Yes	0x0	Absolute	Yes
	11	pe1_dram_lpe02	Yes	Yes	Yes	0x0	Absolute	Yes
	12	pe1_dram_lpe03	Yes	Yes	Yes	0x0	Absolute	Yes
	13	pe1_dram_lpe1	Yes	Yes	Yes	0x0	Absolute	Yes
	14	pe1_dram_lpe11	Yes	Yes	Yes	0x0	Absolute	Yes
	15	pe1_dram_lpe12	Yes	Yes	Yes	0x0	Absolute	Yes
	16	pe1_dram_lpe13	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE4	0	pe2_iram	Yes	Yes	Yes	0x0	Absolute	Yes
	1	pe2_dram	Yes	Yes	Yes	0x0	Absolute	Yes
	2	pe2_dram1	Yes	Yes	Yes	0x0	Absolute	Yes
	3	pe2_dram2	Yes	Yes	Yes	0x0	Absolute	Yes
	4	pe2_dram3	Yes	Yes	Yes	0x0	Absolute	Yes
	5	pe2_dram4	Yes	Yes	Yes	0x0	Absolute	Yes
	6	pe2_dram5	Yes	Yes	Yes	0x0	Absolute	Yes
	7	pe2_dram6	Yes	Yes	Yes	0x0	Absolute	Yes
	8	pe2_dram7	Yes	Yes	Yes	0x0	Absolute	Yes
	9	pe2_dram_lpe0	Yes	Yes	Yes	0x0	Absolute	Yes

Table continues on the next page...

Table 550. ERM instance channel event sources and captured information (continued)

Instance	Instance channel *n*	Memory *n* event source	Single-bit errors	Multi-bit errors	Address details			Event syndrome (bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (hex)	Address alignment	
	10	pe2_dram_lpe01	Yes	Yes	Yes	0x0	Absolute	Yes
	11	pe2_dram_lpe02	Yes	Yes	Yes	0x0	Absolute	Yes
	12	pe2_dram_lpe03	Yes	Yes	Yes	0x0	Absolute	Yes
	13	pe2_dram_lpe1	Yes	Yes	Yes	0x0	Absolute	Yes
	14	pe2_dram_lpe11	Yes	Yes	Yes	0x0	Absolute	Yes
	15	pe2_dram_lpe12	Yes	Yes	Yes	0x0	Absolute	Yes
	16	pe2_dram_lpe13	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE5	0	pe3_iram	Yes	Yes	Yes	0x0	Absolute	Yes
	1	pe3_dram	Yes	Yes	Yes	0x0	Absolute	Yes
	2	pe3_dram1	Yes	Yes	Yes	0x0	Absolute	Yes
	3	pe3_dram2	Yes	Yes	Yes	0x0	Absolute	Yes
	4	pe3_dram3	Yes	Yes	Yes	0x0	Absolute	Yes
	5	pe3_dram4	Yes	Yes	Yes	0x0	Absolute	Yes
	6	pe3_dram5	Yes	Yes	Yes	0x0	Absolute	Yes
	7	pe3_dram6	Yes	Yes	Yes	0x0	Absolute	Yes
	8	pe3_dram7	Yes	Yes	Yes	0x0	Absolute	Yes
	9	pe3_dram_lpe0	Yes	Yes	Yes	0x0	Absolute	Yes
	10	pe3_dram_lpe01	Yes	Yes	Yes	0x0	Absolute	Yes
	11	pe3_dram_lpe02	Yes	Yes	Yes	0x0	Absolute	Yes
	12	pe3_dram_lpe03	Yes	Yes	Yes	0x0	Absolute	Yes
	13	pe3_dram_lpe1	Yes	Yes	Yes	0x0	Absolute	Yes

Table continues on the next page...

Table 550. ERM instance channel event sources and captured information (continued)

Instance	Instance channel *n*	Memory *n* event source	Single-bit errors	Multi-bit errors	Address details			Event syndrome (bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (hex)	Address alignment	
	14	pe3_dram_lpe11	Yes	Yes	Yes	0x0	Absolute	Yes
	15	pe3_dram_lpe12	Yes	Yes	Yes	0x0	Absolute	Yes
	16	pe3_dram_lpe13	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE6	0	pe4_iram	Yes	Yes	Yes	0x0	Absolute	Yes
	1	pe4_dram	Yes	Yes	Yes	0x0	Absolute	Yes
	2	pe4_dram1	Yes	Yes	Yes	0x0	Absolute	Yes
	3	pe4_dram2	Yes	Yes	Yes	0x0	Absolute	Yes
	4	pe4_dram3	Yes	Yes	Yes	0x0	Absolute	Yes
	5	pe4_dram4	Yes	Yes	Yes	0x0	Absolute	Yes
	6	pe4_dram5	Yes	Yes	Yes	0x0	Absolute	Yes
	7	pe4_dram6	Yes	Yes	Yes	0x0	Absolute	Yes
	8	pe4_dram7	Yes	Yes	Yes	0x0	Absolute	Yes
	9	pe4_dram_lpe0	Yes	Yes	Yes	0x0	Absolute	Yes
	10	pe4_dram_lpe01	Yes	Yes	Yes	0x0	Absolute	Yes
	11	pe4_dram_lpe02	Yes	Yes	Yes	0x0	Absolute	Yes
	12	pe4_dram_lpe03	Yes	Yes	Yes	0x0	Absolute	Yes
	13	pe4_dram_lpe1	Yes	Yes	Yes	0x0	Absolute	Yes
	14	pe4_dram_lpe11	Yes	Yes	Yes	0x0	Absolute	Yes
	15	pe4_dram_lpe12	Yes	Yes	Yes	0x0	Absolute	Yes
	16	pe4_dram_lpe13	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE7	0	pe5_iram	Yes	Yes	Yes	0x0	Absolute	Yes
	1	pe5_dram	Yes	Yes	Yes	0x0	Absolute	Yes
	2	pe5_dram1	Yes	Yes	Yes	0x0	Absolute	Yes

Table continues on the next page...

Table 550. ERM instance channel event sources and captured information (continued)

Instance	Instance channel *n*	Memory *n* event source	Single-bit errors	Multi-bit errors	Address details			Event syndrome (bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (hex)	Address alignment	
	3	pe5_dram2	Yes	Yes	Yes	0x0	Absolute	Yes
	4	pe5_dram3	Yes	Yes	Yes	0x0	Absolute	Yes
	5	pe5_dram4	Yes	Yes	Yes	0x0	Absolute	Yes
	6	pe5_dram5	Yes	Yes	Yes	0x0	Absolute	Yes
	7	pe5_dram6	Yes	Yes	Yes	0x0	Absolute	Yes
	8	pe5_dram7	Yes	Yes	Yes	0x0	Absolute	Yes
	9	pe5_dram_lpe0	Yes	Yes	Yes	0x0	Absolute	Yes
	10	pe5_dram_lpe01	Yes	Yes	Yes	0x0	Absolute	Yes
	11	pe5_dram_lpe02	Yes	Yes	Yes	0x0	Absolute	Yes
	12	pe5_dram_lpe03	Yes	Yes	Yes	0x0	Absolute	Yes
	13	pe5_dram_lpe1	Yes	Yes	Yes	0x0	Absolute	Yes
	14	pe5_dram_lpe11	Yes	Yes	Yes	0x0	Absolute	Yes
	15	pe5_dram_lpe12	Yes	Yes	Yes	0x0	Absolute	Yes
	16	pe5_dram_lpe13	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE8	0	pe6_iram	Yes	Yes	Yes	0x0	Absolute	Yes
	1	pe6_dram	Yes	Yes	Yes	0x0	Absolute	Yes
	2	pe6_dram1	Yes	Yes	Yes	0x0	Absolute	Yes
	3	pe6_dram2	Yes	Yes	Yes	0x0	Absolute	Yes
	4	pe6_dram3	Yes	Yes	Yes	0x0	Absolute	Yes
	5	pe6_dram4	Yes	Yes	Yes	0x0	Absolute	Yes
	6	pe6_dram5	Yes	Yes	Yes	0x0	Absolute	Yes
	7	pe6_dram6	Yes	Yes	Yes	0x0	Absolute	Yes
	8	pe6_dram7	Yes	Yes	Yes	0x0	Absolute	Yes
	9	pe6_dram_lpe0	Yes	Yes	Yes	0x0	Absolute	Yes

Table continues on the next page...

Table 550. ERM instance channel event sources and captured information (continued)

Instance	Instance channel *n*	Memory *n* event source	Single-bit errors	Multi-bit errors	Address details			Event syndrome (bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (hex)	Address alignment	
	10	pe6_dram_lpe01	Yes	Yes	Yes	0x0	Absolute	Yes
	11	pe6_dram_lpe02	Yes	Yes	Yes	0x0	Absolute	Yes
	12	pe6_dram_lpe03	Yes	Yes	Yes	0x0	Absolute	Yes
	13	pe6_dram_lpe1	Yes	Yes	Yes	0x0	Absolute	Yes
	14	pe6_dram_lpe11	Yes	Yes	Yes	0x0	Absolute	Yes
	15	pe6_dram_lpe12	Yes	Yes	Yes	0x0	Absolute	Yes
	16	pe6_dram_lpe13	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE9	0	pe7_iram	Yes	Yes	Yes	0x0	Absolute	Yes
	1	pe7_dram	Yes	Yes	Yes	0x0	Absolute	Yes
	2	pe7_dram1	Yes	Yes	Yes	0x0	Absolute	Yes
	3	pe7_dram2	Yes	Yes	Yes	0x0	Absolute	Yes
	4	pe7_dram3	Yes	Yes	Yes	0x0	Absolute	Yes
	5	pe7_dram4	Yes	Yes	Yes	0x0	Absolute	Yes
	6	pe7_dram5	Yes	Yes	Yes	0x0	Absolute	Yes
	7	pe7_dram6	Yes	Yes	Yes	0x0	Absolute	Yes
	8	pe7_dram7	Yes	Yes	Yes	0x0	Absolute	Yes
	9	pe7_dram_lpe0	Yes	Yes	Yes	0x0	Absolute	Yes
	10	pe7_dram_lpe01	Yes	Yes	Yes	0x0	Absolute	Yes
	11	pe7_dram_lpe02	Yes	Yes	Yes	0x0	Absolute	Yes
	12	pe7_dram_lpe03	Yes	Yes	Yes	0x0	Absolute	Yes
	13	pe7_dram_lpe1	Yes	Yes	Yes	0x0	Absolute	Yes

Table continues on the next page...

Table 550. ERM instance channel event sources and captured information (continued)

Instance	Instance channel *n*	Memory *n* event source	Single-bit errors	Multi-bit errors	Address details			Event syndrome (bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (hex)	Address alignment	
	14	pe7_dram_lpe11	Yes	Yes	Yes	0x0	Absolute	Yes
	15	pe7_dram_lpe12	Yes	Yes	Yes	0x0	Absolute	Yes
	16	pe7_dram_lpe13	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE10	0	pe_lmem	Yes	Yes	Yes	0x0	Absolute	Yes
	1	pe_lmem1	Yes	Yes	Yes	0x0	Absolute	Yes
	2	pe_lmem2	Yes	Yes	Yes	0x0	Absolute	Yes
	3	pe_lmem3	Yes	Yes	Yes	0x0	Absolute	Yes
	4	mac_hash	Yes	Yes	Yes	0x0	Absolute	Yes
	5	mac_hash1	Yes	Yes	Yes	0x0	Absolute	Yes
	6	vlan_hash	Yes	Yes	Yes	0x0	Absolute	Yes
	7	vlan_hash1	Yes	Yes	Yes	0x0	Absolute	Yes
	8	smem2arb_mem_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	9	smem2arb_mem_rd1	Yes	Yes	Yes	0x0	Absolute	Yes
	10	ro_buf	Yes	Yes	Yes	0x0	Absolute	Yes
	11	lmem	Yes	Yes	Yes	0x0	Absolute	Yes
	12	class_inq_mem	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE11	0	tlite_cntx	Yes	Yes	Yes	0x0	Absolute	Yes
	1	tlite_cntx1	Yes	Yes	Yes	0x0	Absolute	Yes
	2	tlite_cntx2	Yes	Yes	Yes	0x0	Absolute	Yes
	3	tlite_cntx3	Yes	Yes	Yes	0x0	Absolute	Yes
	4	tlite_cntx4	Yes	Yes	Yes	0x0	Absolute	Yes
	5	tlite_cntx5	Yes	Yes	Yes	0x0	Absolute	Yes
	6	teq_qfifo	Yes	Yes	Yes	0x0	Absolute	Yes
	7	teq_qfifo1	Yes	Yes	Yes	0x0	Absolute	Yes
	8	teq_qfifo2	Yes	Yes	Yes	0x0	Absolute	Yes
	9	teq_qfifo3	Yes	Yes	Yes	0x0	Absolute	Yes

Table continues on the next page...

Table 550. ERM instance channel event sources and captured information (continued)

Instance	Instance channel *n*	Memory *n* event source	Single-bit errors	Multi-bit errors	Address details			Event syndrome (bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (hex)	Address alignment	
	10	teq_qfifo4	Yes	Yes	Yes	0x0	Absolute	Yes
	11	teq_qfifo5	Yes	Yes	Yes	0x0	Absolute	Yes
	12	teq_qfifo6	Yes	Yes	Yes	0x0	Absolute	Yes
	13	teq_qfifo7	Yes	Yes	Yes	0x0	Absolute	Yes
	14	teq_qfifo8	Yes	Yes	Yes	0x0	Absolute	Yes
	15	teq_qfifo9	Yes	Yes	Yes	0x0	Absolute	Yes
	16	teq_qfifo10	Yes	Yes	Yes	0x0	Absolute	Yes
	17	teq_qfifo11	Yes	Yes	Yes	0x0	Absolute	Yes
	18	reclaim_buf_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	19	bmu2	Yes	Yes	Yes	0x0	Absolute	Yes
	20	bcnt2	Yes	Yes	Yes	0x0	Absolute	Yes
	21	tlite_inq_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	22	bmu	Yes	Yes	Yes	0x0	Absolute	Yes
	23	bcnt	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE12	0	hif1_dma_dxr_drx_ctrl_word_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	1	hif1_dma_dxr_dtx_ctrl_word_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	2	hif1_rx_dma_bdp_wr_burst_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	3	hif1_tx_dma_bdp_wr_burst_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	4	hif1_dma_dxr1_drx_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	5	hif1_dma_dxr1_dtx_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	6	hif1_dma_dxr2_drx_rd	Yes	Yes	Yes	0x0	Absolute	Yes

Table continues on the next page...

Table 550. ERM instance channel event sources and captured information (continued)

Instance	Instance channel *n*	Memory *n* event source	Single-bit errors	Multi-bit errors	Address details			Event syndrome (bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (hex)	Address alignment	
	7	hif1_dma_dxr2_dtx_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	8	hif1_dma_dxr3_drx_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	9	hif1_dma_dxr3_dtx_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	10	hif1_dma_dxr4_drx_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	11	hif1_dma_dxr4_dtx_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	12	hif1_rx_dma_bdp_rd_burst_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	13	hif1_tx_dma_bdp_rd_burst_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	14	hif1_tx_bd_ctrl_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	15	hif1_tx_bd_ctrl_rd1	Yes	Yes	Yes	0x0	Absolute	Yes
	16	hif1_tx_bd_ctrl_rd2	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE13	0	upe_dram_lpe1	Yes	Yes	Yes	0x0	Absolute	Yes
	1	upe_dram_lpe11	Yes	Yes	Yes	0x0	Absolute	Yes
	2	upe_dram_lpe12	Yes	Yes	Yes	0x0	Absolute	Yes
	3	upe_dram_lpe13	Yes	Yes	Yes	0x0	Absolute	Yes
	4	upe_iram	Yes	Yes	Yes	0x0	Absolute	Yes
	5	upe_dram	Yes	Yes	Yes	0x0	Absolute	Yes
	6	upe_dram1	Yes	Yes	Yes	0x0	Absolute	Yes
	7	upe_dram2	Yes	Yes	Yes	0x0	Absolute	Yes
	8	upe_dram3	Yes	Yes	Yes	0x0	Absolute	Yes

Table continues on the next page...

Table 550. ERM instance channel event sources and captured information (continued)

Instance	Instance channel *n*	Memory *n* event source	Single-bit errors	Multi-bit errors	Address details			Event syndrome (bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (hex)	Address alignment	
	9	upe_dram4	Yes	Yes	Yes	0x0	Absolute	Yes
	10	upe_dram5	Yes	Yes	Yes	0x0	Absolute	Yes
	11	upe_dram6	Yes	Yes	Yes	0x0	Absolute	Yes
	12	upe_dram7	Yes	Yes	Yes	0x0	Absolute	Yes
	13	upe_dram_lpe0	Yes	Yes	Yes	0x0	Absolute	Yes
	14	upe_dram_lpe01	Yes	Yes	Yes	0x0	Absolute	Yes
	15	upe_dram_lpe02	Yes	Yes	Yes	0x0	Absolute	Yes
	16	upe_dram_lpe03	Yes	Yes	Yes	0x0	Absolute	Yes
	17	upe_inq_mem	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE14	0	emac0_iq_class_entry	Yes	Yes	Yes	0x0	Absolute	Yes
	1	emac0_iq_class_entry1	Yes	Yes	Yes	0x0	Absolute	Yes
	2	emac0_iq_class_entry2	Yes	Yes	Yes	0x0	Absolute	Yes
	3	emac0_iq_class_entry3	Yes	Yes	Yes	0x0	Absolute	Yes
	4	emac0_iq_lru_count	Yes	Yes	Yes	0x0	Absolute	Yes
	5	emac0_iq_lmem1_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	6	emac0_iq_lmem2_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	7	emac0_iq_lmem3_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	8	emac0_iq_lmem4_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	9	emac0_iq_lmem5_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	10	emac0_iq_lmem6_rd	Yes	Yes	Yes	0x0	Absolute	Yes

Table continues on the next page...

Table 550. ERM instance channel event sources and captured information (continued)

Instance	Instance channel *n*	Memory *n* event source	Single-bit errors	Multi-bit errors	Address details			Event syndrome (bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (hex)	Address alignment	
	11	emac0_iq_lmem7_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	12	emac0_iq_lmem8_rd	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_PFE15	0	emac1_iq_class_entry	Yes	Yes	Yes	0x0	Absolute	Yes
	1	emac1_iq_class_entry1	Yes	Yes	Yes	0x0	Absolute	Yes
	2	emac1_iq_class_entry2	Yes	Yes	Yes	0x0	Absolute	Yes
	3	emac1_iq_class_entry3	Yes	Yes	Yes	0x0	Absolute	Yes
	4	emac1_iq_lru_count	Yes	Yes	Yes	0x0	Absolute	Yes
	5	emac1_iq_lmem1_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	6	emac1_iq_lmem2_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	7	emac1_iq_lmem3_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	8	emac1_iq_lmem4_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	9	emac1_iq_lmem5_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	10	emac1_iq_lmem6_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	11	emac1_iq_lmem7_rd	Yes	Yes	Yes	0x0	Absolute	Yes
	12	emac1_iq_lmem8_rd	Yes	Yes	Yes	0x0	Absolute	Yes
ERM_LLCE	0	Data RAM_A	Yes	Yes	Yes	0x0	Absolute	Yes
	1	Data RAM_B	Yes	Yes	Yes	0x0	Absolute	Yes
	2	Reserved	NA	NA	NA	NA	NA	NA
	3	FRPE RAM	Yes	Yes	Yes	0x0	Absolute	Yes
	4	TxPPE RAM	Yes	Yes	Yes	0x0	Absolute	Yes

Table continues on the next page...

Table 550. ERM instance channel event sources and captured information (continued)

Instance	Instance channel *n*	Memory *n* event source	Single-bit errors	Multi-bit errors	Address details			Event syndrome (bit position) (SYN*n*)
					Event address (EAR*n*)	Start address (hex)	Address alignment	
	5	RxPPE RAM	Yes	Yes	Yes	0x0	Absolute	Yes
	6	DTE RAM	Yes	Yes	Yes	0x0	Absolute	Yes
	7	Reserved	NA	NA	NA	NA	NA	NA
	8	Reserved	NA	NA	NA	NA	NA	NA
ERM_Standby_RAM	0	Standby SRAM (SRAMC)	Yes	No	Yes	0x0	Absolute	Yes

65.3 Introduction

65.3.1 Overview

The Error Reporting Module (ERM) provides information and optional interrupt notification on memory error events associated with ECC and parity. The ERM collects error events on memory accesses for memory arrays, such as flash memory, system RAM, or peripheral RAMs. ERM supports various channels for memory sources where each ERM channel is associated with a different memory module. See the chip-specific ERM information for details about supported memory sources and specific memory channel assignments. If memory supports ECC then ERM syndrome and error address information is captured along with error event. ERM does not receive this information in case of cache or memory with parity along with error event.

65.3.2 Features

The ERM includes these features:

- Optional interrupt notification on captured error events
- Capturing of address and syndrome information on single-bit correction and non-correctable ECC events
- Support for error event capturing for memory sources, with individual reporting fields and interrupt configuration per memory channel
- Recording the count value of the number of corrected error events

65.4 ERM_CPU0 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.4.1 ERM_CPU0 memory map

ERM_CPU0 base address: 4031_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h

65.4.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e2-4821-48a8-23f1-3906a0b6a0d1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

Provided under NDA COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-4821-48a8-1251-39013ab62401

65.4.3 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		0				0			
W	W1C	W1C			W1C	W1C										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30	NCE0

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE0	Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.4.4 ERM Memory a Error Address Register (EAR0 - EAR1)

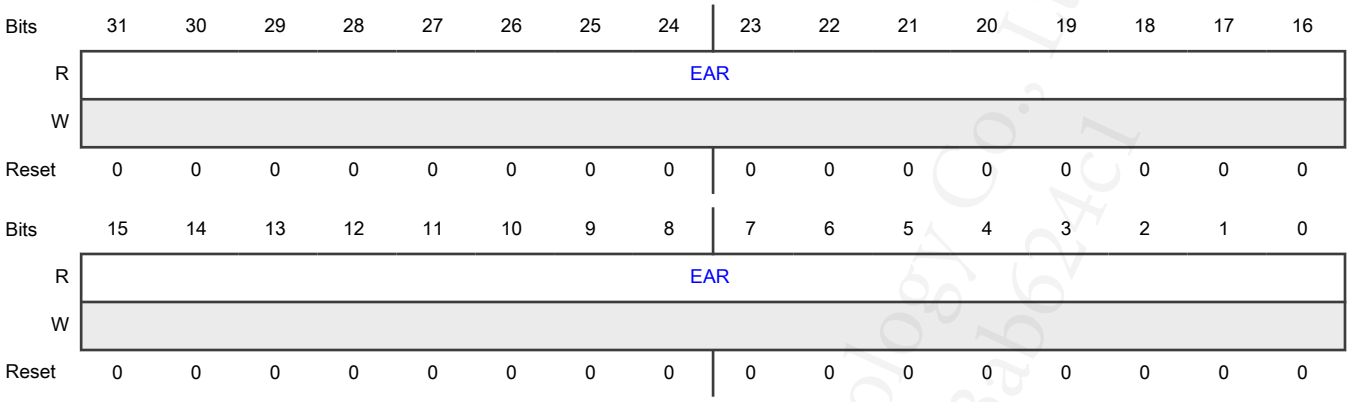
Offset

Register	Offset
EAR0	100h
EAR1	110h

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.4.5 ERM Memory a Syndrome Register (SYN0 - SYN1)

Offset

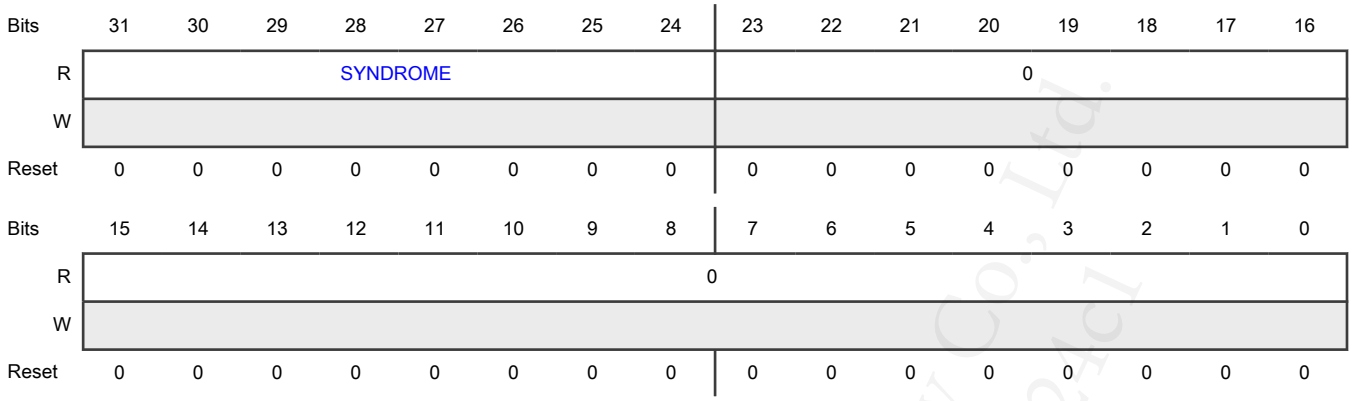
Register	Offset
SYN0	104h
SYN1	114h

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
CONFIDENTIAL NY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.4.6 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT5)

Offset

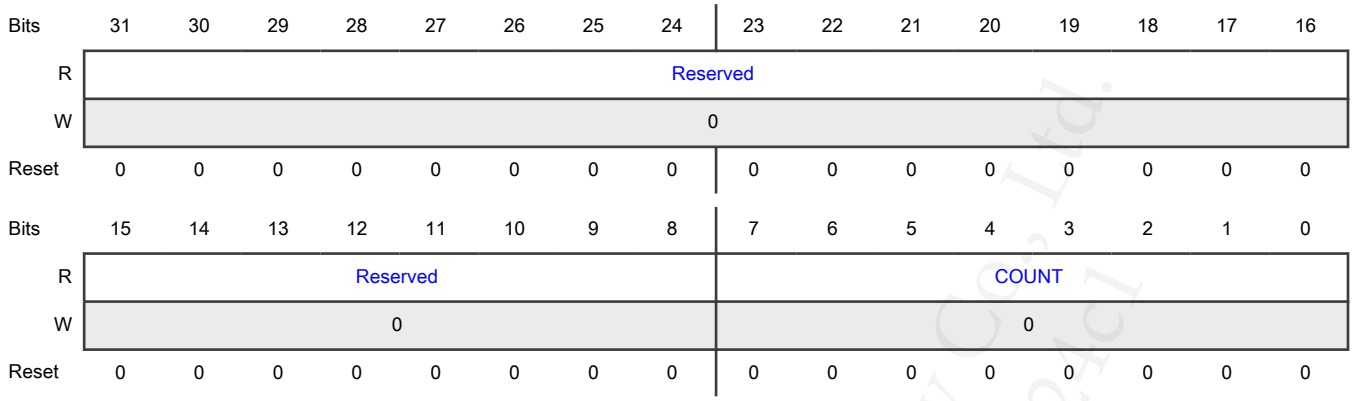
Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c40e54-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.5 ERM_CPU1 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.5.1 ERM_CPU1 memory map

ERM_CPU1 base address: 4031_8400h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h

65.5.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b231-29f0a3a62401

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

Provided under NDA COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-4821-48a8-1251-39013ab62401

65.5.3 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		0				0			
W	W1C	W1C			W1C	W1C										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30	NCE0

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE0	Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.5.4 ERM Memory a Error Address Register (EAR0 - EAR1)

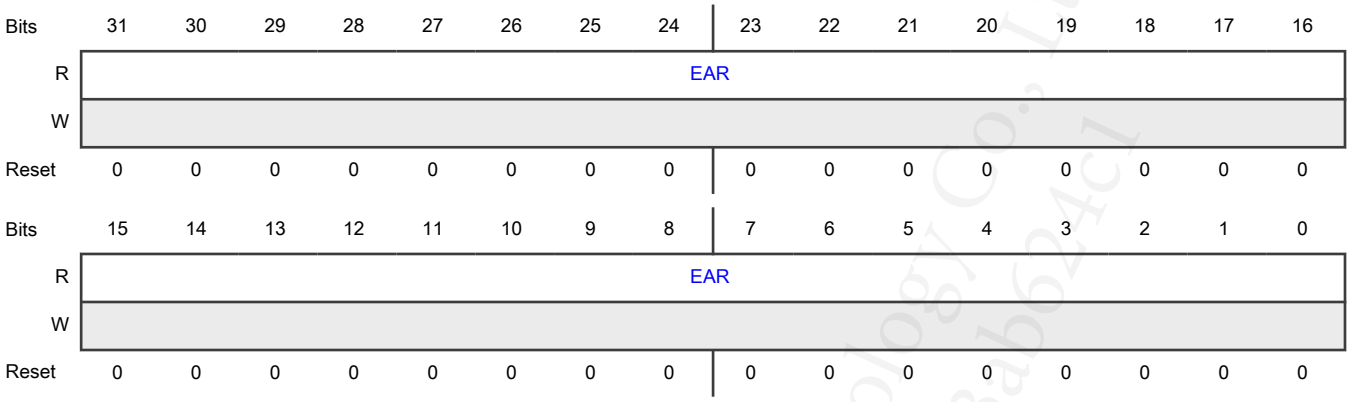
Offset

Register	Offset
EAR0	100h
EAR1	110h

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.5.5 ERM Memory a Syndrome Register (SYN0 - SYN1)

Offset

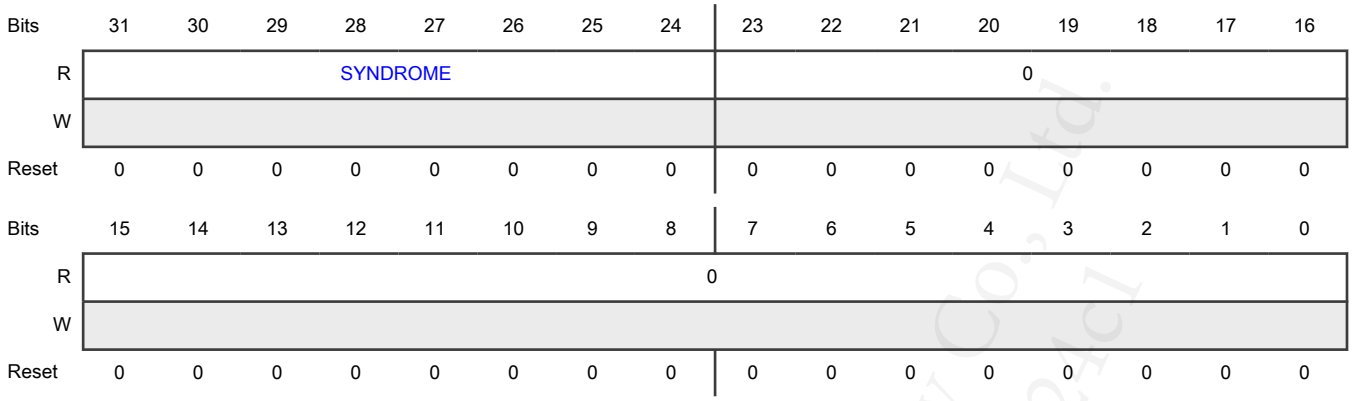
Register	Offset
SYN0	104h
SYN1	114h

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
CONFIDENTIAL NY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.5.6 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT5)

Offset

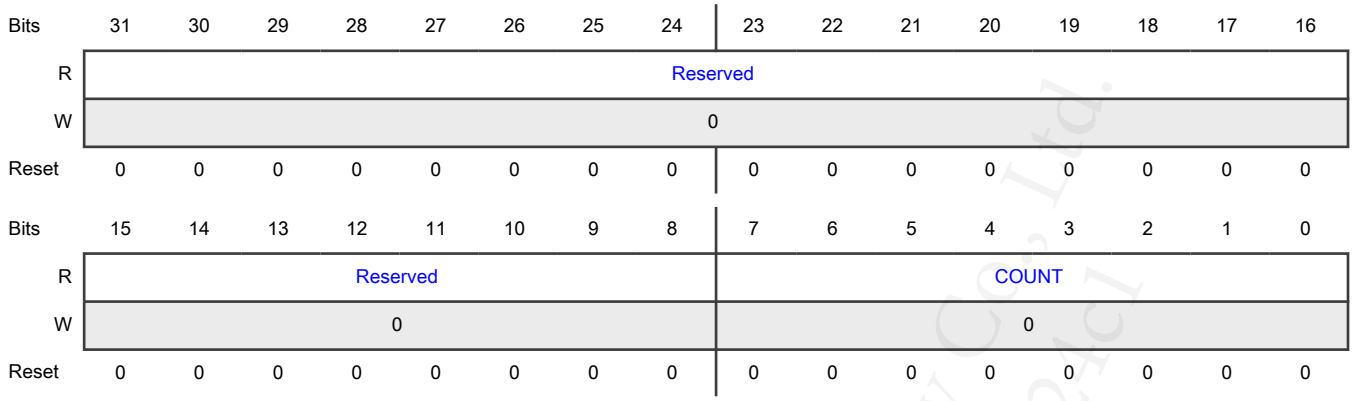
Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c40e54-821-48a8-b251-39dd3ab624c1

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.6 ERM_CPU2 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.6.1 ERM_CPU2 memory map

ERM_CPU2 base address: 4031_8800h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h

65.6.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b231-29f0a3a6240f

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

Provided under NDA COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-4821-48a8-1251-39013ab62401

65.6.3 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		0				0			
W	W1C	W1C			W1C	W1C										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30	NCE0

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE0	Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.6.4 ERM Memory a Error Address Register (EAR0 - EAR1)

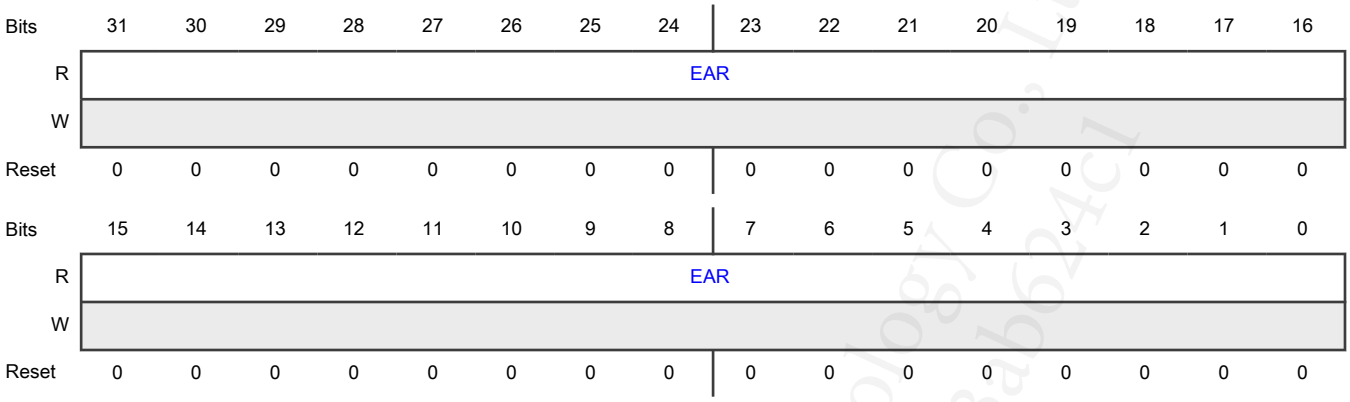
Offset

Register	Offset
EAR0	100h
EAR1	110h

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.6.5 ERM Memory a Syndrome Register (SYN0 - SYN1)

Offset

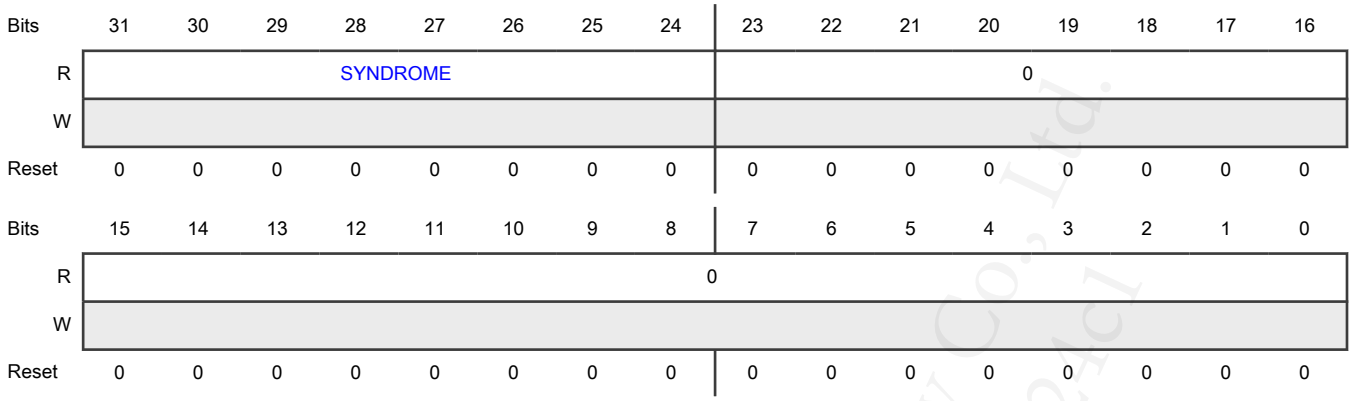
Register	Offset
SYN0	104h
SYN1	114h

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
CONFIDENTIAL NXP PROPRIETARY
Beijing HJ Automotive Technology Co., Ltd.
c8c462e3-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.6.6 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT5)

Offset

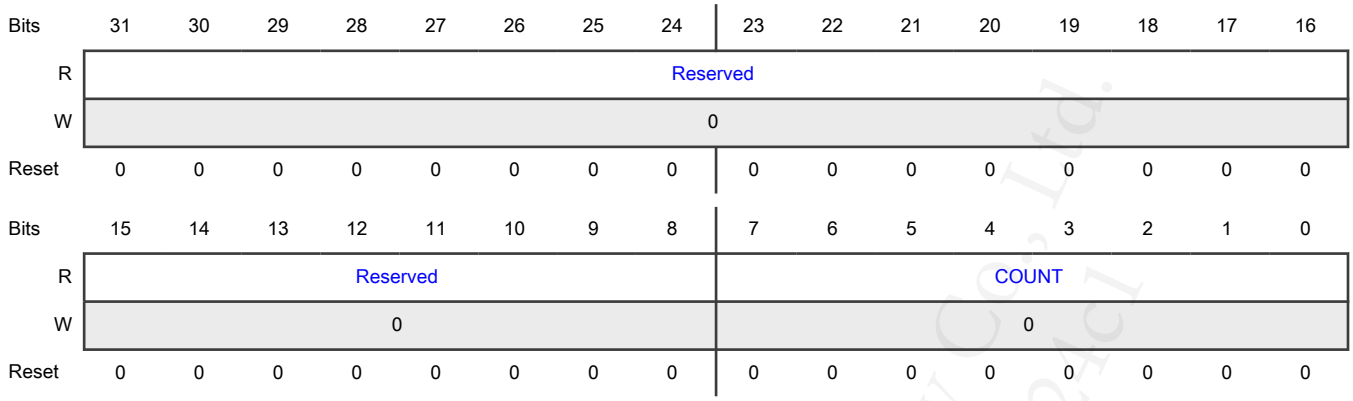
Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c40e54-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.7 ERM_PER register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.7.1 ERM_PER memory map

ERM_PER base address: 4031_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT 9)	32	ROWZ	0000_0000h

Provided under NDA only
COMPANY CONFIDENTIAL
Beijing eSOL Automotive Technology Co.,Ltd.
cache:2021-04-8-b51:39:03a62d1

65.7.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30	ENCIE0

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE0	Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19	ESCIE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE3	Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e2-4821-4888-b251-390d3b624c1

65.7.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE8	ENCIE8	0		ESCIE9	ENCIE9	0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.7.4 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Bejing CHJ Automotive Technology Co., Ltd.
C0462e5-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23	SBC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC2	Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESUN Automotive Technology Co., Ltd.
c8c462e5-4821-48a0-b251-390d3ab624c1

65.7.5 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		0				0			
W	W1C	W1C			W1C	W1C										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	

65.7.6 ERM Memory a Error Address Register (EAR0 - EAR9)

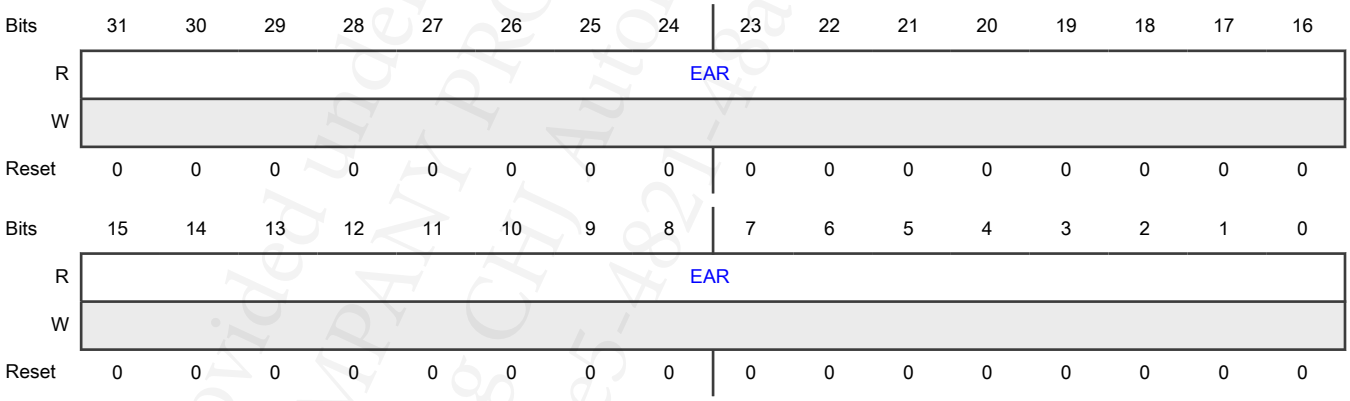
Offset

Register	Offset
EAR0	100h
EAR1	110h
EAR2	120h
EAR3	130h
EAR4	140h
EAR5	150h
EAR8	180h
EAR9	190h

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4da8-9251-390d3ab624c1

65.7.7 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT9)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								COUNT							
W	0								0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count

Table continues on the next page...

Table continued from the previous page...

Field	Function
	For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.7.8 ERM Memory a Syndrome Register (SYN8 - SYN9)

Offset

Register	Offset
SYN8	184h
SYN9	194h

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SYNDROME								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.

Table continues on the next page...

Table continued from the previous page...

Field	Function
23-0 —	Reserved

65.8 ERM_PFE0 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.8.1 ERM_PFE0 memory map

ERM_PFE0 base address: 4403_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT 9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_ CNT10)	32	ROWZ	0000_0000h

65.8.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30	ENCIE0

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE0	Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19	ESCIE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE3	Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e2-4821-48c8-b251-390d3b624c1

65.8.3 ERM Configuration Register 1 (CR1)

Offset

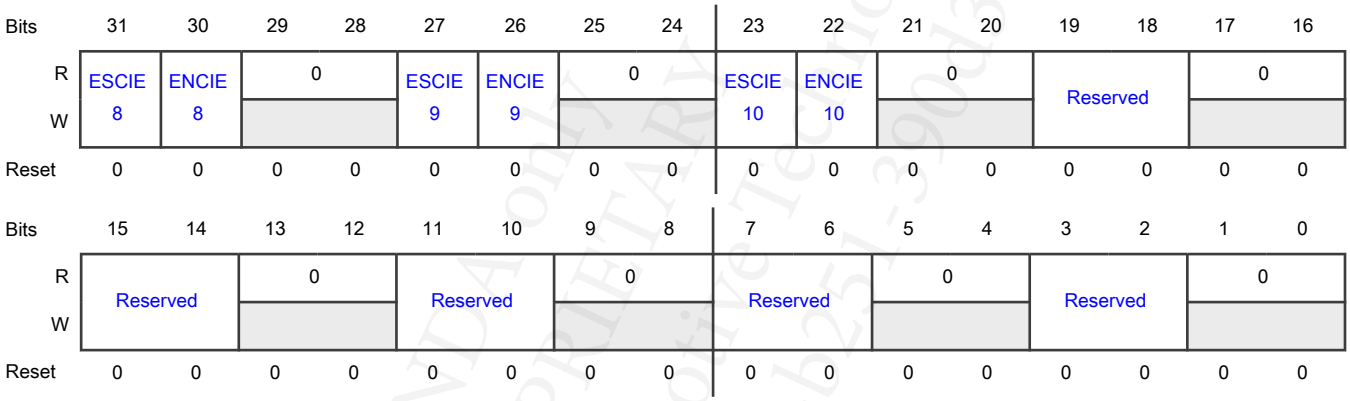
Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10

Diagram



Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.8.4 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-1821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18	NCE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE3	Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.8.5 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		0			
W	W1C	W1C			W1C	W1C			W1C	W1C						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	SBC8 Memory 8 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC8	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23 SBC10	SBC10 Memory 10 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.8.6 ERM Memory a Error Address Register (EAR0 - EAR10)

Offset

For a = 0 to 10:

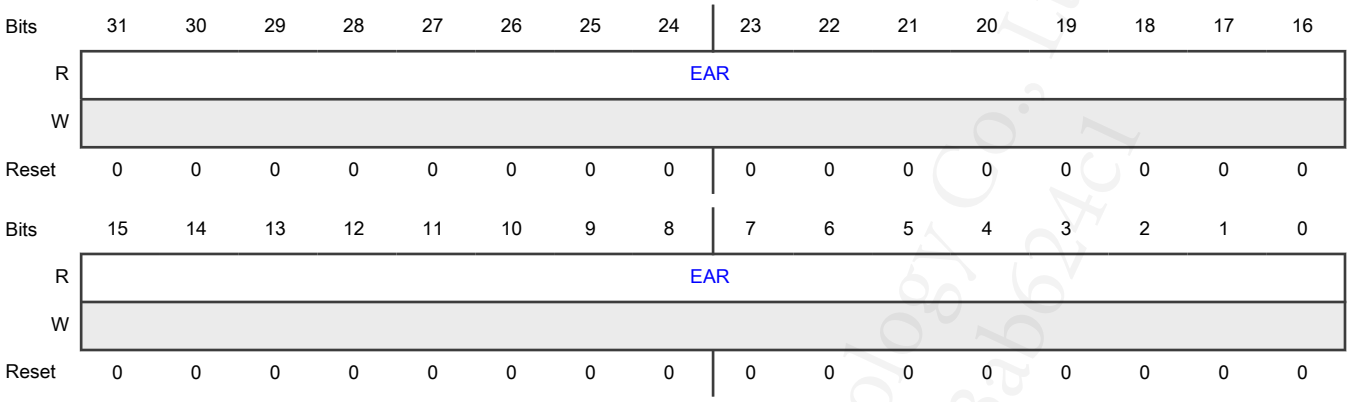
Register	Offset
EARa	100h + (a × 10h)

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-1821-48a8-b251-390d3ab624c1

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.8.7 ERM Memory a Syndrome Register (SYN0 - SYN10)

Offset

For a = 0 to 10:

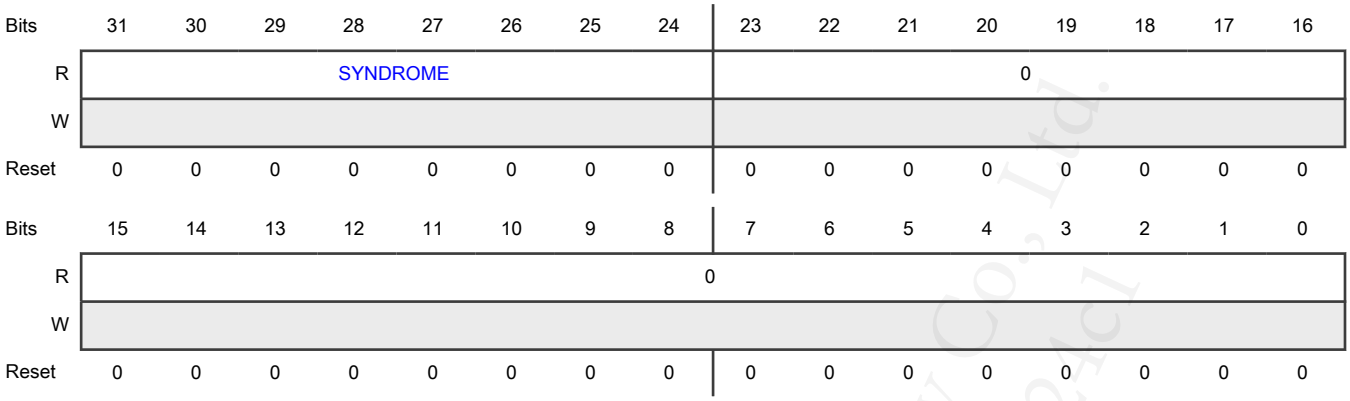
Register	Offset
SYNa	104h + (a × 10h)

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
CONTAINS NXP PROPRIETARY
Beijing HJ Automotive Technology Co., Ltd.
c8c4626e-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.8.8 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT10)

Offset

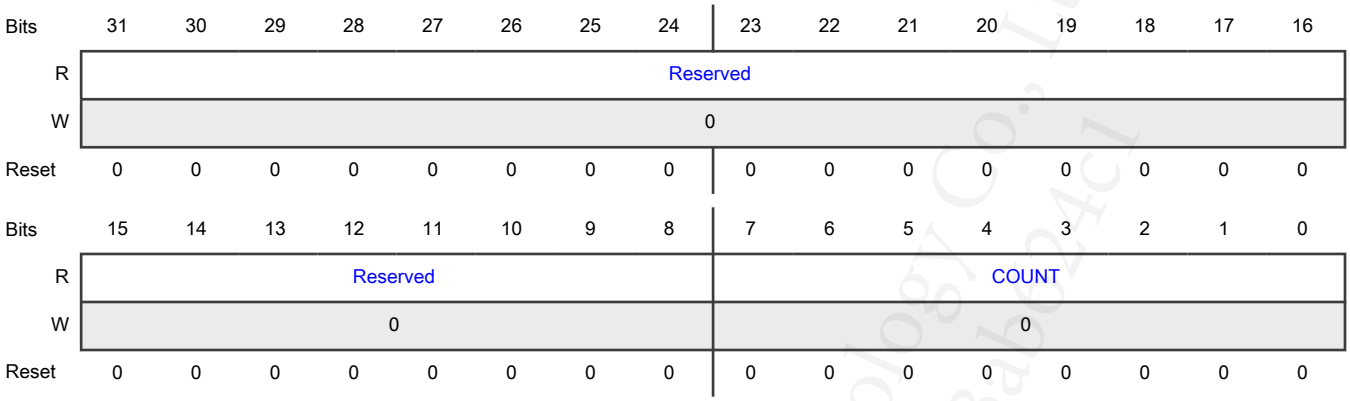
Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.9 ERM_PFE1 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.9.1 ERM_PFE1 memory map

ERM_PFE1 base address: 4403_5000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT 9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_ CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_ CNT11)	32	ROWZ	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_ CNT12)	32	ROWZ	0000_0000h

65.9.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

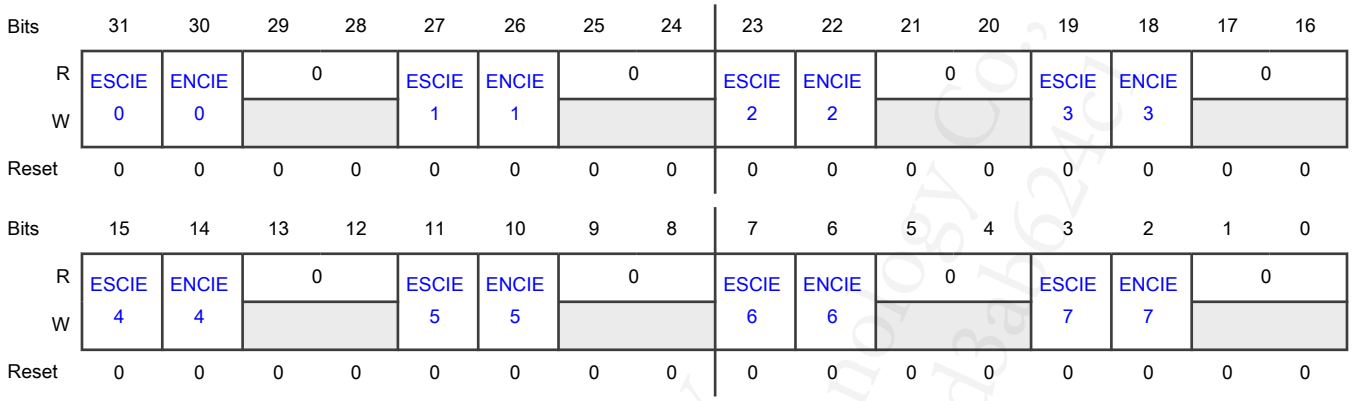
Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3

- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram



Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

65.9.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3b624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE8	ENCIE8	0		ESCIE9	ENCIE9	0		ESCIE10	ENCIE10	0		ESCIE11	ENCIE11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE12	ENCIE12	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.9.4 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab62401

- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.9.5 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23 SBC10	SBC10 Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18	NCE11

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE11	Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-45a8-b251-590d3ab624c1

65.9.6 ERM Memory a Error Address Register (EAR0 - EAR12)

Offset

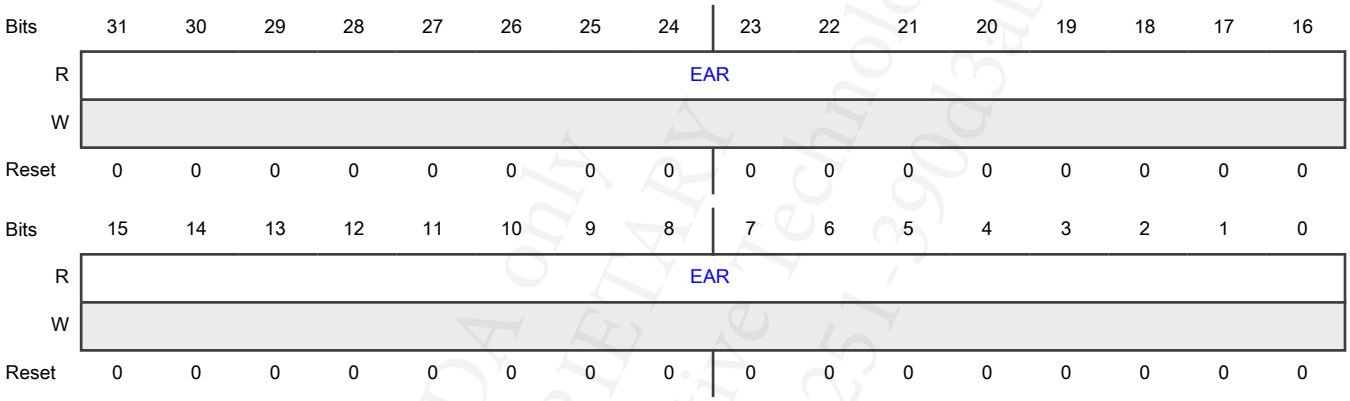
For a = 0 to 12:

Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.9.7 ERM Memory a Syndrome Register (SYN0 - SYN12)

Offset

For a = 0 to 12:

Register	Offset
SYNa	104h + (a × 10h)

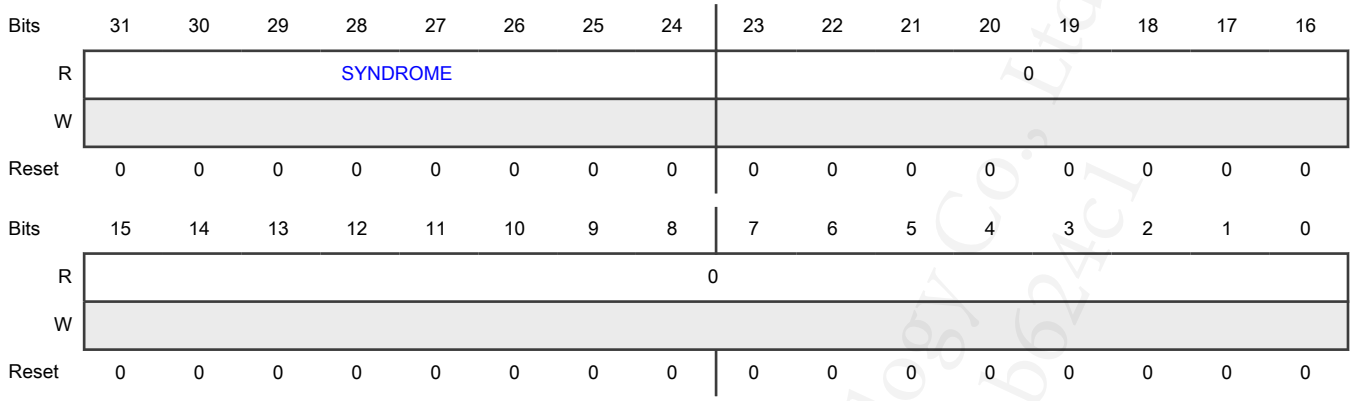
Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-421-48a8-b251-590a3b624c1

position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.9.8 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT12)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h

Table continues on the next page...

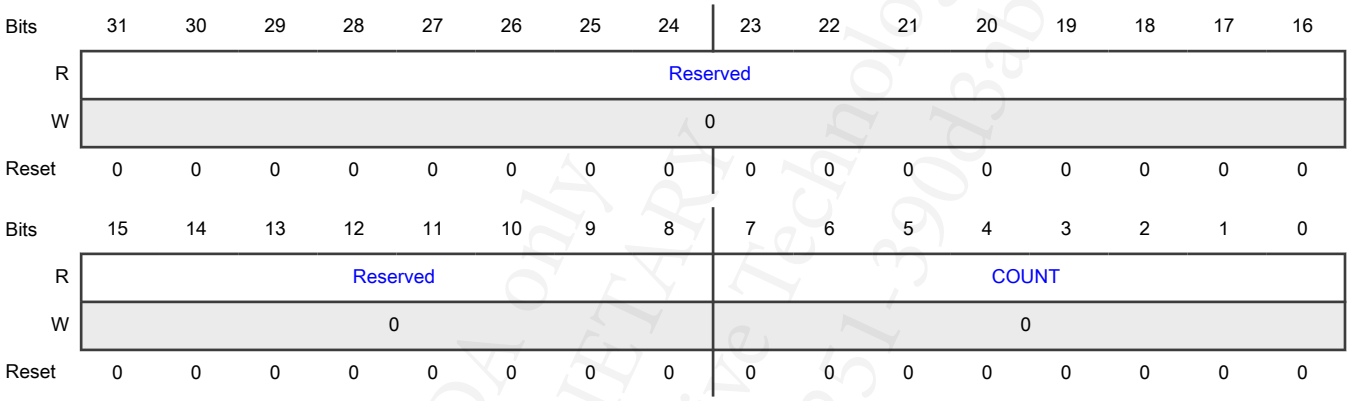
Table continued from the previous page...

Register	Offset
CORR_ERR_CNT10	1A8h
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.10 ERM_PFE10 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode

- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.10.1 ERM_PFE10 memory map

ERM_PFE10 base address: 4403_E000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT 9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_ CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_ CNT11)	32	ROWZ	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_ CNT12)	32	ROWZ	0000_0000h

Preliminary Information
COMPANY CONFIDENTIAL
Beijing ChipON Technology Co., Ltd.
c8c462e-4871-4a8-b151-5003a6b624c1

65.10.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30	ENCIE0

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE0	Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19	ESCIE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE3	Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e2-4821-48c8-b251-390d3b624c1

65.10.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE 8	ENCIE 8	0		ESCIE 9	ENCIE 9	0		ESCIE 10	ENCIE 10	0		ESCIE 11	ENCIE 11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE 12	ENCIE 12	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1-0 —	Reserved

65.10.4 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CINA Automotive Technology Co., Ltd.
c8c46e25-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23	SBC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC2	Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESUN Automotive Technology Co., Ltd.
c8c462e5-4821-48a0-b251-390d3ab624c1

65.10.5 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23 SBC10	SBC10 Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.10.6 ERM Memory a Error Address Register (EAR0 - EAR12)

Offset

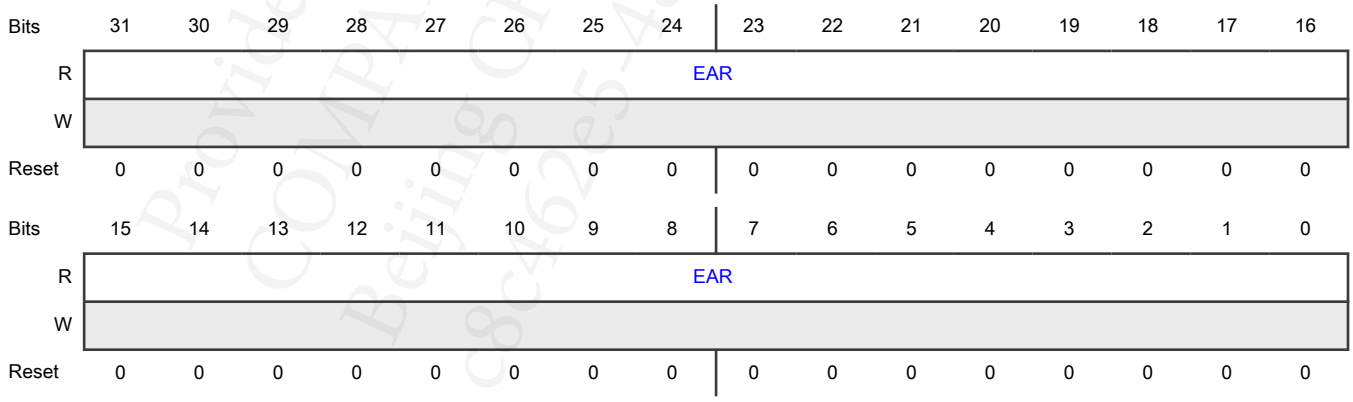
For a = 0 to 12:

Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.10.7 ERM Memory a Syndrome Register (SYN0 - SYN12)

Offset

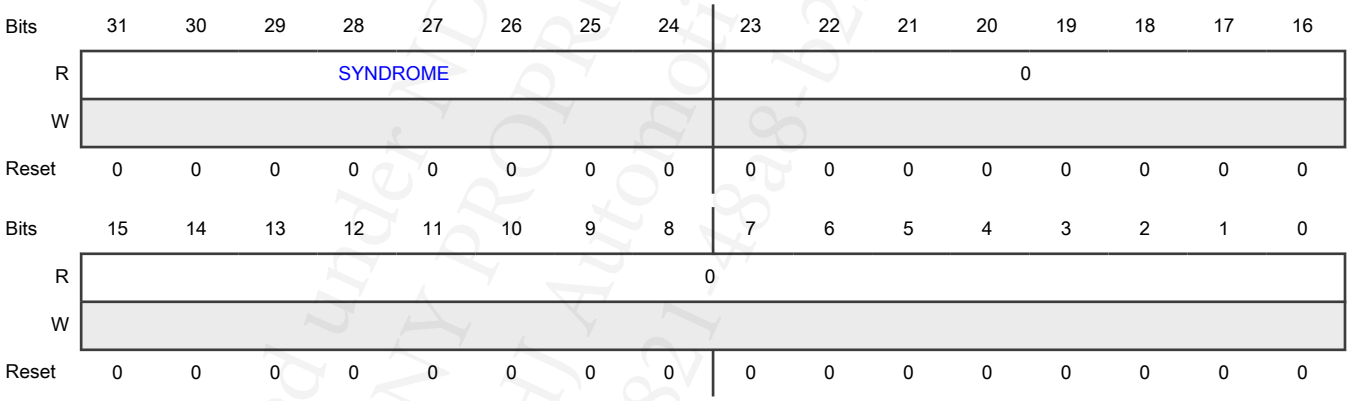
For a = 0 to 12:

Register	Offset
SYNa	104h + (a × 10h)

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.10.8 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT12)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								COUNT							
W	0								0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.11 ERM_PFE11 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.11.1 ERM_PFE11 memory map

ERM_PFE11 base address: 4403_F000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
18h	ERM Status Register 2 (SR2)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_CNT11)	32	ROWZ	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_CNT12)	32	ROWZ	0000_0000h
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	RO	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	RO	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_CNT13)	32	ROWZ	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	RO	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	RO	0000_0000h
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_CNT14)	32	ROWZ	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	RO	0000_0000h
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	RO	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_CNT15)	32	ROWZ	0000_0000h
200h	ERM Memory 16 Error Address Register (EAR16)	32	RO	0000_0000h
204h	ERM Memory 16 Syndrome Register (SYN16)	32	RO	0000_0000h
208h	ERM Memory 16 Correctable Error Count Register (CORR_ERR_CNT16)	32	ROWZ	0000_0000h
210h	ERM Memory 17 Error Address Register (EAR17)	32	RO	0000_0000h
214h	ERM Memory 17 Syndrome Register (SYN17)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
218h	ERM Memory 17 Correctable Error Count Register (CORR_ERR_CNT17)	32	ROWZ	0000_0000h
220h	ERM Memory 18 Error Address Register (EAR18)	32	RO	0000_0000h
224h	ERM Memory 18 Syndrome Register (SYN18)	32	RO	0000_0000h
228h	ERM Memory 18 Correctable Error Count Register (CORR_ERR_CNT18)	32	ROWZ	0000_0000h
230h	ERM Memory 19 Error Address Register (EAR19)	32	RO	0000_0000h
234h	ERM Memory 19 Syndrome Register (SYN19)	32	RO	0000_0000h
238h	ERM Memory 19 Correctable Error Count Register (CORR_ERR_CNT19)	32	ROWZ	0000_0000h
240h	ERM Memory 20 Error Address Register (EAR20)	32	RO	0000_0000h
244h	ERM Memory 20 Syndrome Register (SYN20)	32	RO	0000_0000h
248h	ERM Memory 20 Correctable Error Count Register (CORR_ERR_CNT20)	32	ROWZ	0000_0000h
250h	ERM Memory 21 Error Address Register (EAR21)	32	RO	0000_0000h
254h	ERM Memory 21 Syndrome Register (SYN21)	32	RO	0000_0000h
258h	ERM Memory 21 Correctable Error Count Register (CORR_ERR_CNT21)	32	ROWZ	0000_0000h
260h	ERM Memory 22 Error Address Register (EAR22)	32	RO	0000_0000h
264h	ERM Memory 22 Syndrome Register (SYN22)	32	RO	0000_0000h
268h	ERM Memory 22 Correctable Error Count Register (CORR_ERR_CNT22)	32	ROWZ	0000_0000h
270h	ERM Memory 23 Error Address Register (EAR23)	32	RO	0000_0000h
274h	ERM Memory 23 Syndrome Register (SYN23)	32	RO	0000_0000h
278h	ERM Memory 23 Correctable Error Count Register (CORR_ERR_CNT23)	32	ROWZ	0000_0000h

65.11.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

65.11.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

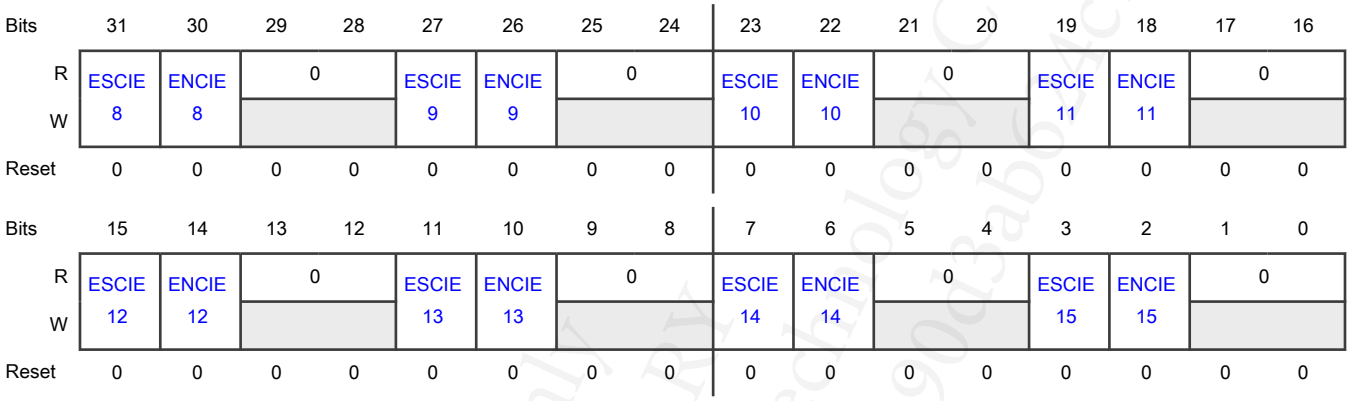
Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9

- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Diagram



Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 single-bit correction events is disabled. 1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 non-correctable error events is disabled. 1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0 —	Reserved

65.11.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 16
- Channel 17
- Channel 18
- Channel 19
- Channel 20

Provided under NDA COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-4821-48a8-1251-39013ab62401

- Channel 21
- Channel 22
- Channel 23

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE16	ENCIE16	0		ESCIE17	ENCIE17	0		ESCIE18	ENCIE18	0		ESCIE19	ENCIE19	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE20	ENCIE20	0		ESCIE21	ENCIE21	0		ESCIE22	ENCIE22	0		ESCIE23	ENCIE23	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE16	ESCIE16 Enable Memory 16 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 single-bit correction events is disabled. 1b - Interrupt notification of Memory 16 single-bit correction events is enabled.
30 ENCIE16	ENCIE16 Enable Memory 16 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 non-correctable error events is disabled. 1b - Interrupt notification of Memory 16 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE17	ESCIE17 Enable Memory 17 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 17 single-bit correction events is disabled. 1b - Interrupt notification of Memory 17 single-bit correction events is enabled.
26	ENCIE17

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE17	Enable Memory 17 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 17 non-correctable error events is disabled. 1b - Interrupt notification of Memory 17 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE18	ESCIE18 Enable Memory 18 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 18 single-bit correction events is disabled. 1b - Interrupt notification of Memory 18 single-bit correction events is enabled.
22 ENCIE18	ENCIE18 Enable Memory 18 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 18 non-correctable error events is disabled. 1b - Interrupt notification of Memory 18 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE19	ESCIE19 Enable Memory 19 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 19 single-bit correction events is disabled. 1b - Interrupt notification of Memory 19 single-bit correction events is enabled.
18 ENCIE19	ENCIE19 Enable Memory 19 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 19 non-correctable error events is disabled. 1b - Interrupt notification of Memory 19 non-correctable error events is enabled.
17-16 —	Reserved
15	ESCIE20

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE20	Enable Memory 20 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 20 single-bit correction events is disabled. 1b - Interrupt notification of Memory 20 single-bit correction events is enabled.
14 ENCIE20	ENCIE20 Enable Memory 20 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 20 non-correctable error events is disabled. 1b - Interrupt notification of Memory 20 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE21	ESCIE21 Enable Memory 21 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 21 single-bit correction events is disabled. 1b - Interrupt notification of Memory 21 single-bit correction events is enabled.
10 ENCIE21	ENCIE21 Enable Memory 21 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 21 non-correctable error events is disabled. 1b - Interrupt notification of Memory 21 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE22	ESCIE22 Enable Memory 22 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 22 single-bit correction events is disabled. 1b - Interrupt notification of Memory 22 single-bit correction events is enabled.
6 ENCIE22	ENCIE22 Enable Memory 22 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 22 non-correctable error events is disabled. 1b - Interrupt notification of Memory 22 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE23	ESCIE23 Enable Memory 23 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 23 single-bit correction events is disabled. 1b - Interrupt notification of Memory 23 single-bit correction events is enabled.
2 ENCIE23	ENCIE23 Enable Memory 23 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 23 non-correctable error events is disabled. 1b - Interrupt notification of Memory 23 non-correctable error events is enabled.
1-0 —	Reserved

65.11.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6

- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.11.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23 SBC10	SBC10 Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18	NCE11

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE11	Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled. 0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

65.11.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 16
- Channel 17
- Channel 18
- Channel 19
- Channel 20
- Channel 21
- Channel 22
- Channel 23

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC16	NCE16	0		SBC17	NCE17	0		SBC18	NCE18	0		SBC19	NCE19	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC20	NCE20	0		SBC21	NCE21	0		SBC22	NCE22	0		SBC23	NCE23	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESU Automotive Technology Co., Ltd.
ca6625e5-1821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC16	SBC16 Memory 16 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE16] is enabled. 0b - No single-bit correction event on Memory 16 detected. 1b - Single-bit correction event on Memory 16 detected.
30 NCE16	NCE16 Memory 16 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE16] is enabled. 0b - No non-correctable error event on Memory 16 detected. 1b - Non-correctable error event on Memory 16 detected.
29-28 —	Reserved
27 SBC17	SBC17 Memory 17 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE17] is enabled. 0b - No single-bit correction event on Memory 17 detected. 1b - Single-bit correction event on Memory 17 detected.
26 NCE17	NCE17 Memory 17 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE17] is enabled. 0b - No non-correctable error event on Memory 17 detected. 1b - Non-correctable error event on Memory 17 detected.
25-24 —	Reserved
23	SBC18

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC18	Memory 18 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE18] is enabled. 0b - No single-bit correction event on Memory 18 detected. 1b - Single-bit correction event on Memory 18 detected.
22 NCE18	NCE18 Memory 18 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE18] is enabled. 0b - No non-correctable error event on Memory 18 detected. 1b - Non-correctable error event on Memory 18 detected.
21-20 —	Reserved
19 SBC19	SBC19 Memory 19 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE19] is enabled. 0b - No single-bit correction event on Memory 19 detected. 1b - Single-bit correction event on Memory 19 detected.
18 NCE19	NCE19 Memory 19 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE19] is enabled. 0b - No non-correctable error event on Memory 19 detected. 1b - Non-correctable error event on Memory 19 detected.
17-16 —	Reserved
15 SBC20	SBC20 Memory 20 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE20] is enabled. 0b - No single-bit correction event on Memory 20 detected. 1b - Single-bit correction event on Memory 20 detected.
14 NCE20	NCE20 Memory 20 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE20] is enabled. 0b - No non-correctable error event on Memory 20 detected. 1b - Non-correctable error event on Memory 20 detected.
13-12 —	Reserved
11 SBC21	SBC21 Memory 21 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE21] is enabled. 0b - No single-bit correction event on Memory 21 detected. 1b - Single-bit correction event on Memory 21 detected.
10 NCE21	NCE21 Memory 21 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE21] is enabled. 0b - No non-correctable error event on Memory 21 detected. 1b - Non-correctable error event on Memory 21 detected.
9-8 —	Reserved
7 SBC22	SBC22 Memory 22 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE22] is enabled. 0b - No single-bit correction event on Memory 22 detected. 1b - Single-bit correction event on Memory 22 detected.
6 NCE22	NCE22 Memory 22 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE22] is enabled. 0b - No non-correctable error event on Memory 22 detected. 1b - Non-correctable error event on Memory 22 detected.
5-4 —	Reserved
3 SBC23	SBC23 Memory 23 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE23] is enabled. 0b - No single-bit correction event on Memory 23 detected. 1b - Single-bit correction event on Memory 23 detected.
2 NCE23	NCE23 Memory 23 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE23] is enabled. 0b - No non-correctable error event on Memory 23 has been not been detected. 1b - Non-correctable error event on Memory 23 detected.
1-0 —	Reserved

65.11.8 ERM Memory a Error Address Register (EAR0 - EAR23)

Offset

For a = 0 to 23:

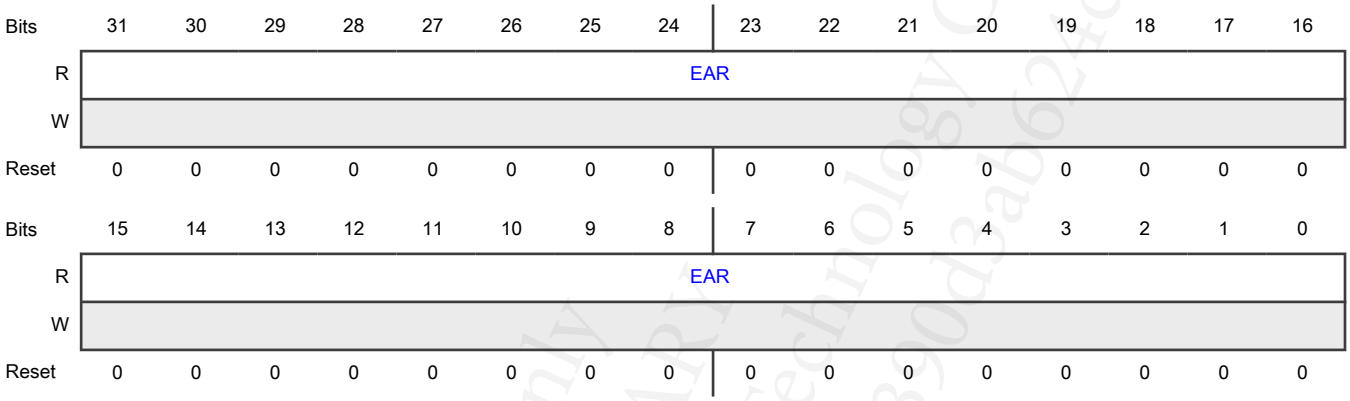
Provided under NDA only
COMPANY PROPRIETARY
Beijing ChiUtoInfotive Technology Co., Ltd.
c8c462e5-4821-48a3-b251-390d3ab624c1

Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.11.9 ERM Memory a Syndrome Register (SYN0 - SYN23)

Offset

For a = 0 to 23:

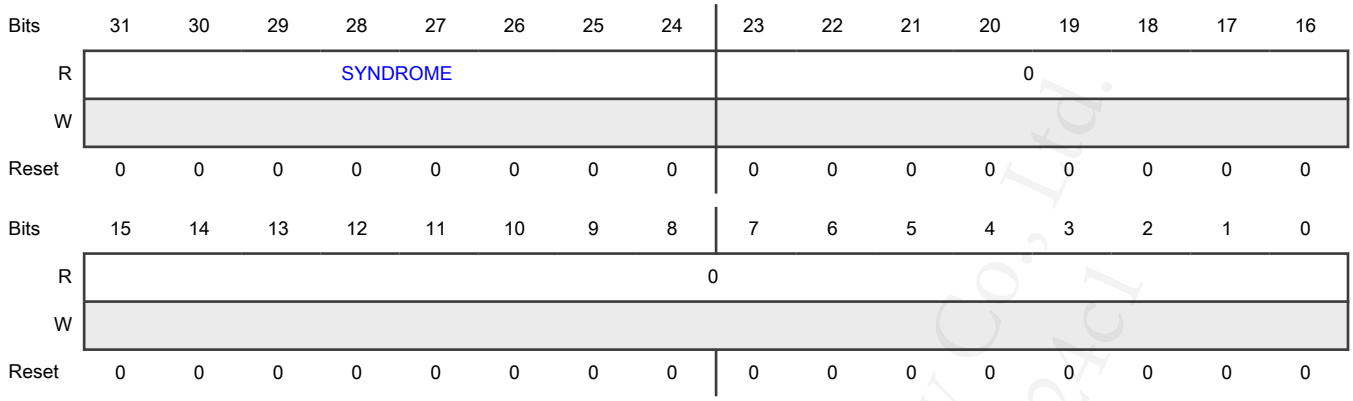
Register	Offset
SYNa	104h + (a × 10h)

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c2e5-4821-4a8-b251-390a3ab62fc1

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.11.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT 23)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h

Table continues on the next page...

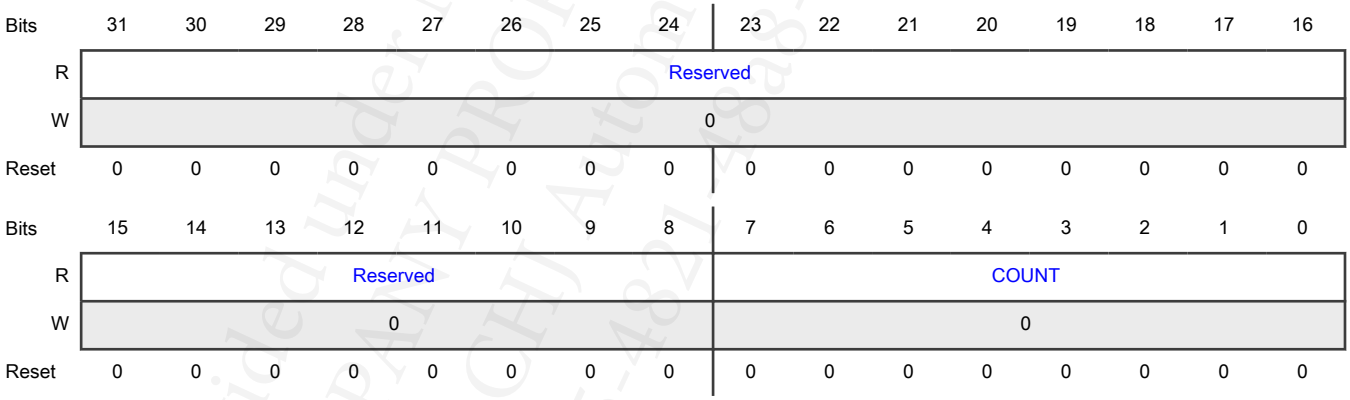
Table continued from the previous page...

Register	Offset
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT16	208h
CORR_ERR_CNT17	218h
CORR_ERR_CNT18	228h
CORR_ERR_CNT19	238h
CORR_ERR_CNT20	248h
CORR_ERR_CNT21	258h
CORR_ERR_CNT22	268h
CORR_ERR_CNT23	278h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram



Fields

Field	Function
31-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.12 ERM_PFE12 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.12.1 ERM_PFE12 memory map

ERM_PFE12 base address: 4404_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
18h	ERM Status Register 2 (SR2)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_CNT11)	32	ROWZ	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_CNT12)	32	ROWZ	0000_0000h
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	RO	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	RO	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_CNT13)	32	ROWZ	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	RO	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	RO	0000_0000h
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_CNT14)	32	ROWZ	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	RO	0000_0000h
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	RO	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_CNT15)	32	ROWZ	0000_0000h
200h	ERM Memory 16 Error Address Register (EAR16)	32	RO	0000_0000h
204h	ERM Memory 16 Syndrome Register (SYN16)	32	RO	0000_0000h
208h	ERM Memory 16 Correctable Error Count Register (CORR_ERR_CNT16)	32	ROWZ	0000_0000h

Preliminary Information
COMPANY CONFIDENTIAL
Beijing ChipLink Technology Co., Ltd.
c8c46251-4871-48a8-b151-39063a6624c1

65.12.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30	ENCIE0

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE0	Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19	ESCIE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE3	Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e2-4821-4888-b251-390d3b624c1

65.12.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE 8	ENCIE 8	0		ESCIE 9	ENCIE 9	0		ESCIE 10	ENCIE 10	0		ESCIE 11	ENCIE 11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE 12	ENCIE 12	0		ESCIE 13	ENCIE 13	0		ESCIE 14	ENCIE 14	0		ESCIE 15	ENCIE 15	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30	ENCIE8

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE8	Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19	ESCIE11

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE11	Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 single-bit correction events is disabled. 1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 non-correctable error events is disabled. 1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462c-4821-4888-b251-390d3b624c1

65.12.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE16	ENCIE16	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE16	ESCIE16 Enable Memory 16 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 single-bit correction events is disabled. 1b - Interrupt notification of Memory 16 single-bit correction events is enabled.
30 ENCIE16	ENCIE16 Enable Memory 16 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 non-correctable error events is disabled. 1b - Interrupt notification of Memory 16 non-correctable error events is enabled.
29-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

65.12.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22	NCE2

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE2	Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.12.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10

- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27	SBC9

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC9	Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23 SBC10	SBC10 Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled. 0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

65.12.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC16	NCE16	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CUJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC16	SBC16 Memory 16 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE16] is enabled. 0b - No single-bit correction event on Memory 16 detected. 1b - Single-bit correction event on Memory 16 detected.
30 NCE16	NCE16 Memory 16 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE16] is enabled. 0b - No non-correctable error event on Memory 16 detected. 1b - Non-correctable error event on Memory 16 detected.
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-590a3ab624c1

65.12.8 ERM Memory a Error Address Register (EAR0 - EAR16)

Offset

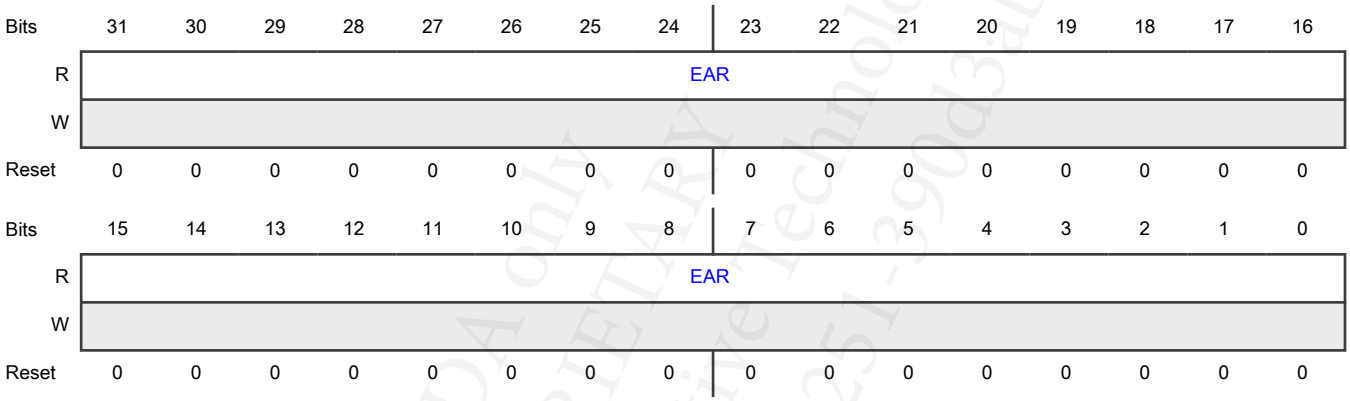
For a = 0 to 16:

Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.12.9 ERM Memory a Syndrome Register (SYN0 - SYN16)

Offset

For a = 0 to 16:

Register	Offset
SYNa	104h + (a × 10h)

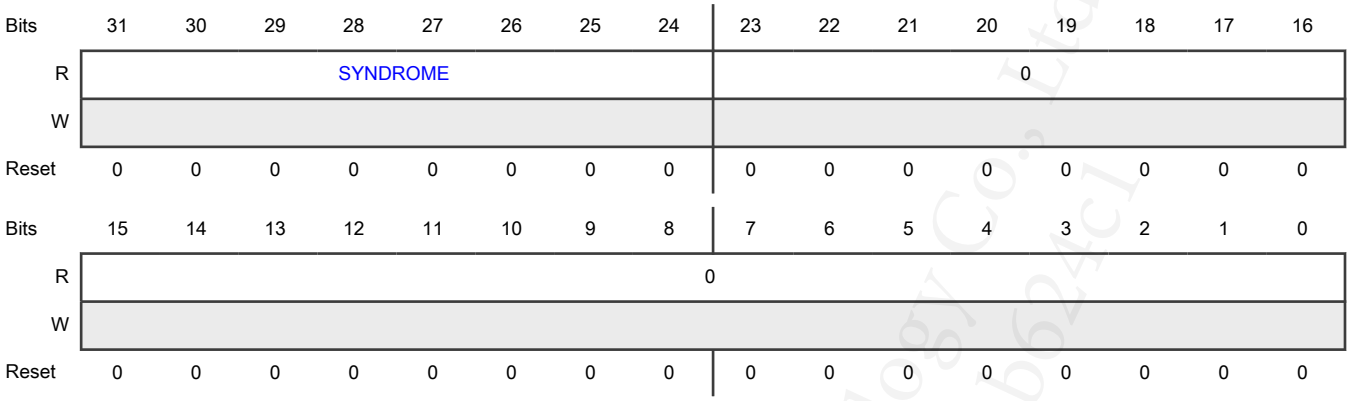
Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4d21-48a8-b251-590d3b624c1

position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.12.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT 16)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h

Table continues on the next page...

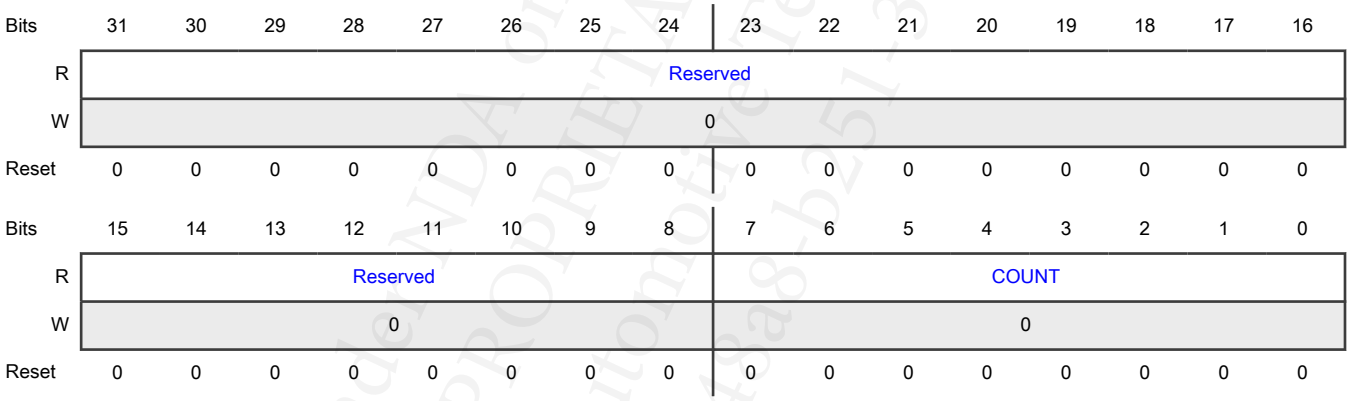
Table continued from the previous page...

Register	Offset
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT16	208h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8a6e5-4821-45a8-b251-3903ab624c1

65.13 ERM_PFE13 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.13.1 ERM_PFE13 memory map

ERM_PFE13 base address: 4404_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
18h	ERM Status Register 2 (SR2)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT 9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_ CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_ CNT11)	32	ROWZ	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_CNT12)	32	ROWZ	0000_0000h
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	RO	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	RO	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_CNT13)	32	ROWZ	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	RO	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	RO	0000_0000h
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_CNT14)	32	ROWZ	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	RO	0000_0000h
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	RO	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_CNT15)	32	ROWZ	0000_0000h
200h	ERM Memory 16 Error Address Register (EAR16)	32	RO	0000_0000h
204h	ERM Memory 16 Syndrome Register (SYN16)	32	RO	0000_0000h
208h	ERM Memory 16 Correctable Error Count Register (CORR_ERR_CNT16)	32	ROWZ	0000_0000h
210h	ERM Memory 17 Error Address Register (EAR17)	32	RO	0000_0000h
214h	ERM Memory 17 Syndrome Register (SYN17)	32	RO	0000_0000h
218h	ERM Memory 17 Correctable Error Count Register (CORR_ERR_CNT17)	32	ROWZ	0000_0000h

65.13.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

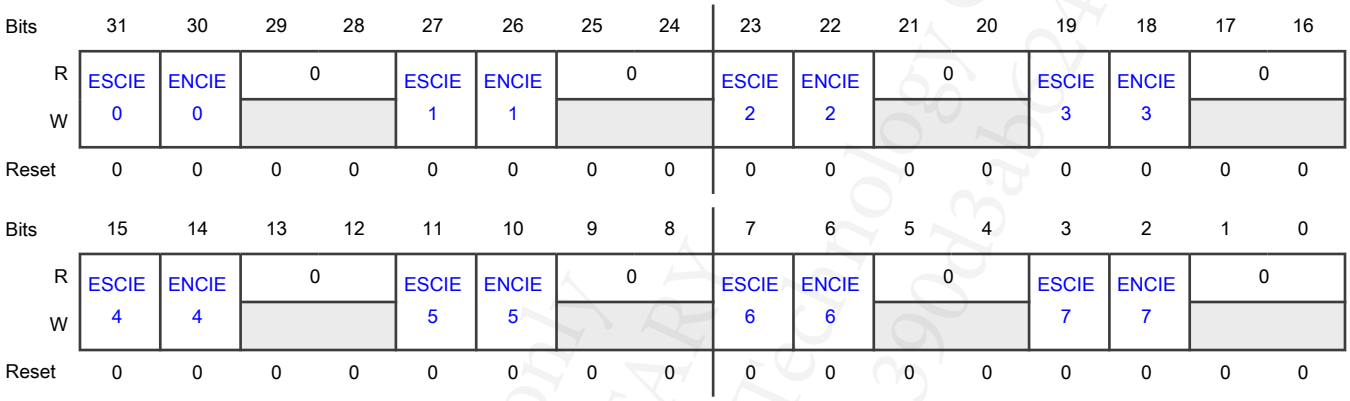
This 32-bit control register configures the interrupt notification capability for:

- Channel 0

Provided under NDA
COMPANY CONFIDENTIAL
Beijing CIU Automotive Technology Co.,Ltd.
8c462e5-4821-48ac-b251-390dab624c1

- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram



Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

65.13.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12

Provided under NDA COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-4821-48a8-1251-3901 3ab62401

- Channel 13
- Channel 14
- Channel 15

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE8	ENCIE8	0		ESCIE9	ENCIE9	0		ESCIE10	ENCIE10	0		ESCIE11	ENCIE11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE12	ENCIE12	0		ESCIE13	ENCIE13	0		ESCIE14	ENCIE14	0		ESCIE15	ENCIE15	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26	ENCIE9

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE9	Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15	ESCIE12

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE12	Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 single-bit correction events is disabled. 1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 non-correctable error events is disabled. 1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0 —	Reserved

65.13.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

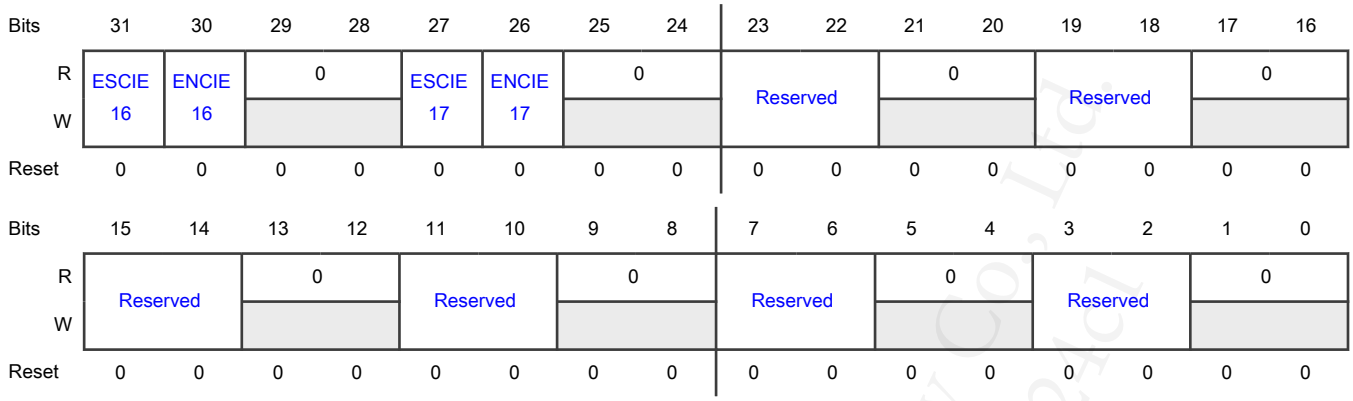
Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 16
- Channel 17

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 ESCIE16	ESCIE16 Enable Memory 16 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 single-bit correction events is disabled. 1b - Interrupt notification of Memory 16 single-bit correction events is enabled.
30 ENCIE16	ENCIE16 Enable Memory 16 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 non-correctable error events is disabled. 1b - Interrupt notification of Memory 16 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE17	ESCIE17 Enable Memory 17 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 17 single-bit correction events is disabled. 1b - Interrupt notification of Memory 17 single-bit correction events is enabled.
26 ENCIE17	ENCIE17 Enable Memory 17 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 17 non-correctable error events is disabled. 1b - Interrupt notification of Memory 17 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

65.13.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22	NCE2

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE2	Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.13.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10

- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27	SBC9

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC9	Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23 SBC10	SBC10 Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled. 0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

65.13.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 16
- Channel 17

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC16	NCE16	0		SBC17	NCE17	0		0				0			
W	W1C	W1C			W1C	W1C										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Bosch CHJ Automotive Technology Co., Ltd.
c8c67e54-821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC16	SBC16 Memory 16 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE16] is enabled. 0b - No single-bit correction event on Memory 16 detected. 1b - Single-bit correction event on Memory 16 detected.
30 NCE16	NCE16 Memory 16 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE16] is enabled. 0b - No non-correctable error event on Memory 16 detected. 1b - Non-correctable error event on Memory 16 detected.
29-28 —	Reserved
27 SBC17	SBC17 Memory 17 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE17] is enabled. 0b - No single-bit correction event on Memory 17 detected. 1b - Single-bit correction event on Memory 17 detected.
26 NCE17	NCE17 Memory 17 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE17] is enabled. 0b - No non-correctable error event on Memory 17 detected. 1b - Non-correctable error event on Memory 17 detected.
25-24 —	Reserved
23-20	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.13.8 ERM Memory a Error Address Register (EAR0 - EAR17)

Offset

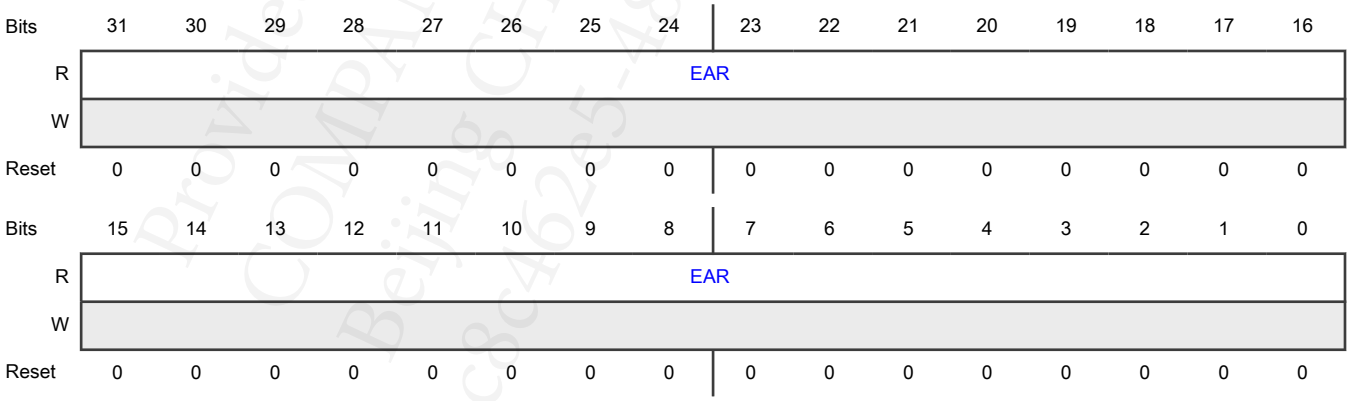
For a = 0 to 17:

Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46255-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.13.9 ERM Memory a Syndrome Register (SYN0 - SYN17)

Offset

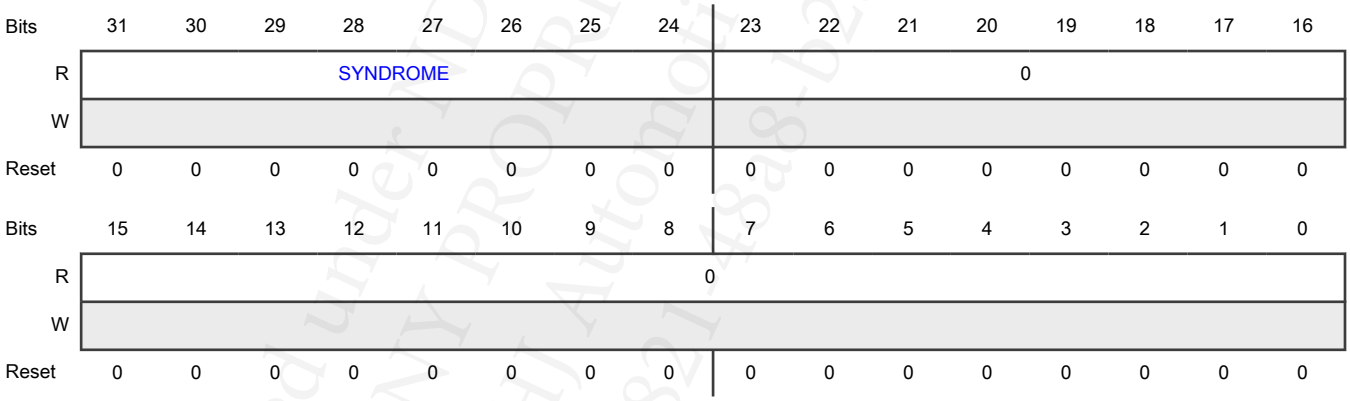
For a = 0 to 17:

Register	Offset
SYNa	104h + (a × 10h)

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.13.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT 17)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT16	208h
CORR_ERR_CNT17	218h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								COUNT							
W	0								0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8462e5-482-48a8-b251-390d3ab624c1

Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.14 ERM_PFE14 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.14.1 ERM_PFE14 memory map

ERM_PFE14 base address: 4404_2000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_CNT11)	32	ROWZ	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_CNT12)	32	ROWZ	0000_0000h

65.14.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39013a6624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

65.14.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-43a8-b251-390d3b624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE8	ENCIE8	0		ESCIE9	ENCIE9	0		ESCIE10	ENCIE10	0		ESCIE11	ENCIE11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE12	ENCIE12	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.14.4 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab62401

- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.14.5 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23 SBC10	SBC10 Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18	NCE11

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE11	Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-45a8-b251-590d3ab624c1

65.14.6 ERM Memory a Error Address Register (EAR0 - EAR12)

Offset

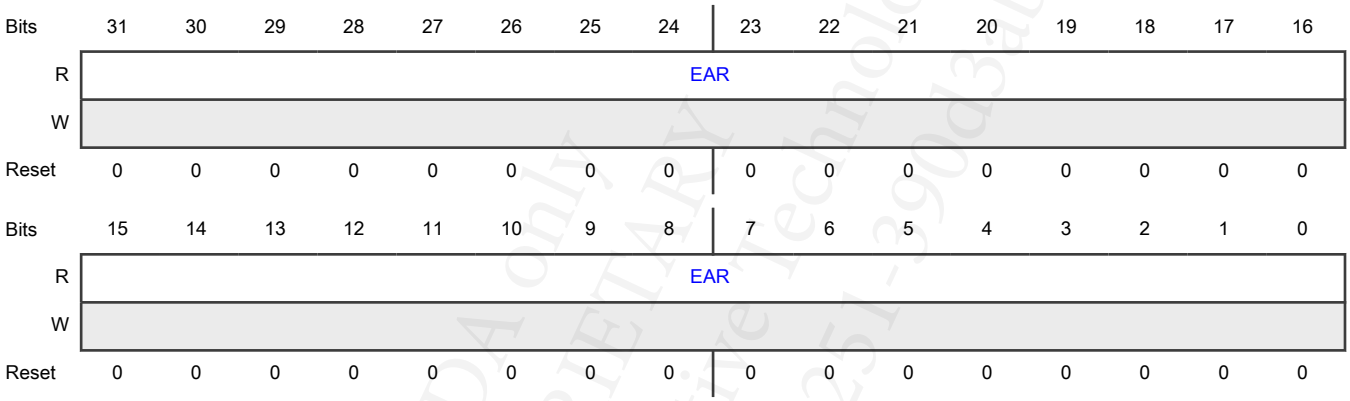
For a = 0 to 12:

Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.14.7 ERM Memory a Syndrome Register (SYN0 - SYN12)

Offset

For a = 0 to 12:

Register	Offset
SYNa	104h + (a × 10h)

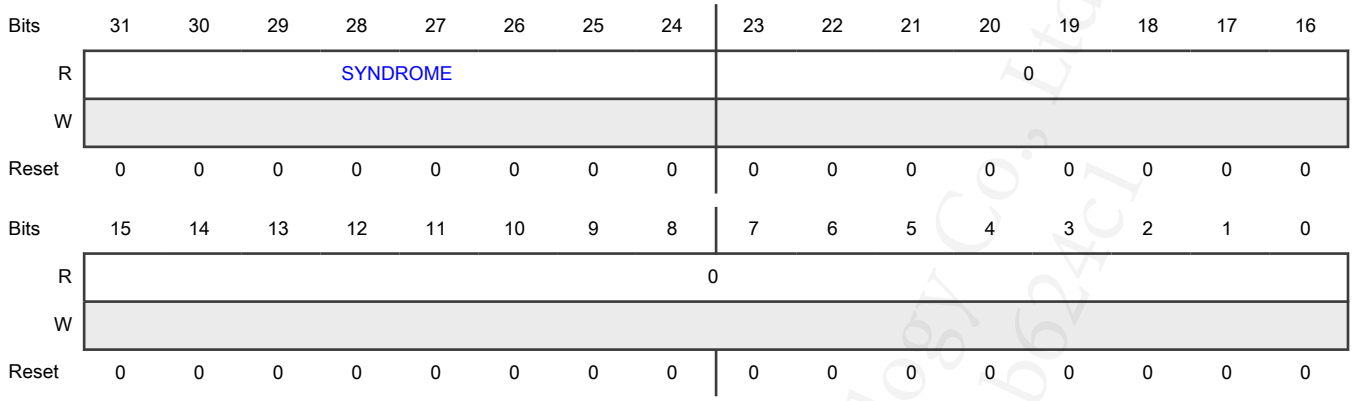
Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4d21-48a8-b251-590d3b624c1

position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.14.8 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT12)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h

Table continues on the next page...

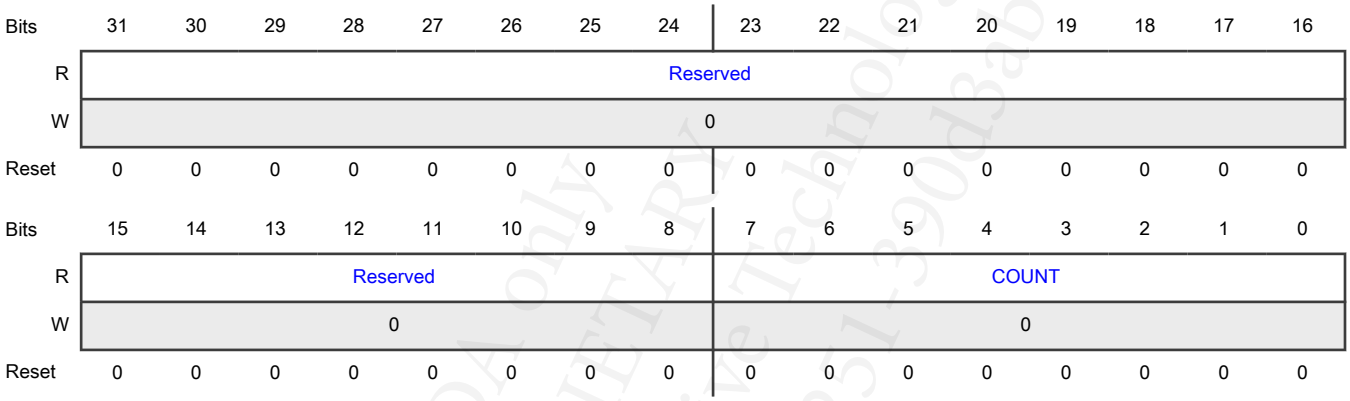
Table continued from the previous page...

Register	Offset
CORR_ERR_CNT10	1A8h
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.15 ERM_PFE15 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode

- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.15.1 ERM_PFE15 memory map

ERM_PFE15 base address: 4404_3000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT 9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_ CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_ CNT11)	32	ROWZ	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_ CNT12)	32	ROWZ	0000_0000h

Preliminary Information
COMPANY CONFIDENTIAL
Beijing ChipON Technology Co., Ltd.
c8c46251-4871-4a8-b151-5003a6b624c1

65.15.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE 0	ENCIE 0	0		ESCIE 1	ENCIE 1	0		ESCIE 2	ENCIE 2	0		ESCIE 3	ENCIE 3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE 4	ENCIE 4	0		ESCIE 5	ENCIE 5	0		ESCIE 6	ENCIE 6	0		ESCIE 7	ENCIE 7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30	ENCIE0

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE0	Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19	ESCIE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE3	Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e2-4821-4888-b251-390d3b624c1

65.15.3 ERM Configuration Register 1 (CR1)

Offset

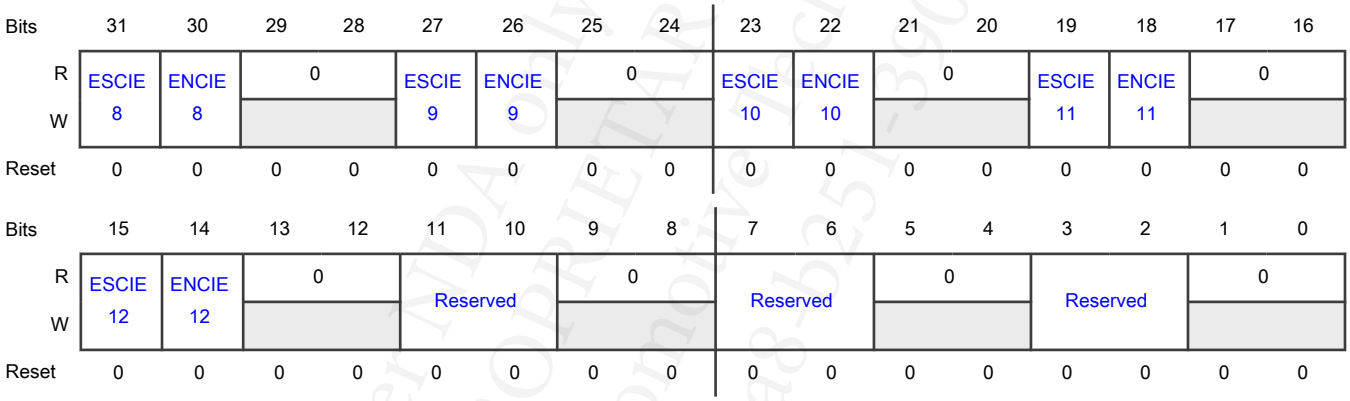
Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12

Diagram



Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1-0 —	Reserved

65.15.4 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CINO Automotive Technology Co., Ltd.
c8c46e25-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23	SBC2

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC2	Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESUN Automotive Technology Co., Ltd.
c8c462e5-4821-48a0-b251-390d3ab624c1

65.15.5 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23 SBC10	SBC10 Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.15.6 ERM Memory a Error Address Register (EAR0 - EAR12)

Offset

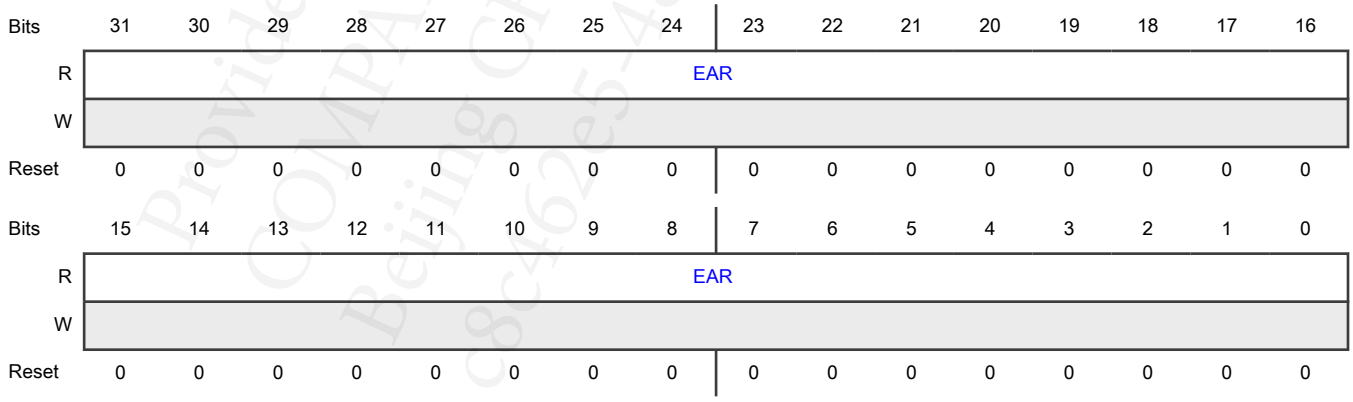
For a = 0 to 12:

Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.15.7 ERM Memory a Syndrome Register (SYN0 - SYN12)

Offset

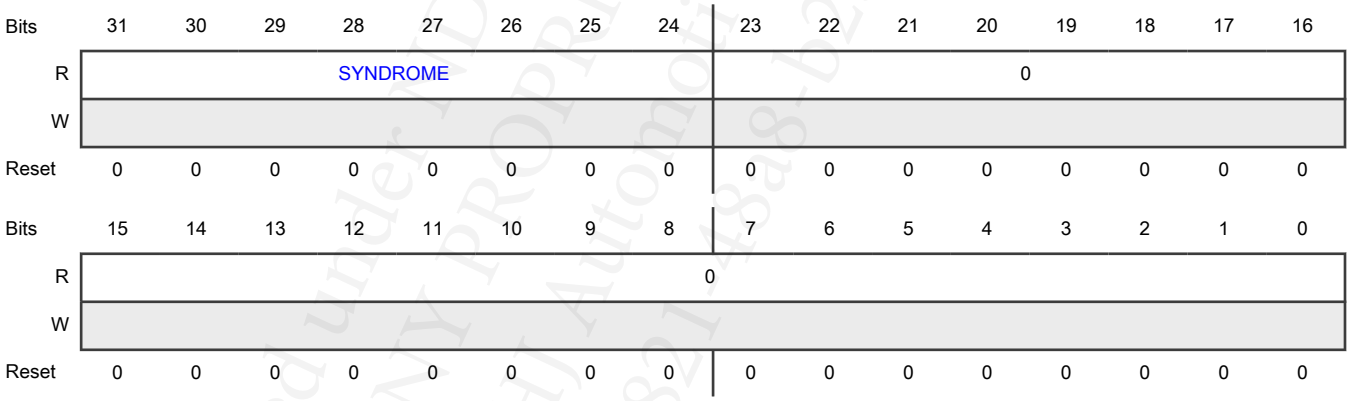
For a = 0 to 12:

Register	Offset
SYNa	104h + (a × 10h)

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.15.8 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT12)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								COUNT							
W	0								0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4626e-4821-42a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
—	
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.16 ERM_PFE2 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.16.1 ERM_PFE2 memory map

ERM_PFE2 base address: 4403_6000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
18h	ERM Status Register 2 (SR2)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT 9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_ CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_ CNT11)	32	ROWZ	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_ CNT12)	32	ROWZ	0000_0000h
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	RO	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	RO	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_ CNT13)	32	ROWZ	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	RO	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	RO	0000_0000h
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_ CNT14)	32	ROWZ	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	RO	0000_0000h
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	RO	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_ CNT15)	32	ROWZ	0000_0000h
200h	ERM Memory 16 Error Address Register (EAR16)	32	RO	0000_0000h
204h	ERM Memory 16 Syndrome Register (SYN16)	32	RO	0000_0000h
208h	ERM Memory 16 Correctable Error Count Register (CORR_ERR_ CNT16)	32	ROWZ	0000_0000h

Beijing CHANGJIANG Automotive Technology Co., Ltd.
c8c4e2e2-4871-4848-9251-39daab6246c1

65.16.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30	ENCIE0

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE0	Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19	ESCIE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE3	Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e2-4821-4888-b251-390d3b624c1

65.16.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE 8	ENCIE 8	0		ESCIE 9	ENCIE 9	0		ESCIE 10	ENCIE 10	0		ESCIE 11	ENCIE 11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE 12	ENCIE 12	0		ESCIE 13	ENCIE 13	0		ESCIE 14	ENCIE 14	0		ESCIE 15	ENCIE 15	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30	ENCIE8

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE8	Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19	ESCIE11

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE11	Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 single-bit correction events is disabled. 1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 non-correctable error events is disabled. 1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462c-4821-4888-b251-390d3b624c1

65.16.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE16	ENCIE16	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE16	ESCIE16 Enable Memory 16 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 single-bit correction events is disabled. 1b - Interrupt notification of Memory 16 single-bit correction events is enabled.
30 ENCIE16	ENCIE16 Enable Memory 16 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 non-correctable error events is disabled. 1b - Interrupt notification of Memory 16 non-correctable error events is enabled.
29-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

65.16.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22	NCE2

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE2	Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.16.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10

- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27	SBC9

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC9	Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23 SBC10	SBC10 Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled. 0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

65.16.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC16	NCE16	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CUJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC16	SBC16 Memory 16 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE16] is enabled. 0b - No single-bit correction event on Memory 16 detected. 1b - Single-bit correction event on Memory 16 detected.
30 NCE16	NCE16 Memory 16 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE16] is enabled. 0b - No non-correctable error event on Memory 16 detected. 1b - Non-correctable error event on Memory 16 detected.
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-590a3ab624c1

65.16.8 ERM Memory a Error Address Register (EAR0 - EAR16)

Offset

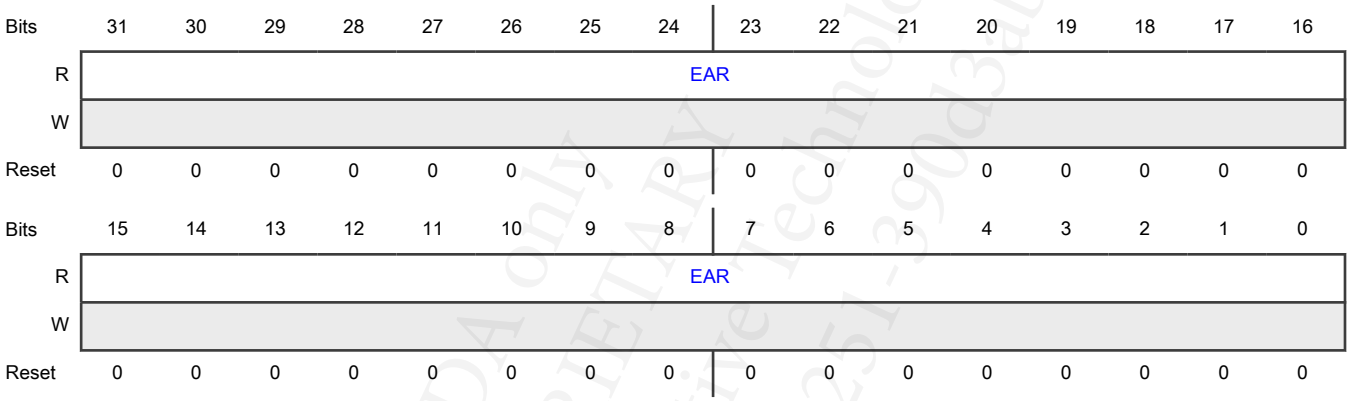
For a = 0 to 16:

Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.16.9 ERM Memory a Syndrome Register (SYN0 - SYN16)

Offset

For a = 0 to 16:

Register	Offset
SYNa	104h + (a × 10h)

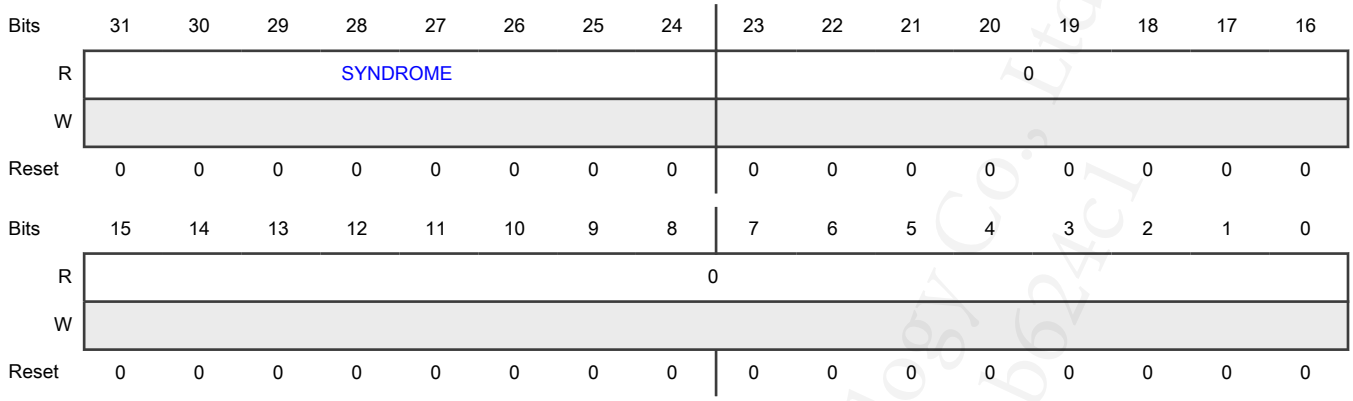
Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4021-48a8-b251-590a3b624c1

position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.16.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT 16)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h

Table continues on the next page...

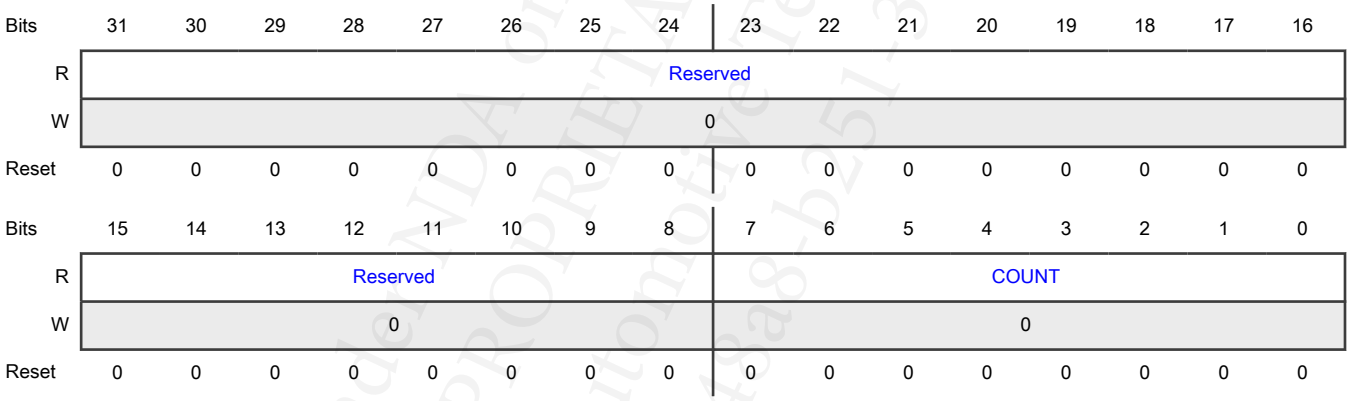
Table continued from the previous page...

Register	Offset
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT16	208h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8a6e5-4821-43a8-b251-3903ab624c1

65.17 ERM_PFE3 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.17.1 ERM_PFE3 memory map

ERM_PFE3 base address: 4403_7000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
18h	ERM Status Register 2 (SR2)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT 9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_ CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_ CNT11)	32	ROWZ	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_CNT12)	32	ROWZ	0000_0000h
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	RO	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	RO	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_CNT13)	32	ROWZ	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	RO	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	RO	0000_0000h
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_CNT14)	32	ROWZ	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	RO	0000_0000h
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	RO	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_CNT15)	32	ROWZ	0000_0000h
200h	ERM Memory 16 Error Address Register (EAR16)	32	RO	0000_0000h
204h	ERM Memory 16 Syndrome Register (SYN16)	32	RO	0000_0000h
208h	ERM Memory 16 Correctable Error Count Register (CORR_ERR_CNT16)	32	ROWZ	0000_0000h

65.17.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

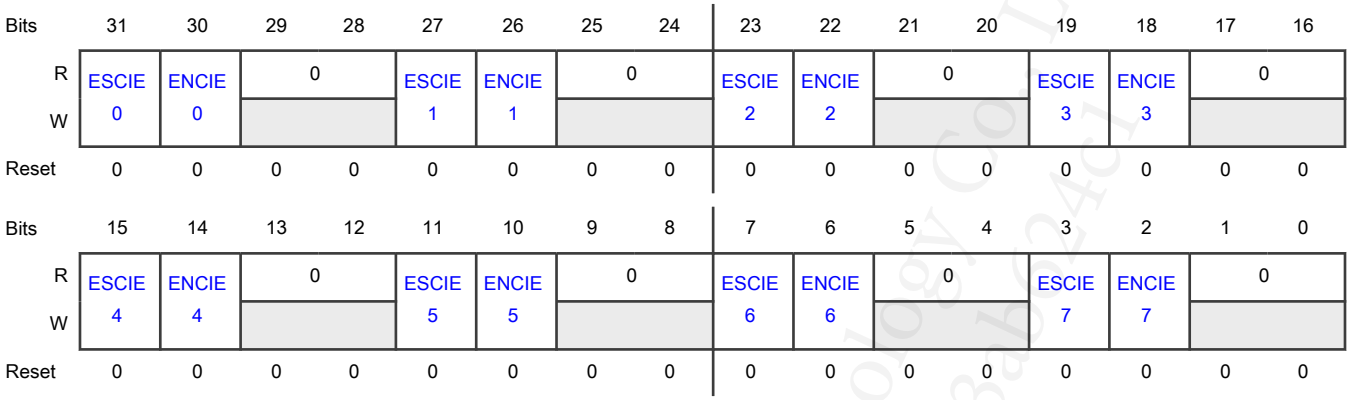
This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48ac-b251-390dab624c1

- Channel 5
- Channel 6
- Channel 7

Diagram



Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26	ENCIE1

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE1	Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15	ESCIE4

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE4	Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

65.17.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14

- Channel 15

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE8	ENCIE8	0		ESCIE9	ENCIE9	0		ESCIE10	ENCIE10	0		ESCIE11	ENCIE11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE12	ENCIE12	0		ESCIE13	ENCIE13	0		ESCIE14	ENCIE14	0		ESCIE15	ENCIE15	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 single-bit correction events is disabled. 1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 non-correctable error events is disabled. 1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0 —	Reserved

65.17.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

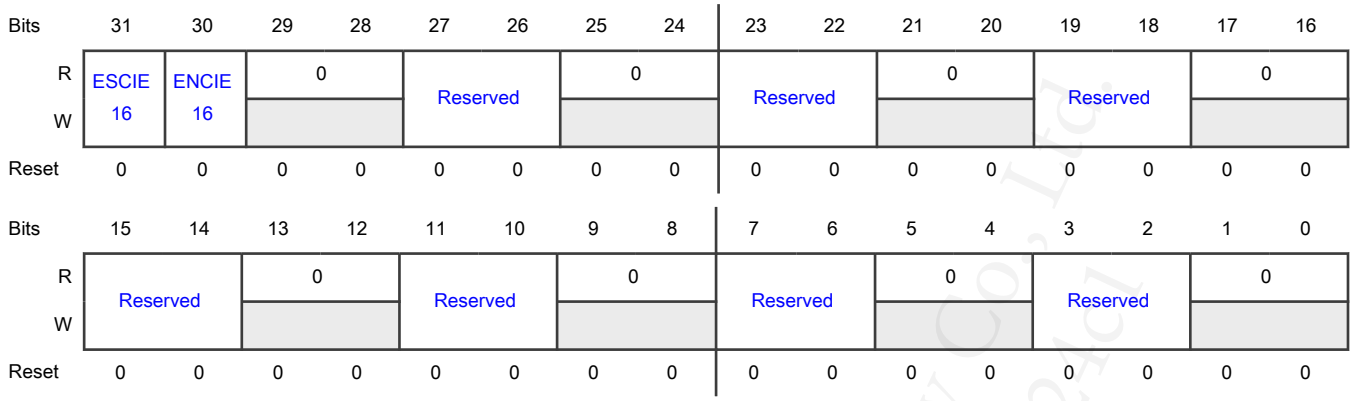
Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 16

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-90d3a1624c1

Diagram



Fields

Field	Function
31 ESCIE16	ESCIE16 Enable Memory 16 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 single-bit correction events is disabled. 1b - Interrupt notification of Memory 16 single-bit correction events is enabled.
30 ENCIE16	ENCIE16 Enable Memory 16 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 non-correctable error events is disabled. 1b - Interrupt notification of Memory 16 non-correctable error events is enabled.
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.17.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.17.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23 SBC10	SBC10 Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18	NCE11

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE11	Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled. 0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

65.17.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC16	NCE16	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC16	SBC16 Memory 16 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE16] is enabled. 0b - No single-bit correction event on Memory 16 detected. 1b - Single-bit correction event on Memory 16 detected.
30 NCE16	NCE16 Memory 16 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE16] is enabled. 0b - No non-correctable error event on Memory 16 detected. 1b - Non-correctable error event on Memory 16 detected.
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.17.8 ERM Memory a Error Address Register (EAR0 - EAR16)

Offset

For a = 0 to 16:

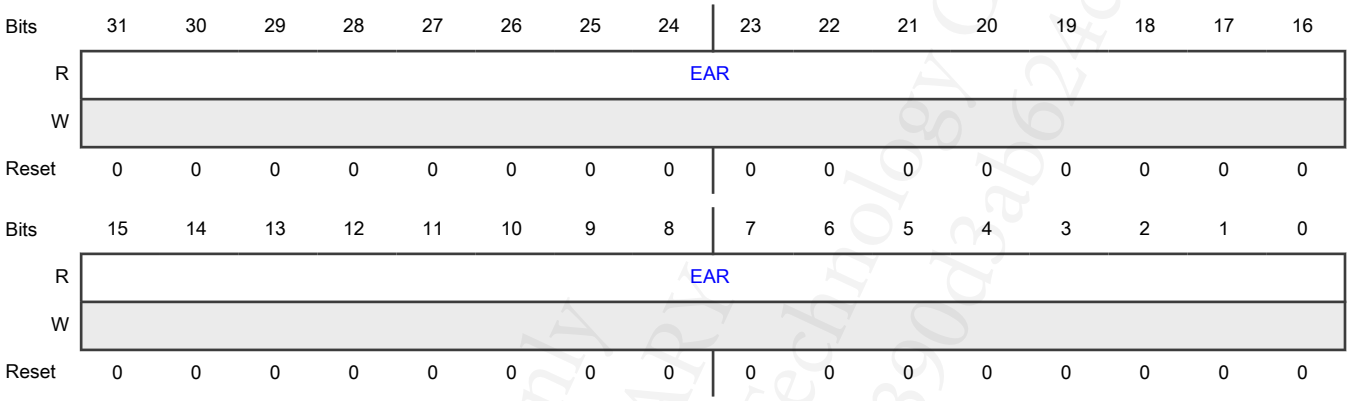
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.17.9 ERM Memory a Syndrome Register (SYN0 - SYN16)

Offset

For a = 0 to 16:

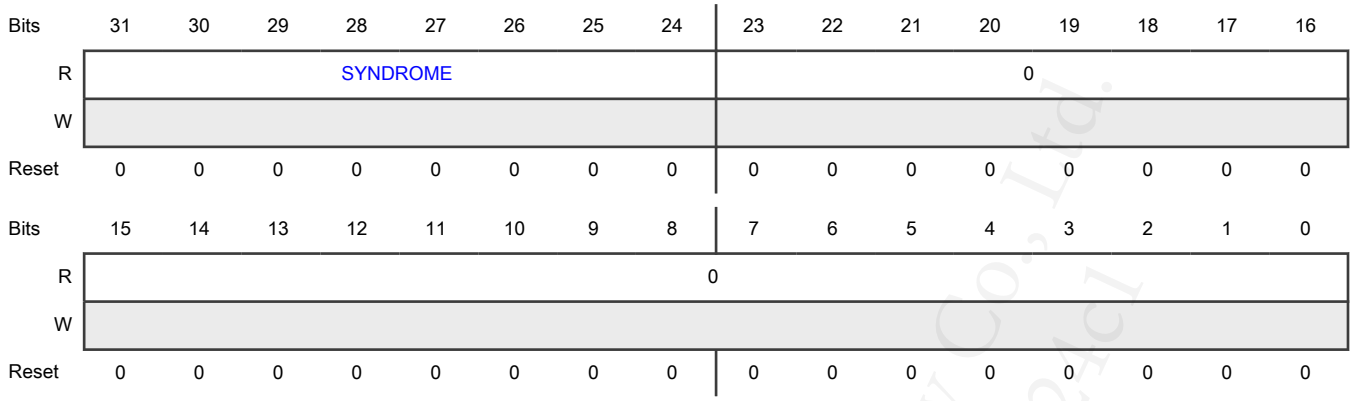
Register	Offset
SYNa	104h + (a × 10h)

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c2e5-4821-4a8-b251-390a3ab62fc1

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.17.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT16	208h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								COUNT							
W	0								0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.18 ERM_PFE4 register descriptions

You can access the programming model:

- Only in supervisor mode

- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.18.1 ERM_PFE4 memory map

ERM_PFE4 base address: 4403_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
18h	ERM Status Register 2 (SR2)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT 9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_ CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_ CNT11)	32	ROWZ	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_ CNT12)	32	ROWZ	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	RO	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	RO	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_CNT13)	32	ROWZ	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	RO	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	RO	0000_0000h
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_CNT14)	32	ROWZ	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	RO	0000_0000h
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	RO	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_CNT15)	32	ROWZ	0000_0000h
200h	ERM Memory 16 Error Address Register (EAR16)	32	RO	0000_0000h
204h	ERM Memory 16 Syndrome Register (SYN16)	32	RO	0000_0000h
208h	ERM Memory 16 Correctable Error Count Register (CORR_ERR_CNT16)	32	ROWZ	0000_0000h

65.18.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4d21-48a8-b251-390dab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

65.18.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-43a8-b251-390d3b624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE 8	ENCIE 8	0		ESCIE 9	ENCIE 9	0		ESCIE 10	ENCIE 10	0		ESCIE 11	ENCIE 11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE 12	ENCIE 12	0		ESCIE 13	ENCIE 13	0		ESCIE 14	ENCIE 14	0		ESCIE 15	ENCIE 15	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 single-bit correction events is disabled. 1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 non-correctable error events is disabled. 1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0 —	Reserved

65.18.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE 16	ENCIE 16	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca2a2e5-4921-48a8-b251-390a3b6240c1

Fields

Field	Function
31 ESCIE16	ESCIE16 Enable Memory 16 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 single-bit correction events is disabled. 1b - Interrupt notification of Memory 16 single-bit correction events is enabled.
30 ENCIE16	ENCIE16 Enable Memory 16 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 non-correctable error events is disabled. 1b - Interrupt notification of Memory 16 non-correctable error events is enabled.
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390a3b6240c1

Table continued from the previous page...

Field	Function
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.18.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18	NCE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE3	Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.18.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESUI Automotive Technology Co., Ltd.
ca6b625-1821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23	SBC10

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC10	Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled. 0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a3-b251-390d3ab624c1

65.18.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC16	NCE16	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC16	SBC16 Memory 16 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE16] is enabled. 0b - No single-bit correction event on Memory 16 detected. 1b - Single-bit correction event on Memory 16 detected.
30 NCE16	NCE16 Memory 16 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE16] is enabled. 0b - No non-correctable error event on Memory 16 detected. 1b - Non-correctable error event on Memory 16 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.18.8 ERM Memory a Error Address Register (EAR0 - EAR16)

Offset

For a = 0 to 16:

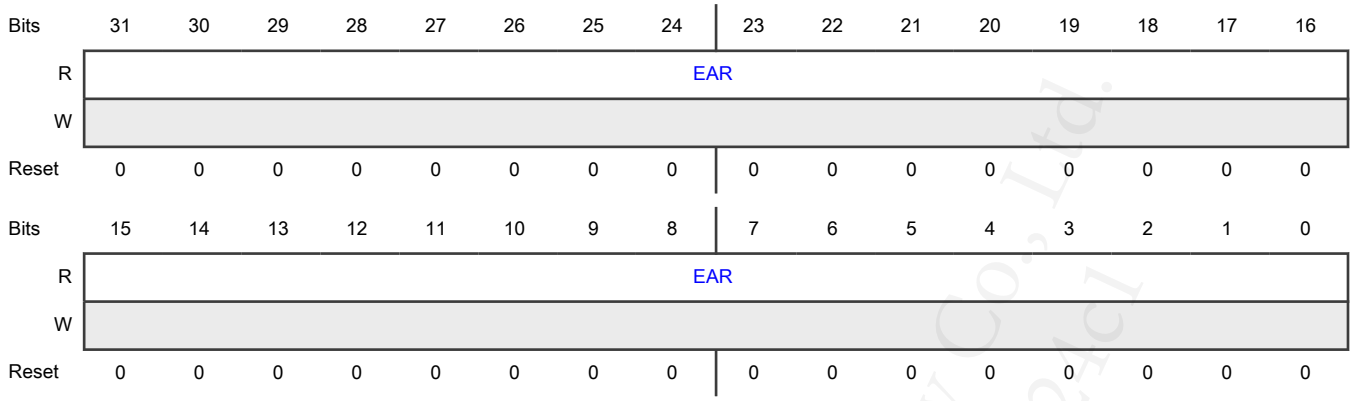
Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.18.9 ERM Memory a Syndrome Register (SYN0 - SYN16)

Offset

For a = 0 to 16:

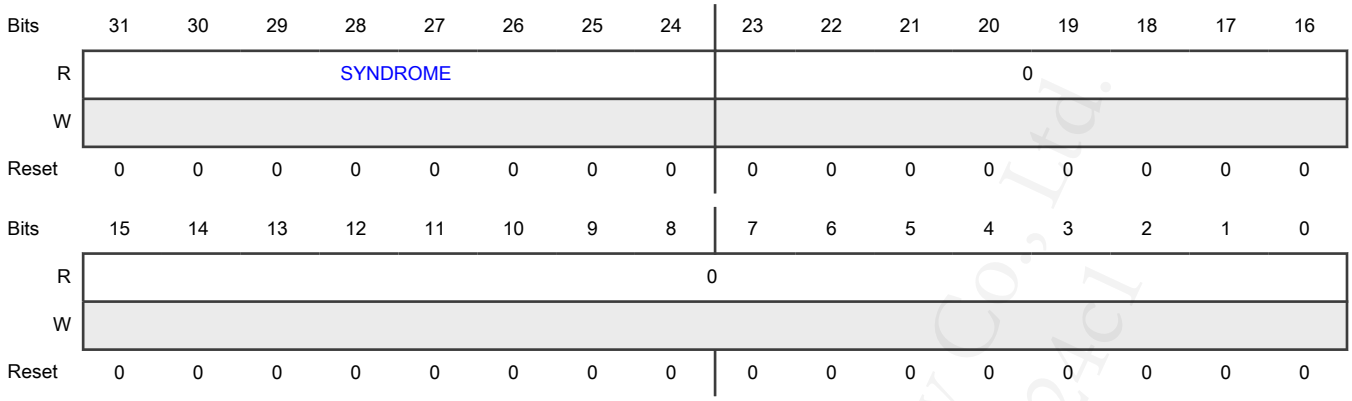
Register	Offset
SYNa	104h + (a × 10h)

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.18.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h

Table continues on the next page...

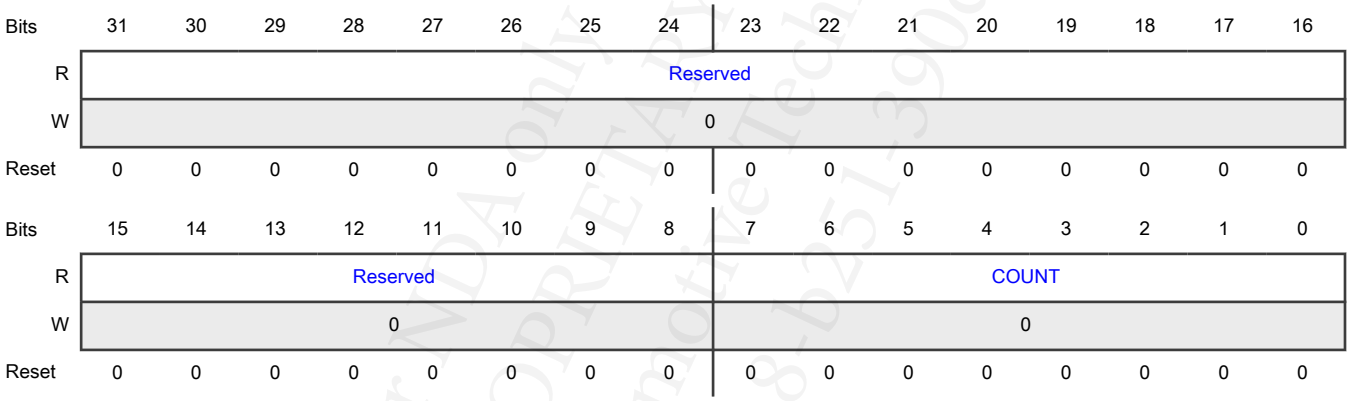
Table continued from the previous page...

Register	Offset
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT16	208h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.19 ERM_PFE5 register descriptions

You can access the programming model:

- Only in supervisor mode

- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.19.1 ERM_PFE5 memory map

ERM_PFE5 base address: 4403_9000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
18h	ERM Status Register 2 (SR2)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_CNT11)	32	ROWZ	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_CNT12)	32	ROWZ	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	RO	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	RO	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_CNT13)	32	ROWZ	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	RO	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	RO	0000_0000h
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_CNT14)	32	ROWZ	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	RO	0000_0000h
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	RO	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_CNT15)	32	ROWZ	0000_0000h
200h	ERM Memory 16 Error Address Register (EAR16)	32	RO	0000_0000h
204h	ERM Memory 16 Syndrome Register (SYN16)	32	RO	0000_0000h
208h	ERM Memory 16 Correctable Error Count Register (CORR_ERR_CNT16)	32	ROWZ	0000_0000h

65.19.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4d21-48a8-b251-390dab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

65.19.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-43a8-b251-390d3b624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE 8	ENCIE 8	0		ESCIE 9	ENCIE 9	0		ESCIE 10	ENCIE 10	0		ESCIE 11	ENCIE 11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE 12	ENCIE 12	0		ESCIE 13	ENCIE 13	0		ESCIE 14	ENCIE 14	0		ESCIE 15	ENCIE 15	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 single-bit correction events is disabled. 1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 non-correctable error events is disabled. 1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0 —	Reserved

65.19.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE16	ENCIE16	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca2a2e5-4921-48a8-b251-390a3b6240c1

Fields

Field	Function
31 ESCIE16	ESCIE16 Enable Memory 16 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 single-bit correction events is disabled. 1b - Interrupt notification of Memory 16 single-bit correction events is enabled.
30 ENCIE16	ENCIE16 Enable Memory 16 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 non-correctable error events is disabled. 1b - Interrupt notification of Memory 16 non-correctable error events is enabled.
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390a3b6240c1

Table continued from the previous page...

Field	Function
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.19.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18	NCE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE3	Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.19.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESUI Automotive Technology Co., Ltd.
ca6b625-1821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23	SBC10

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC10	Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled. 0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing ChiUtoInfotive Technology Co., Ltd.
c8c462e5-4821-48a3-b251-390d3ab624c1

65.19.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC16	NCE16	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC16	SBC16 Memory 16 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE16] is enabled. 0b - No single-bit correction event on Memory 16 detected. 1b - Single-bit correction event on Memory 16 detected.
30 NCE16	NCE16 Memory 16 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE16] is enabled. 0b - No non-correctable error event on Memory 16 detected. 1b - Non-correctable error event on Memory 16 detected.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c46e5-5432-48a8-b251-3903ab624c1

Table continued from the previous page...

Field	Function
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.19.8 ERM Memory a Error Address Register (EAR0 - EAR16)

Offset

For a = 0 to 16:

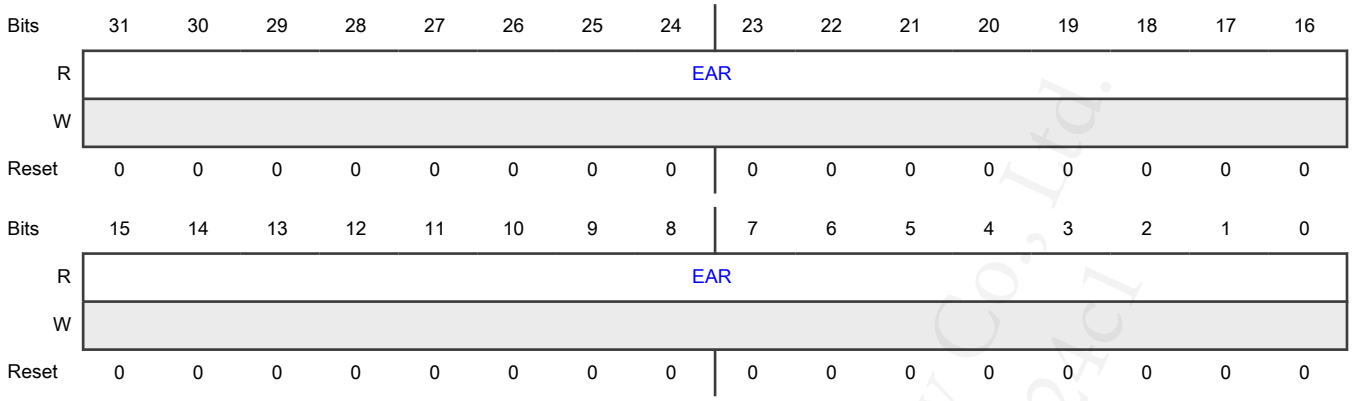
Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.19.9 ERM Memory a Syndrome Register (SYN0 - SYN16)

Offset

For a = 0 to 16:

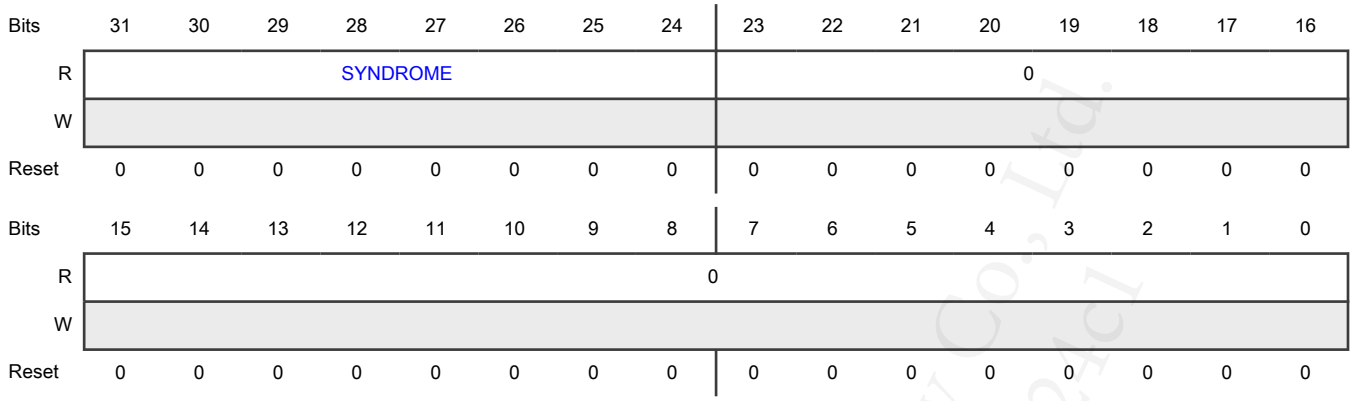
Register	Offset
SYNa	104h + (a × 10h)

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.19.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT16	208h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								COUNT							
W	0								0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.20 ERM_PFE6 register descriptions

You can access the programming model:

- Only in supervisor mode

- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.20.1 ERM_PFE6 memory map

ERM_PFE6 base address: 4403_A000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
18h	ERM Status Register 2 (SR2)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT 9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_ CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_ CNT11)	32	ROWZ	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_ CNT12)	32	ROWZ	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	RO	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	RO	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_CNT13)	32	ROWZ	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	RO	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	RO	0000_0000h
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_CNT14)	32	ROWZ	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	RO	0000_0000h
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	RO	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_CNT15)	32	ROWZ	0000_0000h
200h	ERM Memory 16 Error Address Register (EAR16)	32	RO	0000_0000h
204h	ERM Memory 16 Syndrome Register (SYN16)	32	RO	0000_0000h
208h	ERM Memory 16 Correctable Error Count Register (CORR_ERR_CNT16)	32	ROWZ	0000_0000h

65.20.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4a21-48a8-b251-390dab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

65.20.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-43a8-b251-390d3b624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE8	ENCIE8	0		ESCIE9	ENCIE9	0		ESCIE10	ENCIE10	0		ESCIE11	ENCIE11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE12	ENCIE12	0		ESCIE13	ENCIE13	0		ESCIE14	ENCIE14	0		ESCIE15	ENCIE15	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 single-bit correction events is disabled. 1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 non-correctable error events is disabled. 1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0 —	Reserved

65.20.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE16	ENCIE16	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca2a2e5-4921-48a8-b251-390a3b6240c1

Fields

Field	Function
31 ESCIE16	ESCIE16 Enable Memory 16 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 single-bit correction events is disabled. 1b - Interrupt notification of Memory 16 single-bit correction events is enabled.
30 ENCIE16	ENCIE16 Enable Memory 16 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 non-correctable error events is disabled. 1b - Interrupt notification of Memory 16 non-correctable error events is enabled.
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390a3b6240c1

Table continued from the previous page...

Field	Function
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.20.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18	NCE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE3	Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.20.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESUI Automotive Technology Co., Ltd.
ca6625e5-1821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23	SBC10

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC10	Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled. 0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESUN Automotive Technology Co., Ltd.
c8c462e5-4821-48a3-b251-390d3ab624c1

65.20.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC16	NCE16	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC16	SBC16 Memory 16 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE16] is enabled. 0b - No single-bit correction event on Memory 16 detected. 1b - Single-bit correction event on Memory 16 detected.
30 NCE16	NCE16 Memory 16 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE16] is enabled. 0b - No non-correctable error event on Memory 16 detected. 1b - Non-correctable error event on Memory 16 detected.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c46b5c5-4321-48a8-b251-3903a624c1

Table continued from the previous page...

Field	Function
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.20.8 ERM Memory a Error Address Register (EAR0 - EAR16)

Offset

For a = 0 to 16:

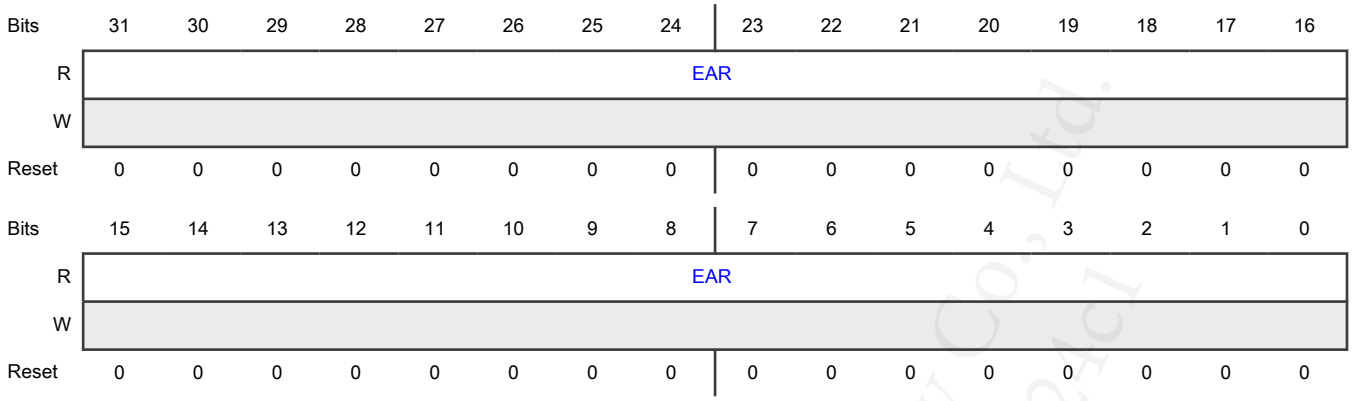
Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.20.9 ERM Memory a Syndrome Register (SYN0 - SYN16)

Offset

For a = 0 to 16:

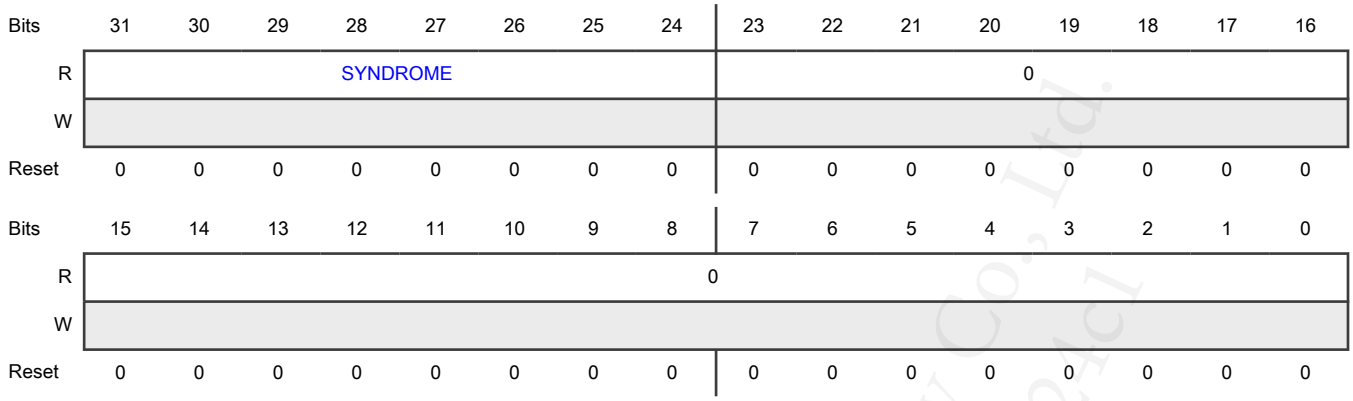
Register	Offset
SYNa	104h + (a × 10h)

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.20.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT16	208h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								COUNT							
W	0								0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.21 ERM_PFE7 register descriptions

You can access the programming model:

- Only in supervisor mode

- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.21.1 ERM_PFE7 memory map

ERM_PFE7 base address: 4403_B000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
18h	ERM Status Register 2 (SR2)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT 9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_ CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_ CNT11)	32	ROWZ	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_ CNT12)	32	ROWZ	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	RO	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	RO	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_CNT13)	32	ROWZ	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	RO	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	RO	0000_0000h
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_CNT14)	32	ROWZ	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	RO	0000_0000h
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	RO	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_CNT15)	32	ROWZ	0000_0000h
200h	ERM Memory 16 Error Address Register (EAR16)	32	RO	0000_0000h
204h	ERM Memory 16 Syndrome Register (SYN16)	32	RO	0000_0000h
208h	ERM Memory 16 Correctable Error Count Register (CORR_ERR_CNT16)	32	ROWZ	0000_0000h

65.21.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4d21-48a8-b251-390dab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

65.21.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3b624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE8	ENCIE8	0		ESCIE9	ENCIE9	0		ESCIE10	ENCIE10	0		ESCIE11	ENCIE11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE12	ENCIE12	0		ESCIE13	ENCIE13	0		ESCIE14	ENCIE14	0		ESCIE15	ENCIE15	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 single-bit correction events is disabled. 1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 non-correctable error events is disabled. 1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0 —	Reserved

65.21.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE16	ENCIE16	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca2a2e5-4921-48a8-b251-390a3b6240c1

Fields

Field	Function
31 ESCIE16	ESCIE16 Enable Memory 16 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 single-bit correction events is disabled. 1b - Interrupt notification of Memory 16 single-bit correction events is enabled.
30 ENCIE16	ENCIE16 Enable Memory 16 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 non-correctable error events is disabled. 1b - Interrupt notification of Memory 16 non-correctable error events is enabled.
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390a3b6240c1

Table continued from the previous page...

Field	Function
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.21.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18	NCE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE3	Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.21.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESUI Automotive Technology Co., Ltd.
ca6625e5-1821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23	SBC10

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC10	Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled. 0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing ChiUtoInfotive Technology Co., Ltd.
c8c462e5-4821-48a3-b251-390d3ab624c1

65.21.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC16	NCE16	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC16	SBC16 Memory 16 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE16] is enabled. 0b - No single-bit correction event on Memory 16 detected. 1b - Single-bit correction event on Memory 16 detected.
30 NCE16	NCE16 Memory 16 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE16] is enabled. 0b - No non-correctable error event on Memory 16 detected. 1b - Non-correctable error event on Memory 16 detected.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c46b5c5-4321-48a8-b251-5903a3624c1

Table continued from the previous page...

Field	Function
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.21.8 ERM Memory a Error Address Register (EAR0 - EAR16)

Offset

For a = 0 to 16:

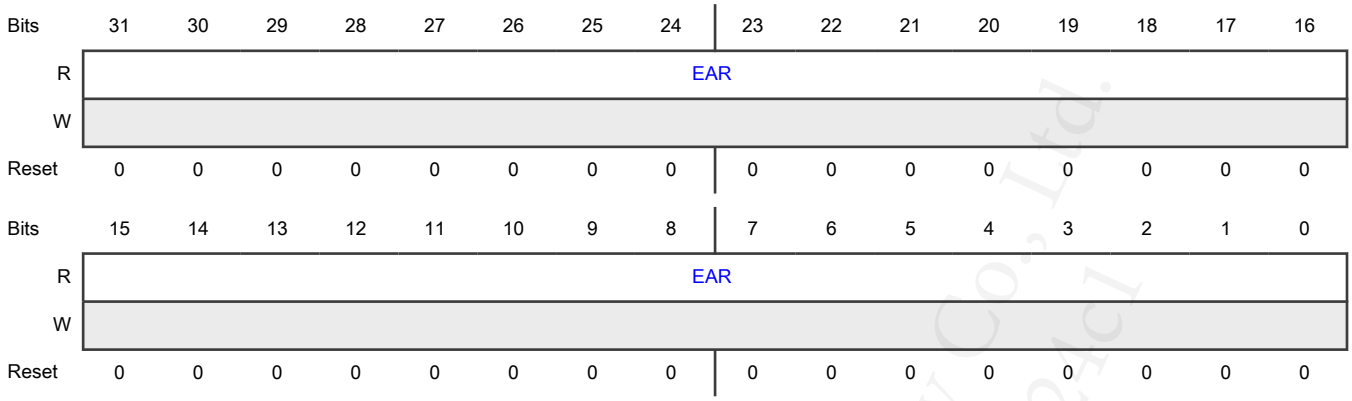
Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.21.9 ERM Memory a Syndrome Register (SYN0 - SYN16)

Offset

For a = 0 to 16:

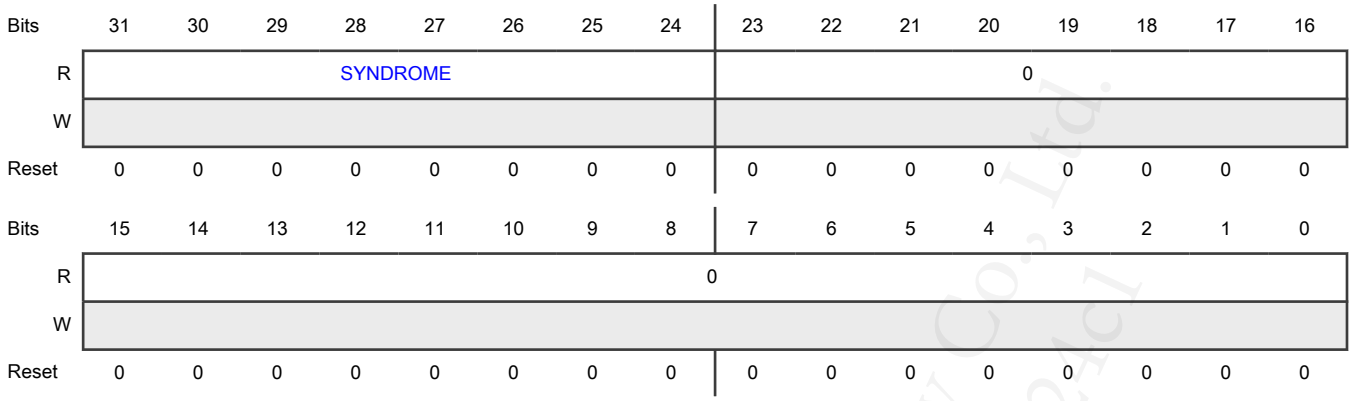
Register	Offset
SYNa	104h + (a × 10h)

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.21.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h

Table continues on the next page...

Table continued from the previous page...

Register	Offset
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT16	208h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								COUNT							
W	0								0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.22 ERM_PFE8 register descriptions

You can access the programming model:

- Only in supervisor mode

- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.22.1 ERM_PFE8 memory map

ERM_PFE8 base address: 4403_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
18h	ERM Status Register 2 (SR2)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT 9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_ CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_ CNT11)	32	ROWZ	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_ CNT12)	32	ROWZ	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	RO	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	RO	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_CNT13)	32	ROWZ	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	RO	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	RO	0000_0000h
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_CNT14)	32	ROWZ	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	RO	0000_0000h
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	RO	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_CNT15)	32	ROWZ	0000_0000h
200h	ERM Memory 16 Error Address Register (EAR16)	32	RO	0000_0000h
204h	ERM Memory 16 Syndrome Register (SYN16)	32	RO	0000_0000h
208h	ERM Memory 16 Correctable Error Count Register (CORR_ERR_CNT16)	32	ROWZ	0000_0000h

65.22.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4d21-48a8-b251-390dab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

65.22.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-43a8-b251-390d3b624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE 8	ENCIE 8	0		ESCIE 9	ENCIE 9	0		ESCIE 10	ENCIE 10	0		ESCIE 11	ENCIE 11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE 12	ENCIE 12	0		ESCIE 13	ENCIE 13	0		ESCIE 14	ENCIE 14	0		ESCIE 15	ENCIE 15	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 single-bit correction events is disabled. 1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 non-correctable error events is disabled. 1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0 —	Reserved

65.22.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE16	ENCIE16	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca2a2e5-4921-4a8-b251-390a3b6240

Fields

Field	Function
31 ESCIE16	ESCIE16 Enable Memory 16 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 single-bit correction events is disabled. 1b - Interrupt notification of Memory 16 single-bit correction events is enabled.
30 ENCIE16	ENCIE16 Enable Memory 16 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 non-correctable error events is disabled. 1b - Interrupt notification of Memory 16 non-correctable error events is enabled.
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390a3b6240c1

Table continued from the previous page...

Field	Function
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.22.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18	NCE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE3	Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.22.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESUI Automotive Technology Co., Ltd.
ca6b625-1821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23	SBC10

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC10	Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled. 0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing ChiUtoInfotive Technology Co., Ltd.
c8c462e5-4821-48a3-b251-390d3ab624c1

65.22.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC16	NCE16	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC16	SBC16 Memory 16 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE16] is enabled. 0b - No single-bit correction event on Memory 16 detected. 1b - Single-bit correction event on Memory 16 detected.
30 NCE16	NCE16 Memory 16 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE16] is enabled. 0b - No non-correctable error event on Memory 16 detected. 1b - Non-correctable error event on Memory 16 detected.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c46b55-5432-48a8-b251-3903ab624c1

Table continued from the previous page...

Field	Function
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.22.8 ERM Memory a Error Address Register (EAR0 - EAR16)

Offset

For a = 0 to 16:

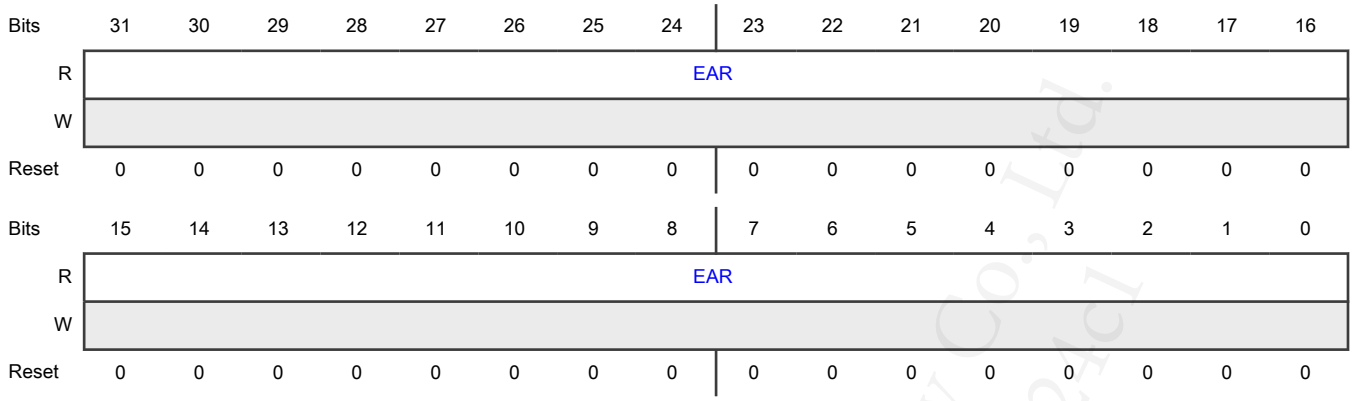
Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.22.9 ERM Memory a Syndrome Register (SYN0 - SYN16)

Offset

For a = 0 to 16:

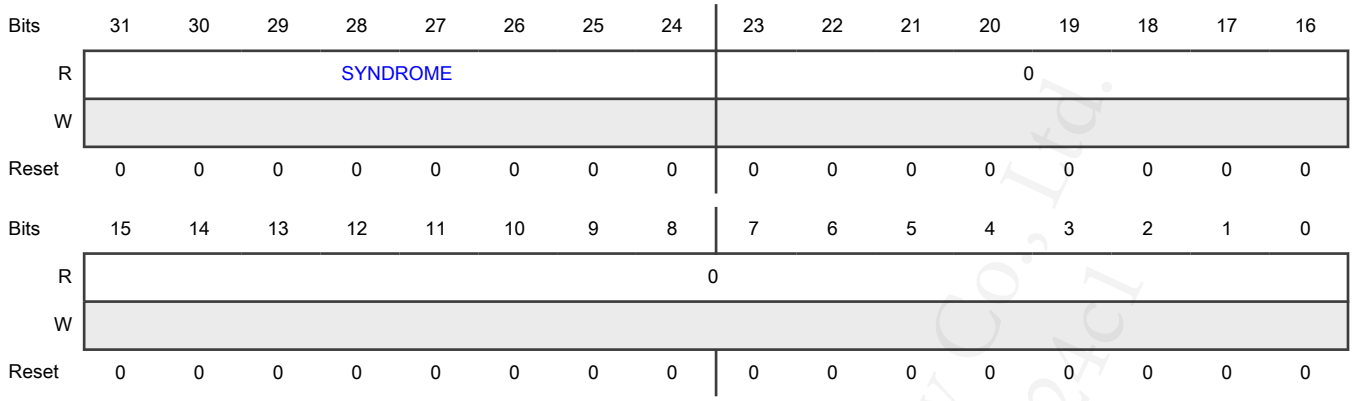
Register	Offset
SYNa	104h + (a × 10h)

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-48a8-b251-390d8ab624c1

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.22.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h

Table continues on the next page...

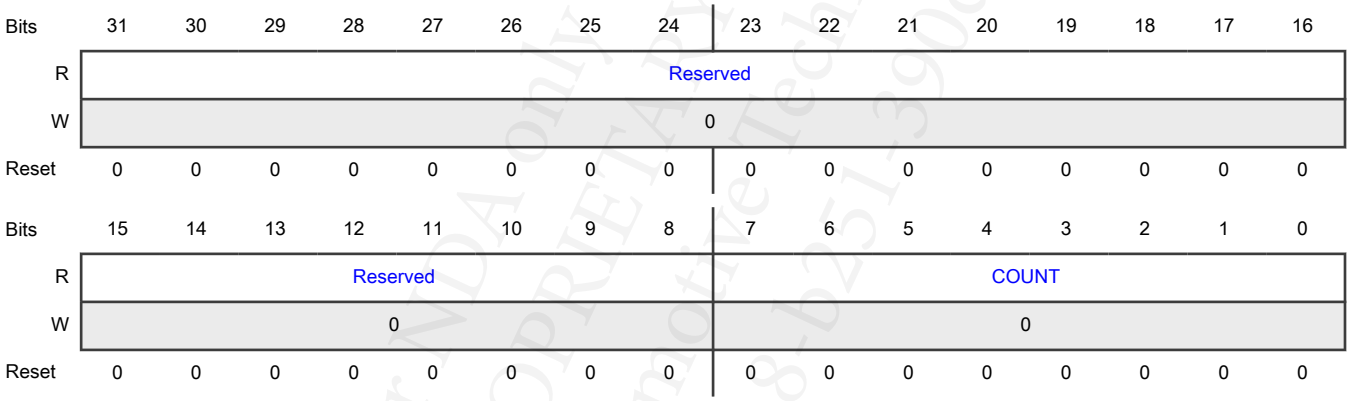
Table continued from the previous page...

Register	Offset
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT16	208h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.23 ERM_PFE9 register descriptions

You can access the programming model:

- Only in supervisor mode

- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.23.1 ERM_PFE9 memory map

ERM_PFE9 base address: 4403_D000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
8h	ERM Configuration Register 2 (CR2)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
18h	ERM Status Register 2 (SR2)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
120h	ERM Memory 2 Error Address Register (EAR2)	32	RO	0000_0000h
124h	ERM Memory 2 Syndrome Register (SYN2)	32	RO	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
170h	ERM Memory 7 Error Address Register (EAR7)	32	RO	0000_0000h
174h	ERM Memory 7 Syndrome Register (SYN7)	32	RO	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
180h	ERM Memory 8 Error Address Register (EAR8)	32	RO	0000_0000h
184h	ERM Memory 8 Syndrome Register (SYN8)	32	RO	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h
190h	ERM Memory 9 Error Address Register (EAR9)	32	RO	0000_0000h
194h	ERM Memory 9 Syndrome Register (SYN9)	32	RO	0000_0000h
198h	ERM Memory 9 Correctable Error Count Register (CORR_ERR_CNT 9)	32	ROWZ	0000_0000h
1A0h	ERM Memory 10 Error Address Register (EAR10)	32	RO	0000_0000h
1A4h	ERM Memory 10 Syndrome Register (SYN10)	32	RO	0000_0000h
1A8h	ERM Memory 10 Correctable Error Count Register (CORR_ERR_ CNT10)	32	ROWZ	0000_0000h
1B0h	ERM Memory 11 Error Address Register (EAR11)	32	RO	0000_0000h
1B4h	ERM Memory 11 Syndrome Register (SYN11)	32	RO	0000_0000h
1B8h	ERM Memory 11 Correctable Error Count Register (CORR_ERR_ CNT11)	32	ROWZ	0000_0000h
1C0h	ERM Memory 12 Error Address Register (EAR12)	32	RO	0000_0000h
1C4h	ERM Memory 12 Syndrome Register (SYN12)	32	RO	0000_0000h
1C8h	ERM Memory 12 Correctable Error Count Register (CORR_ERR_ CNT12)	32	ROWZ	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1D0h	ERM Memory 13 Error Address Register (EAR13)	32	RO	0000_0000h
1D4h	ERM Memory 13 Syndrome Register (SYN13)	32	RO	0000_0000h
1D8h	ERM Memory 13 Correctable Error Count Register (CORR_ERR_CNT13)	32	ROWZ	0000_0000h
1E0h	ERM Memory 14 Error Address Register (EAR14)	32	RO	0000_0000h
1E4h	ERM Memory 14 Syndrome Register (SYN14)	32	RO	0000_0000h
1E8h	ERM Memory 14 Correctable Error Count Register (CORR_ERR_CNT14)	32	ROWZ	0000_0000h
1F0h	ERM Memory 15 Error Address Register (EAR15)	32	RO	0000_0000h
1F4h	ERM Memory 15 Syndrome Register (SYN15)	32	RO	0000_0000h
1F8h	ERM Memory 15 Correctable Error Count Register (CORR_ERR_CNT15)	32	ROWZ	0000_0000h
200h	ERM Memory 16 Error Address Register (EAR16)	32	RO	0000_0000h
204h	ERM Memory 16 Syndrome Register (SYN16)	32	RO	0000_0000h
208h	ERM Memory 16 Correctable Error Count Register (CORR_ERR_CNT16)	32	ROWZ	0000_0000h

65.23.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4d21-48a8-b251-390dab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		ESCIE1	ENCIE1	0		ESCIE2	ENCIE2	0		ESCIE3	ENCIE3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE4	ENCIE4	0		ESCIE5	ENCIE5	0		ESCIE6	ENCIE6	0		ESCIE7	ENCIE7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE3	ESCIE3 Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

65.23.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-43a8-b251-390d3b624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE 8	ENCIE 8	0		ESCIE 9	ENCIE 9	0		ESCIE 10	ENCIE 10	0		ESCIE 11	ENCIE 11	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE 12	ENCIE 12	0		ESCIE 13	ENCIE 13	0		ESCIE 14	ENCIE 14	0		ESCIE 15	ENCIE 15	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE9	ESCIE9 Enable Memory 9 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 single-bit correction events is disabled. 1b - Interrupt notification of Memory 9 single-bit correction events is enabled.
26 ENCIE9	ENCIE9 Enable Memory 9 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 9 non-correctable error events is disabled. 1b - Interrupt notification of Memory 9 non-correctable error events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
25-24 —	Reserved
23 ESCIE10	ESCIE10 Enable Memory 10 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 single-bit correction events is disabled. 1b - Interrupt notification of Memory 10 single-bit correction events is enabled.
22 ENCIE10	ENCIE10 Enable Memory 10 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 10 non-correctable error events is disabled. 1b - Interrupt notification of Memory 10 non-correctable error events is enabled.
21-20 —	Reserved
19 ESCIE11	ESCIE11 Enable Memory 11 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 single-bit correction events is disabled. 1b - Interrupt notification of Memory 11 single-bit correction events is enabled.
18 ENCIE11	ENCIE11 Enable Memory 11 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 11 non-correctable error events is disabled. 1b - Interrupt notification of Memory 11 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE12	ESCIE12 Enable Memory 12 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 single-bit correction events is disabled. 1b - Interrupt notification of Memory 12 single-bit correction events is enabled.

Table continues on the next page...

Table continued from the previous page...

Field	Function
14 ENCIE12	ENCIE12 Enable Memory 12 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 12 non-correctable error events is disabled. 1b - Interrupt notification of Memory 12 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE13	ESCIE13 Enable Memory 13 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 single-bit correction events is disabled. 1b - Interrupt notification of Memory 13 single-bit correction events is enabled.
10 ENCIE13	ENCIE13 Enable Memory 13 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 13 non-correctable error events is disabled. 1b - Interrupt notification of Memory 13 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE14	ESCIE14 Enable Memory 14 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 single-bit correction events is disabled. 1b - Interrupt notification of Memory 14 single-bit correction events is enabled.
6 ENCIE14	ENCIE14 Enable Memory 14 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 14 non-correctable error events is disabled. 1b - Interrupt notification of Memory 14 non-correctable error events is enabled.
5-4 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
3 ESCIE15	ESCIE15 Enable Memory 15 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 single-bit correction events is disabled. 1b - Interrupt notification of Memory 15 single-bit correction events is enabled.
2 ENCIE15	ENCIE15 Enable Memory 15 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 15 non-correctable error events is disabled. 1b - Interrupt notification of Memory 15 non-correctable error events is enabled.
1-0 —	Reserved

65.23.4 ERM Configuration Register 2 (CR2)

Offset

Register	Offset
CR2	8h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE 16	ENCIE 16	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing eMU Automotive Technology Co., Ltd.
ca2a2e5-4921-48a8-b251-390a3b6240c1

Fields

Field	Function
31 ESCIE16	ESCIE16 Enable Memory 16 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 single-bit correction events is disabled. 1b - Interrupt notification of Memory 16 single-bit correction events is enabled.
30 ENCIE16	ENCIE16 Enable Memory 16 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 16 non-correctable error events is disabled. 1b - Interrupt notification of Memory 16 non-correctable error events is enabled.
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390a3b6240c1

Table continued from the previous page...

Field	Function
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.23.5 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22 NCE2	NCE2 Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18	NCE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE3	Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.23.6 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8
- Channel 9
- Channel 10
- Channel 11
- Channel 12
- Channel 13
- Channel 14
- Channel 15

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		SBC9	NCE9	0		SBC10	NCE10	0		SBC11	NCE11	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC12	NCE12	0		SBC13	NCE13	0		SBC14	NCE14	0		SBC15	NCE15	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESUI Automotive Technology Co., Ltd.
ca6b625-1821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27 SBC9	SBC9 Memory 9 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE9] is enabled. 0b - No single-bit correction event on Memory 9 detected. 1b - Single-bit correction event on Memory 9 detected.
26 NCE9	NCE9 Memory 9 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE9] is enabled. 0b - No non-correctable error event on Memory 9 detected. 1b - Non-correctable error event on Memory 9 detected.
25-24 —	Reserved
23	SBC10

Table continues on the next page...

Table continued from the previous page...

Field	Function
SBC10	Memory 10 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE10] is enabled. 0b - No single-bit correction event on Memory 10 detected. 1b - Single-bit correction event on Memory 10 detected.
22 NCE10	NCE10 Memory 10 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE10] is enabled. 0b - No non-correctable error event on Memory 10 detected. 1b - Non-correctable error event on Memory 10 detected.
21-20 —	Reserved
19 SBC11	SBC11 Memory 11 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE11] is enabled. 0b - No single-bit correction event on Memory 11 detected. 1b - Single-bit correction event on Memory 11 detected.
18 NCE11	NCE11 Memory 11 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE11] is enabled. 0b - No non-correctable error event on Memory 11 detected. 1b - Non-correctable error event on Memory 11 detected.
17-16 —	Reserved
15 SBC12	SBC12 Memory 12 Single-Bit Correction Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE12] is enabled. 0b - No single-bit correction event on Memory 12 detected. 1b - Single-bit correction event on Memory 12 detected.
14 NCE12	NCE12 Memory 12 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE12] is enabled. 0b - No non-correctable error event on Memory 12 detected. 1b - Non-correctable error event on Memory 12 detected.
13-12 —	Reserved
11 SBC13	SBC13 Memory 13 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE13] is enabled. 0b - No single-bit correction event on Memory 13 detected. 1b - Single-bit correction event on Memory 13 detected.
10 NCE13	NCE13 Memory 13 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE13] is enabled. 0b - No non-correctable error event on Memory 13 detected. 1b - Non-correctable error event on Memory 13 detected.
9-8 —	Reserved
7 SBC14	SBC14 Memory 14 Single-Bit Correction Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE14] is enabled. 0b - No single-bit correction event on Memory 14 detected. 1b - Single-bit correction event on Memory 14 detected.
6 NCE14	NCE14 Memory 14 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE14] is enabled. 0b - No non-correctable error event on Memory 14 detected. 1b - Non-correctable error event on Memory 14 detected.
5-4 —	Reserved
3 SBC15	SBC15 Memory 15 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE15] is enabled. 0b - No single-bit correction event on Memory 15 detected. 1b - Single-bit correction event on Memory 15 detected.
2 NCE15	NCE15 Memory 15 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE15] is enabled. 0b - No non-correctable error event on Memory 15 detected. 1b - Non-correctable error event on Memory 15 detected.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing ChiUtoInfotive Technology Co., Ltd.
c8c462e5-4821-48a3-b251-390d3ab624c1

65.23.7 ERM Status Register 2 (SR2)

Offset

Register	Offset
SR2	18h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 16

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC16	NCE16	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC16	SBC16 Memory 16 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ESCIE16] is enabled. 0b - No single-bit correction event on Memory 16 detected. 1b - Single-bit correction event on Memory 16 detected.
30 NCE16	NCE16 Memory 16 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR2[ENCIE16] is enabled. 0b - No non-correctable error event on Memory 16 detected. 1b - Non-correctable error event on Memory 16 detected.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c46b5c5-4321-48a8-b251-3903a624c1

Table continued from the previous page...

Field	Function
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.23.8 ERM Memory a Error Address Register (EAR0 - EAR16)

Offset

For a = 0 to 16:

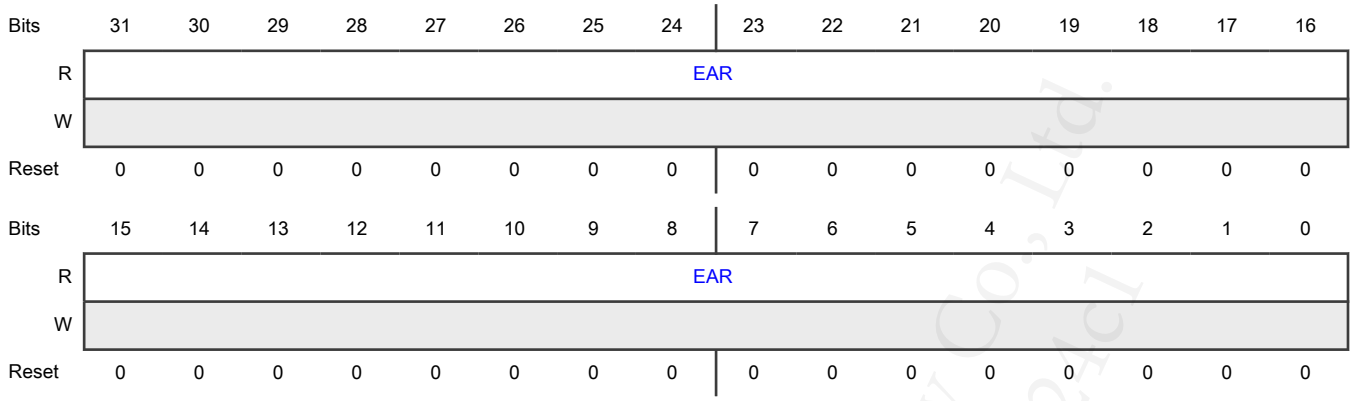
Register	Offset
EARa	100h + (a × 10h)

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.23.9 ERM Memory a Syndrome Register (SYN0 - SYN16)

Offset

For a = 0 to 16:

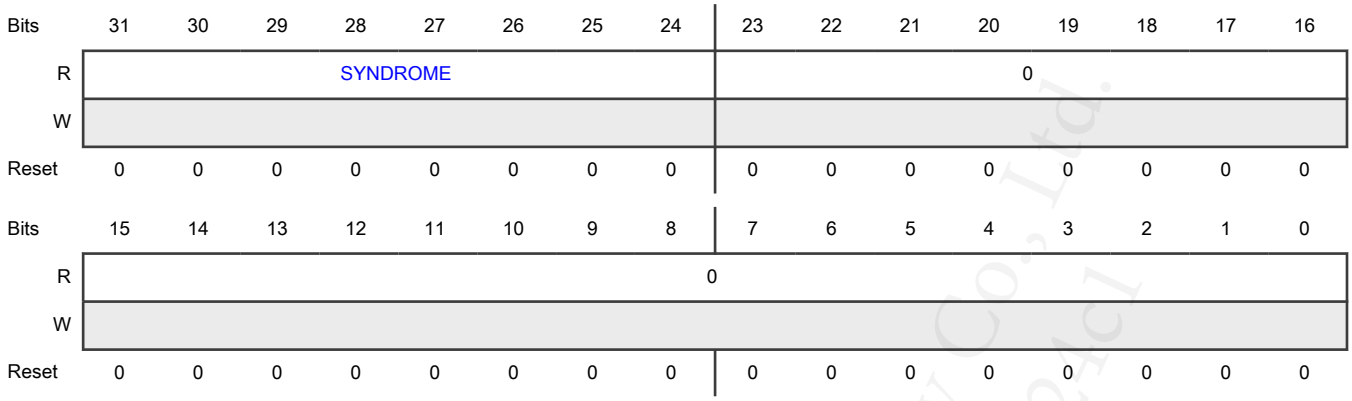
Register	Offset
SYNa	104h + (a × 10h)

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.23.10 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT16)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h
CORR_ERR_CNT9	198h
CORR_ERR_CNT10	1A8h

Table continues on the next page...

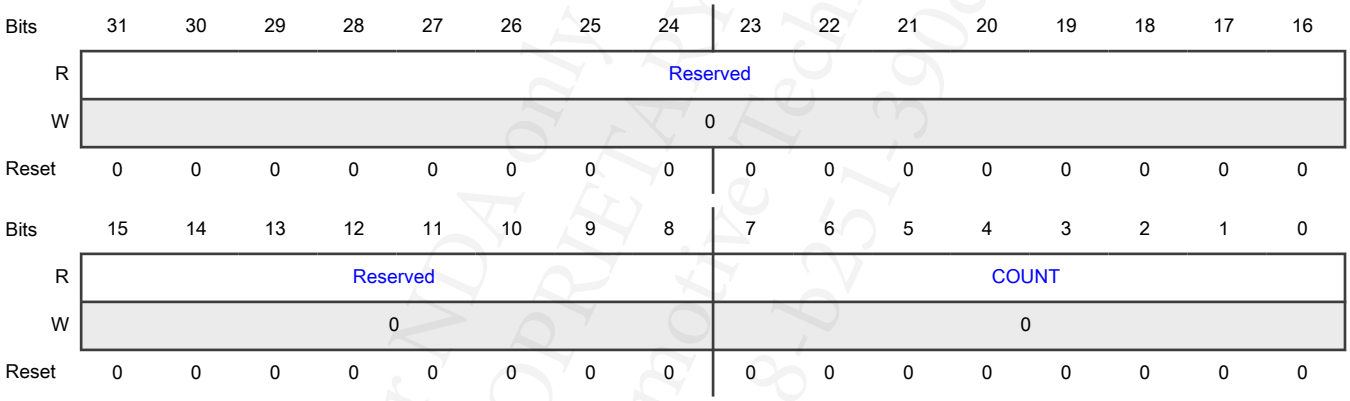
Table continued from the previous page...

Register	Offset
CORR_ERR_CNT11	1B8h
CORR_ERR_CNT12	1C8h
CORR_ERR_CNT13	1D8h
CORR_ERR_CNT14	1E8h
CORR_ERR_CNT15	1F8h
CORR_ERR_CNT16	208h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.24 ERM_STDBY_SRAM register descriptions

You can access the programming model:

- Only in supervisor mode

- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.24.1 ERM_STDBY_SRAM memory map

ERM_STDBY_SRAM base address: 4404_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)	32	ROWZ	0000_0000h

65.24.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

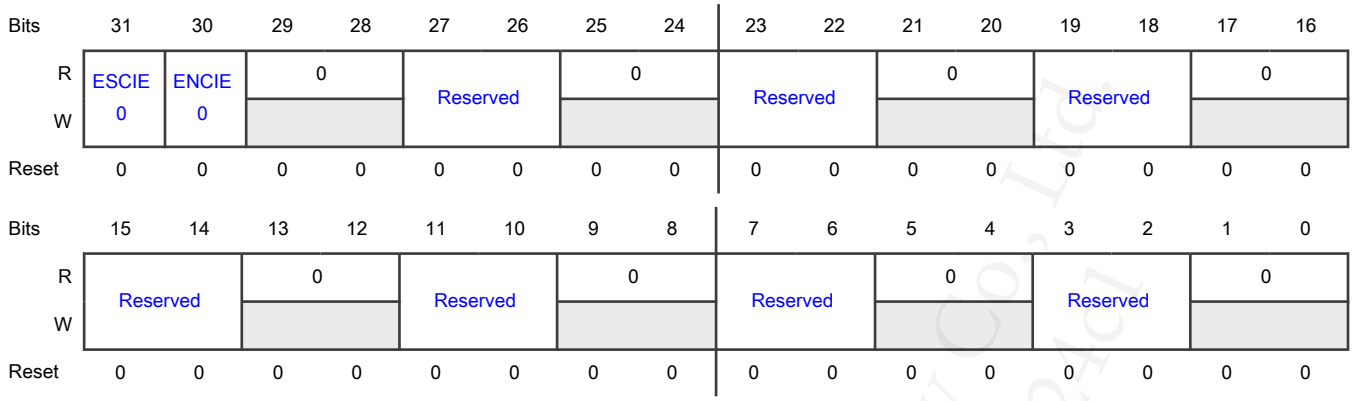
Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-3900a1f624c1

Diagram



Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.24.3 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.24.4 ERM Memory 0 Error Address Register (EAR0)

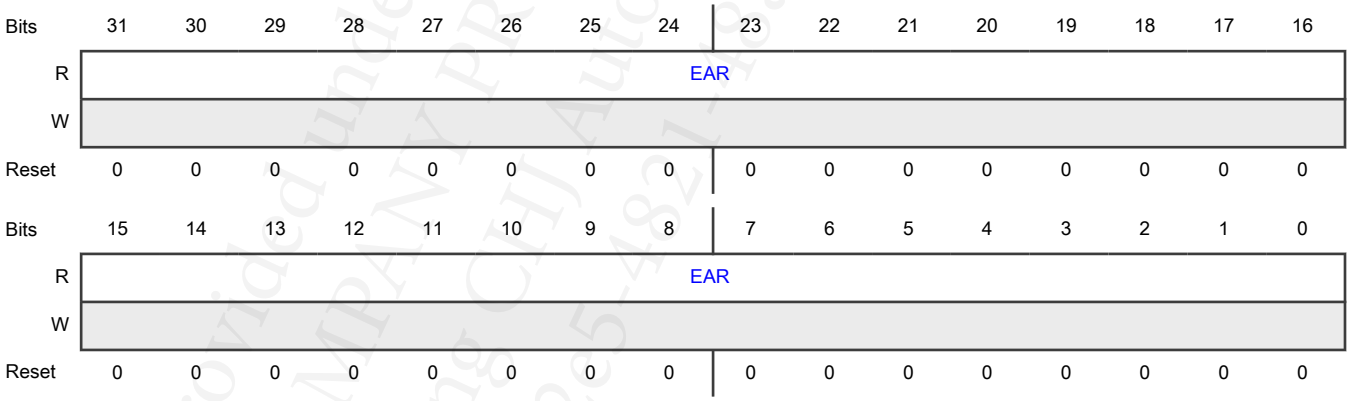
Offset

Register	Offset
EAR0	100h

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0	EAR

Table continues on the next page...

Field	Function
EAR	Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.24.5 ERM Memory 0 Syndrome Register (SYN0)

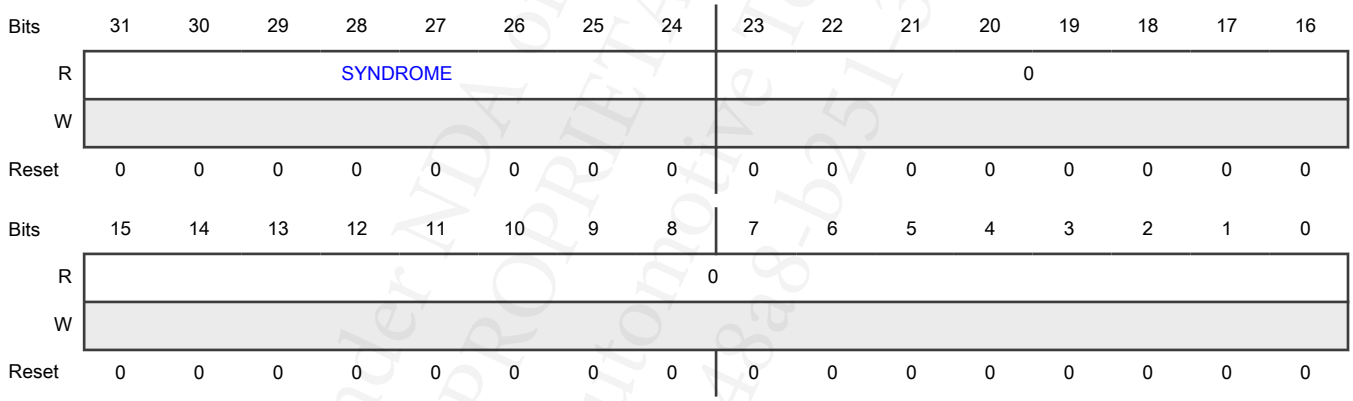
Offset

Register	Offset
SYN0	104h

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-b251-3903ab624c1

65.24.6 ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)

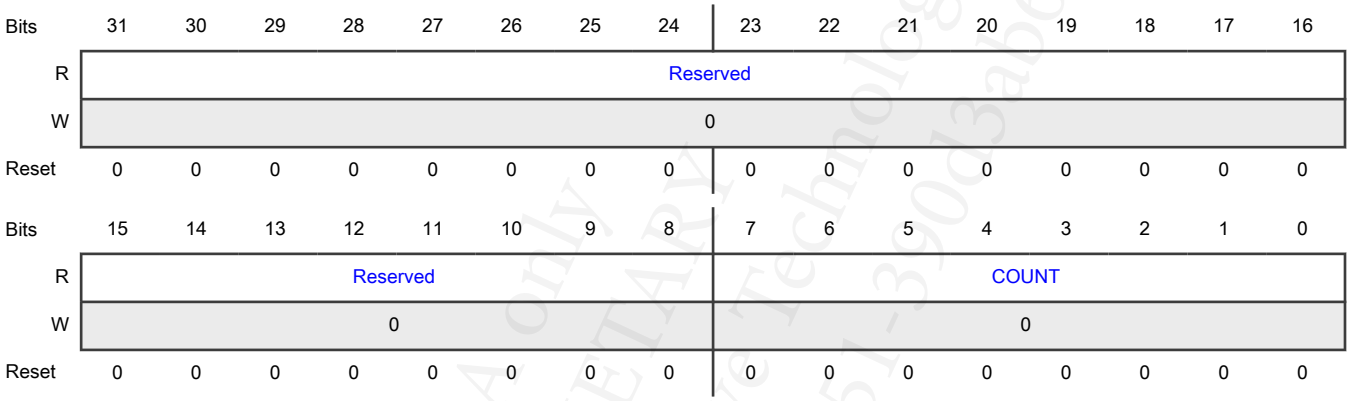
Offset

Register	Offset
CORR_ERR_CNT0	108h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.25 ERM_eDMA0 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode

- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.25.1 ERM_eDMA0 memory map

ERM_eDMA0 base address: 4031_4400h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)	32	ROWZ	0000_0000h

65.25.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c26e5-4821-8a8-b251-39dda6b2401

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.25.3 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c1625-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390a3ab624c1

65.25.4 ERM Memory 0 Error Address Register (EAR0)

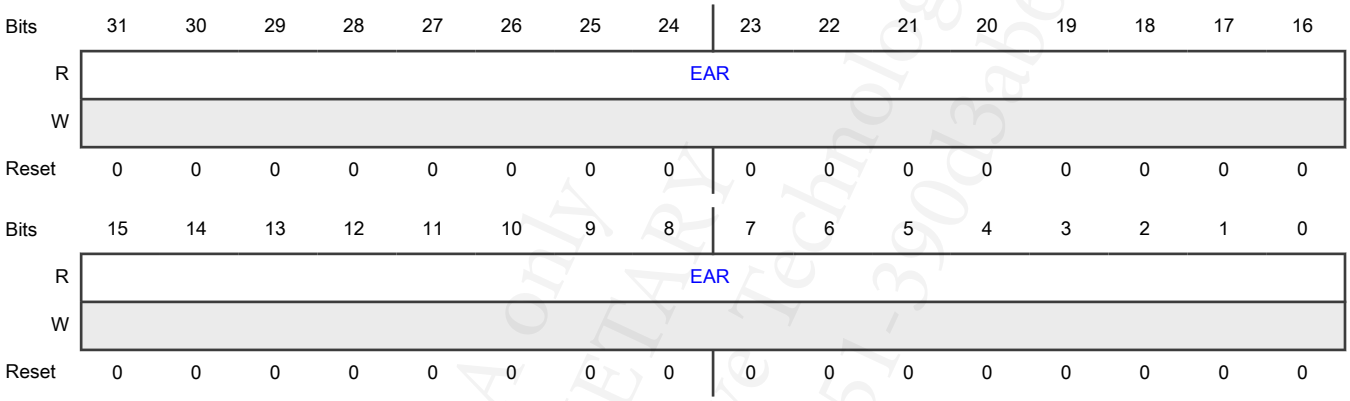
Offset

Register	Offset
EAR0	100h

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.25.5 ERM Memory 0 Syndrome Register (SYN0)

Offset

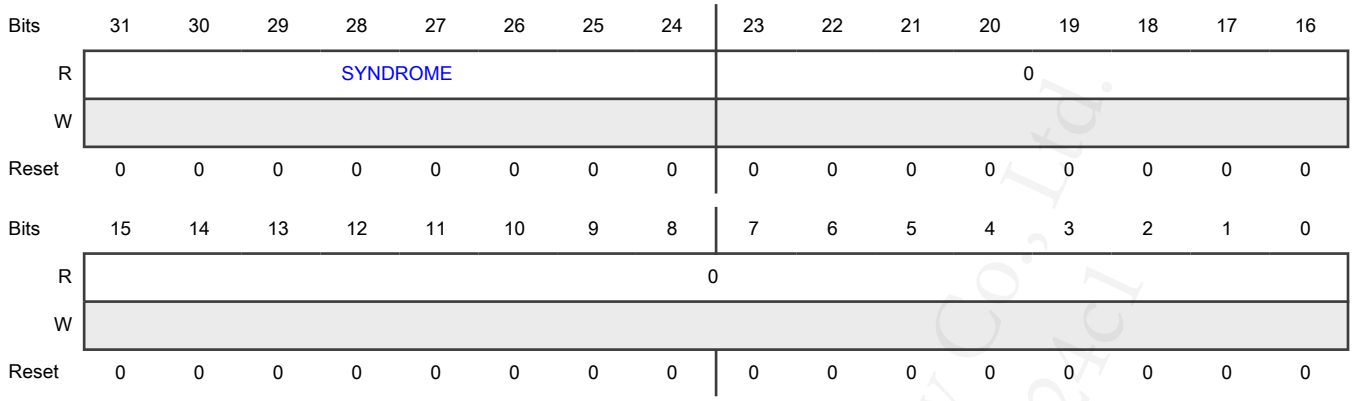
Register	Offset
SYN0	104h

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
Q462e5-482-A8a8-b251-39003ab62401

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.25.6 ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)

Offset

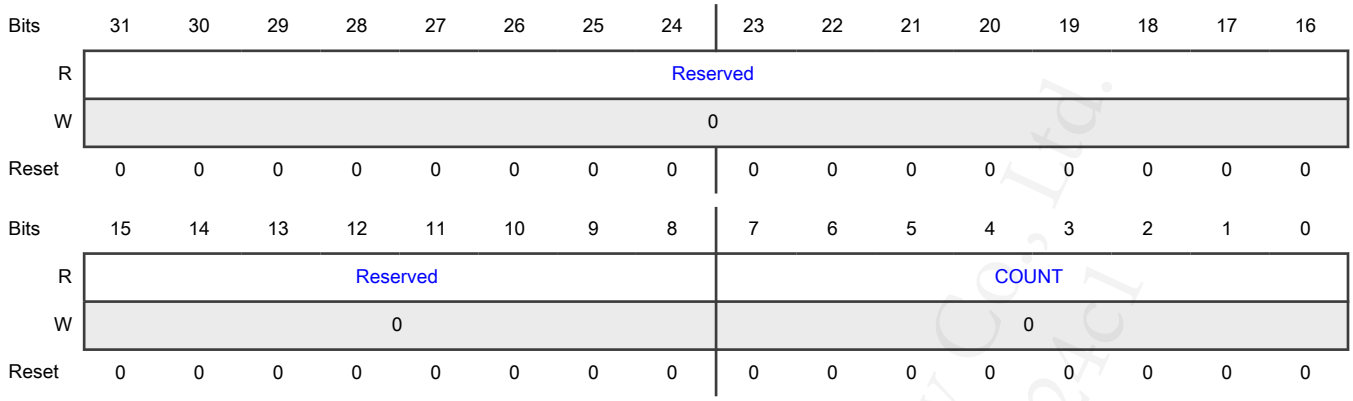
Register	Offset
CORR_ERR_CNT0	108h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.26 ERM_eDMA1 register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.26.1 ERM_eDMA1 memory map

ERM_eDMA1 base address: 4031_4800h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)	32	ROWZ	0000_0000h

65.26.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE0	ENCIE0	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30 ENCIE0	ENCIE0 Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

65.26.3 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31	SBC0

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46255-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
SBC0	Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3b624c1

65.26.4 ERM Memory 0 Error Address Register (EAR0)

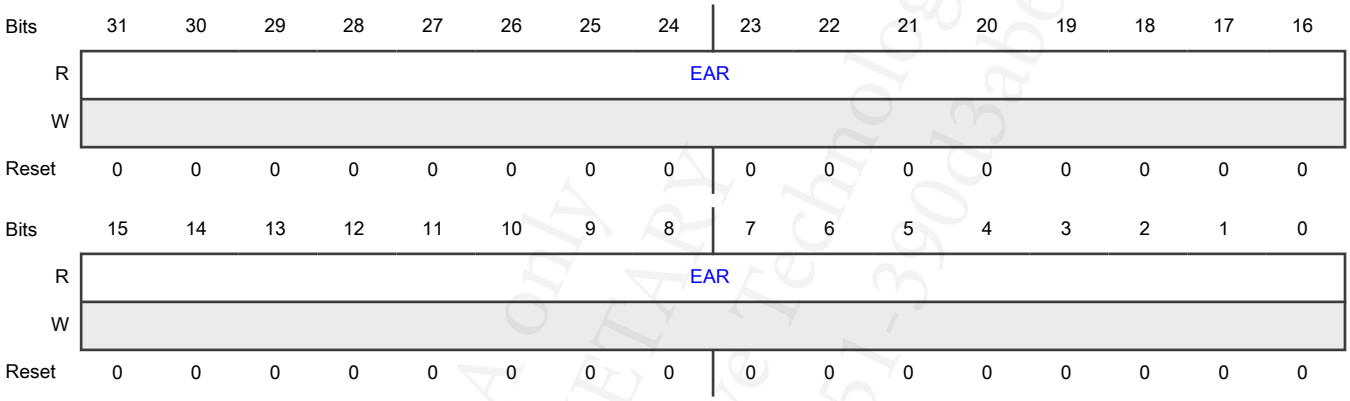
Offset

Register	Offset
EAR0	100h

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.26.5 ERM Memory 0 Syndrome Register (SYN0)

Offset

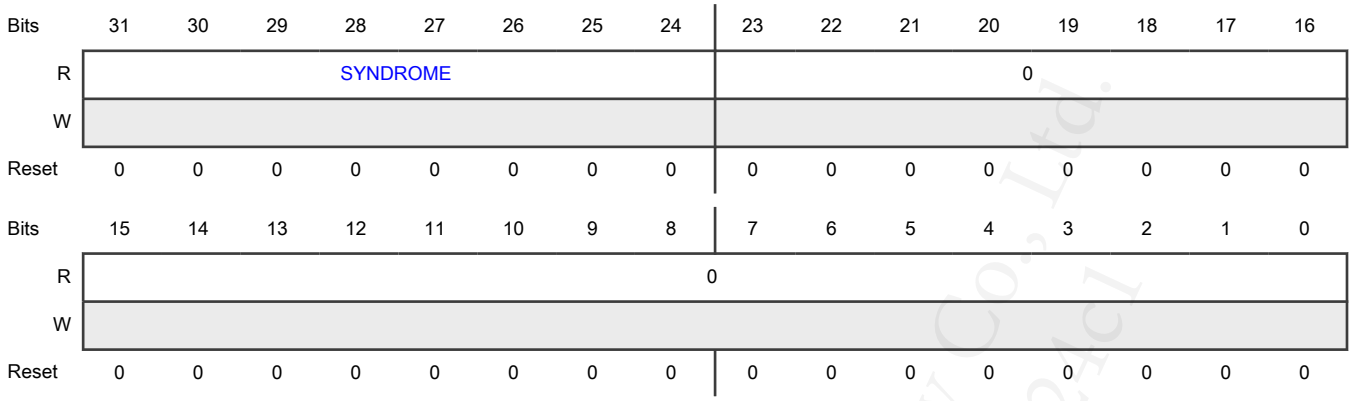
Register	Offset
SYN0	104h

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
e462e5-482f-8a8-b251-3903ab62401

Diagram



Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.26.6 ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT0)

Offset

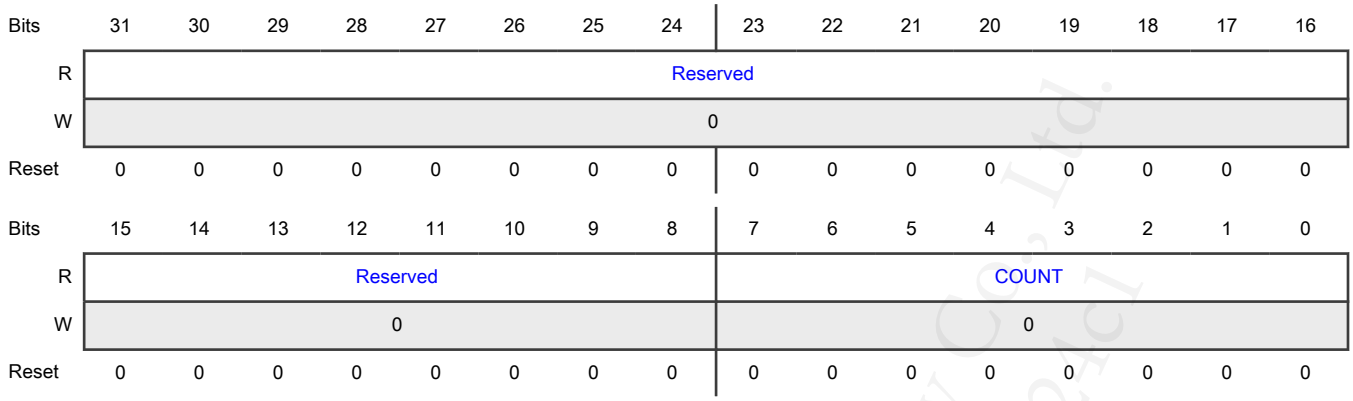
Register	Offset
CORR_ERR_CNT0	108h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.27 ERM register descriptions

You can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes

Attempted accesses to undefined (reserved) memory regions can result in undefined behavior.

NOTE

See the chip-specific ERM information for details on Memory channel mapping.

65.27.1 ERM memory map

ERM_LLCE base address: 4403_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	ERM Configuration Register 0 (CR0)	32	RW	0000_0000h
4h	ERM Configuration Register 1 (CR1)	32	RW	0000_0000h
10h	ERM Status Register 0 (SR0)	32	W1C	0000_0000h
14h	ERM Status Register 1 (SR1)	32	W1C	0000_0000h
100h	ERM Memory 0 Error Address Register (EAR0)	32	RO	0000_0000h
104h	ERM Memory 0 Syndrome Register (SYN0)	32	RO	0000_0000h
108h	ERM Memory 0 Correctable Error Count Register (CORR_ERR_CNT 0)	32	ROWZ	0000_0000h
110h	ERM Memory 1 Error Address Register (EAR1)	32	RO	0000_0000h
114h	ERM Memory 1 Syndrome Register (SYN1)	32	RO	0000_0000h
118h	ERM Memory 1 Correctable Error Count Register (CORR_ERR_CNT 1)	32	ROWZ	0000_0000h
128h	ERM Memory 2 Correctable Error Count Register (CORR_ERR_CNT 2)	32	ROWZ	0000_0000h
130h	ERM Memory 3 Error Address Register (EAR3)	32	RO	0000_0000h
134h	ERM Memory 3 Syndrome Register (SYN3)	32	RO	0000_0000h
138h	ERM Memory 3 Correctable Error Count Register (CORR_ERR_CNT 3)	32	ROWZ	0000_0000h
140h	ERM Memory 4 Error Address Register (EAR4)	32	RO	0000_0000h
144h	ERM Memory 4 Syndrome Register (SYN4)	32	RO	0000_0000h
148h	ERM Memory 4 Correctable Error Count Register (CORR_ERR_CNT 4)	32	ROWZ	0000_0000h
150h	ERM Memory 5 Error Address Register (EAR5)	32	RO	0000_0000h
154h	ERM Memory 5 Syndrome Register (SYN5)	32	RO	0000_0000h
158h	ERM Memory 5 Correctable Error Count Register (CORR_ERR_CNT 5)	32	ROWZ	0000_0000h
160h	ERM Memory 6 Error Address Register (EAR6)	32	RO	0000_0000h
164h	ERM Memory 6 Syndrome Register (SYN6)	32	RO	0000_0000h
168h	ERM Memory 6 Correctable Error Count Register (CORR_ERR_CNT 6)	32	ROWZ	0000_0000h
178h	ERM Memory 7 Correctable Error Count Register (CORR_ERR_CNT 7)	32	ROWZ	0000_0000h
188h	ERM Memory 8 Correctable Error Count Register (CORR_ERR_CNT 8)	32	ROWZ	0000_0000h

Preliminary Information
COMPANY CONFIDENTIAL
Beijing Autonomotive Technology Co.,Ltd.
c84025821-4888-b251-3904ab6d1

65.27.2 ERM Configuration Register 0 (CR0)

Offset

Register	Offset
CR0	0h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE 0	ENCIE 0	0		ESCIE 1	ENCIE 1	0		ESCIE 2	ENCIE 2	0		ESCIE 3	ENCIE 3	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCIE 4	ENCIE 4	0		ESCIE 5	ENCIE 5	0		ESCIE 6	ENCIE 6	0		ESCIE 7	ENCIE 7	0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE0	ESCIE0 Enable Memory 0 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 single-bit correction events is disabled. 1b - Interrupt notification of Memory 0 single-bit correction events is enabled.
30	ENCIE0

Table continues on the next page...

Table continued from the previous page...

Field	Function
ENCIE0	Enable Memory 0 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 0 non-correctable error events is disabled. 1b - Interrupt notification of Memory 0 non-correctable error events is enabled.
29-28 —	Reserved
27 ESCIE1	ESCIE1 Enable Memory 1 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 single-bit correction events is disabled. 1b - Interrupt notification of Memory 1 single-bit correction events is enabled.
26 ENCIE1	ENCIE1 Enable Memory 1 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 1 non-correctable error events is disabled. 1b - Interrupt notification of Memory 1 non-correctable error events is enabled.
25-24 —	Reserved
23 ESCIE2	ESCIE2 Enable Memory 2 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 single-bit correction events is disabled. 1b - Interrupt notification of Memory 2 single-bit correction events is enabled.
22 ENCIE2	ENCIE2 Enable Memory 2 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 2 non-correctable error events is disabled. 1b - Interrupt notification of Memory 2 non-correctable error events is enabled.
21-20 —	Reserved
19	ESCIE3

Table continues on the next page...

Table continued from the previous page...

Field	Function
ESCIE3	Enable Memory 3 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 single-bit correction events is disabled. 1b - Interrupt notification of Memory 3 single-bit correction events is enabled.
18 ENCIE3	ENCIE3 Enable Memory 3 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 3 non-correctable error events is disabled. 1b - Interrupt notification of Memory 3 non-correctable error events is enabled.
17-16 —	Reserved
15 ESCIE4	ESCIE4 Enable Memory 4 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 single-bit correction events is disabled. 1b - Interrupt notification of Memory 4 single-bit correction events is enabled.
14 ENCIE4	ENCIE4 Enable Memory 4 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 4 non-correctable error events is disabled. 1b - Interrupt notification of Memory 4 non-correctable error events is enabled.
13-12 —	Reserved
11 ESCIE5	ESCIE5 Enable Memory 5 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 single-bit correction events is disabled. 1b - Interrupt notification of Memory 5 single-bit correction events is enabled.
10 ENCIE5	ENCIE5 Enable Memory 5 Non-Correctable Interrupt Notification

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. 0b - Interrupt notification of Memory 5 non-correctable error events is disabled. 1b - Interrupt notification of Memory 5 non-correctable error events is enabled.
9-8 —	Reserved
7 ESCIE6	ESCIE6 Enable Memory 6 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 single-bit correction events is disabled. 1b - Interrupt notification of Memory 6 single-bit correction events is enabled.
6 ENCIE6	ENCIE6 Enable Memory 6 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 6 non-correctable error events is disabled. 1b - Interrupt notification of Memory 6 non-correctable error events is enabled.
5-4 —	Reserved
3 ESCIE7	ESCIE7 Enable Memory 7 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 single-bit correction events is disabled. 1b - Interrupt notification of Memory 7 single-bit correction events is enabled.
2 ENCIE7	ENCIE7 Enable Memory 7 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 7 non-correctable error events is disabled. 1b - Interrupt notification of Memory 7 non-correctable error events is enabled.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e2-4821-4888-b251-390d3b624c1

65.27.3 ERM Configuration Register 1 (CR1)

Offset

Register	Offset
CR1	4h

Function

This 32-bit control register configures the interrupt notification capability for:

- Channel 8

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ESCIE8	ENCIE8	0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		0		Reserved		0		Reserved		0		Reserved		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 ESCIE8	ESCIE8 Enable Memory 8 Single Correction Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 single-bit correction events is disabled. 1b - Interrupt notification of Memory 8 single-bit correction events is enabled.
30 ENCIE8	ENCIE8 Enable Memory 8 Non-Correctable Interrupt Notification This field is initialized by hardware reset. 0b - Interrupt notification of Memory 8 non-correctable error events is disabled. 1b - Interrupt notification of Memory 8 non-correctable error events is enabled.
29-28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27-26 —	Reserved
25-24 —	Reserved
23-22 —	Reserved
21-20 —	Reserved
19-18 —	Reserved
17-16 —	Reserved
15-14 —	Reserved
13-12 —	Reserved
11-10 —	Reserved
9-8 —	Reserved
7-6 —	Reserved
5-4 —	Reserved
3-2 —	Reserved
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

65.27.4 ERM Status Register 0 (SR0)

Offset

Register	Offset
SR0	10h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 0
- Channel 1
- Channel 2
- Channel 3
- Channel 4
- Channel 5
- Channel 6
- Channel 7

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC0	NCE0	0		SBC1	NCE1	0		SBC2	NCE2	0		SBC3	NCE3	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBC4	NCE4	0		SBC5	NCE5	0		SBC6	NCE6	0		SBC7	NCE7	0	
W	W1C	W1C			W1C	W1C			W1C	W1C			W1C	W1C		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC0	SBC0 Memory 0 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE0] is enabled. 0b - No single-bit correction event on Memory 0 detected. 1b - Single-bit correction event on Memory 0 detected.

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 NCE0	NCE0 Memory 0 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE0] is enabled. 0b - No non-correctable error event on Memory 0 detected. 1b - Non-correctable error event on Memory 0 detected.
29-28 —	Reserved
27 SBC1	SBC1 Memory 1 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE1] is enabled. 0b - No single-bit correction event on Memory 1 detected. 1b - Single-bit correction event on Memory 1 detected.
26 NCE1	NCE1 Memory 1 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE1] is enabled. 0b - No non-correctable error event on Memory 1 detected. 1b - Non-correctable error event on Memory 1 detected.
25-24 —	Reserved
23 SBC2	SBC2 Memory 2 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE2] is enabled. 0b - No single-bit correction event on Memory 2 detected. 1b - Single-bit correction event on Memory 2 detected.
22	NCE2

Table continues on the next page...

Table continued from the previous page...

Field	Function
NCE2	Memory 2 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE2] is enabled. 0b - No non-correctable error event on Memory 2 detected. 1b - Non-correctable error event on Memory 2 detected.
21-20 —	Reserved
19 SBC3	SBC3 Memory 3 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE3] is enabled. 0b - No single-bit correction event on Memory 3 detected. 1b - Single-bit correction event on Memory 3 detected.
18 NCE3	NCE3 Memory 3 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE3] is enabled. 0b - No non-correctable error event on Memory 3 detected. 1b - Non-correctable error event on Memory 3 detected.
17-16 —	Reserved
15 SBC4	SBC4 Memory 4 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE4] is enabled. 0b - No single-bit correction event on Memory 4 detected. 1b - Single-bit correction event on Memory 4 detected.
14 NCE4	NCE4 Memory 4 Non-Correctable Error Event

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE4] is enabled. 0b - No non-correctable error event on Memory 4 detected. 1b - Non-correctable error event on Memory 4 detected.
13-12 —	Reserved
11 SBC5	SBC5 Memory 5 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE5] is enabled. 0b - No single-bit correction event on Memory 5 detected. 1b - Single-bit correction event on Memory 5 detected.
10 NCE5	NCE5 Memory 5 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE5] is enabled. 0b - No non-correctable error event on Memory 5 detected. 1b - Non-correctable error event on Memory 5 detected.
9-8 —	Reserved
7 SBC6	SBC6 Memory 6 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE6] is enabled. 0b - No single-bit correction event on Memory 6 detected. 1b - Single-bit correction event on Memory 6 detected.
6 NCE6	NCE6 Memory 6 Non-Correctable Error Event This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE6] is enabled. 0b - No non-correctable error event on Memory 6 detected. 1b - Non-correctable error event on Memory 6 detected.
5-4 —	Reserved
3 SBC7	SBC7 Memory 7 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ESCIE7] is enabled. 0b - No single-bit correction event on Memory 7 detected. 1b - Single-bit correction event on Memory 7 detected.
2 NCE7	NCE7 Memory 7 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR0[ENCIE7] is enabled. 0b - No non-correctable error event on Memory 7 detected. 1b - Non-correctable error event on Memory 7 detected.
1-0 —	Reserved

65.27.5 ERM Status Register 1 (SR1)

Offset

Register	Offset
SR1	14h

Function

This 32-bit status register configures the interrupt notification capability for:

- Channel 8

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBC8	NCE8	0		0				0				0			
W	W1C	W1C														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 SBC8	SBC8 Memory 8 Single-Bit Correction Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ESCIE8] is enabled. 0b - No single-bit correction event on Memory 8 detected. 1b - Single-bit correction event on Memory 8 detected.
30 NCE8	NCE8 Memory 8 Non-Correctable Error Event This field is initialized by hardware reset. Write 1 to clear this field. This write also clears the corresponding interrupt notification, if CR1[ENCIE8] is enabled. 0b - No non-correctable error event on Memory 8 detected. 1b - Non-correctable error event on Memory 8 detected.
29-28 —	Reserved
27-24 —	Reserved
23-20 —	Reserved
19-16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-12 —	Reserved
11-8 —	Reserved
7-4 —	Reserved
3-0 —	Reserved

65.27.6 ERM Memory a Error Address Register (EAR0 - EAR6)

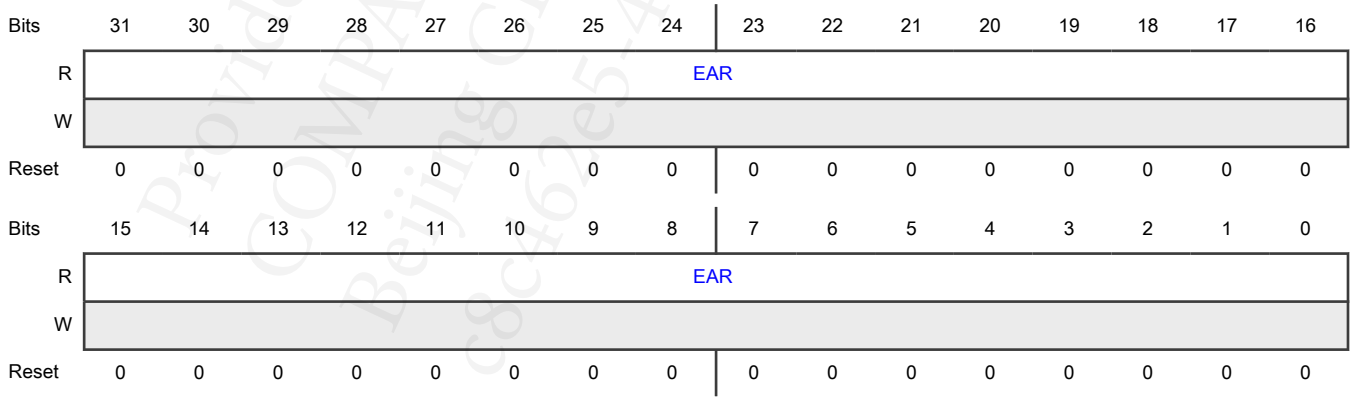
Offset

Register	Offset
EAR0	100h
EAR1	110h
EAR3	130h
EAR4	140h
EAR5	150h
EAR6	160h

Function

Each ERM Memory n Error Address Register is a 32-bit register for capturing the address of the last ECC event in Memory *n*, where *n* denotes the memory channel. Any attempted write to EAR*n* is ignored.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482-48a8-b251-390d3ab624c1

Fields

Field	Function
31-0 EAR	EAR Memory *n* Error Address — This field contains the faulting system address of the last recorded ECC event on Memory *n*.

65.27.7 ERM Memory a Syndrome Register (SYN0 - SYN6)

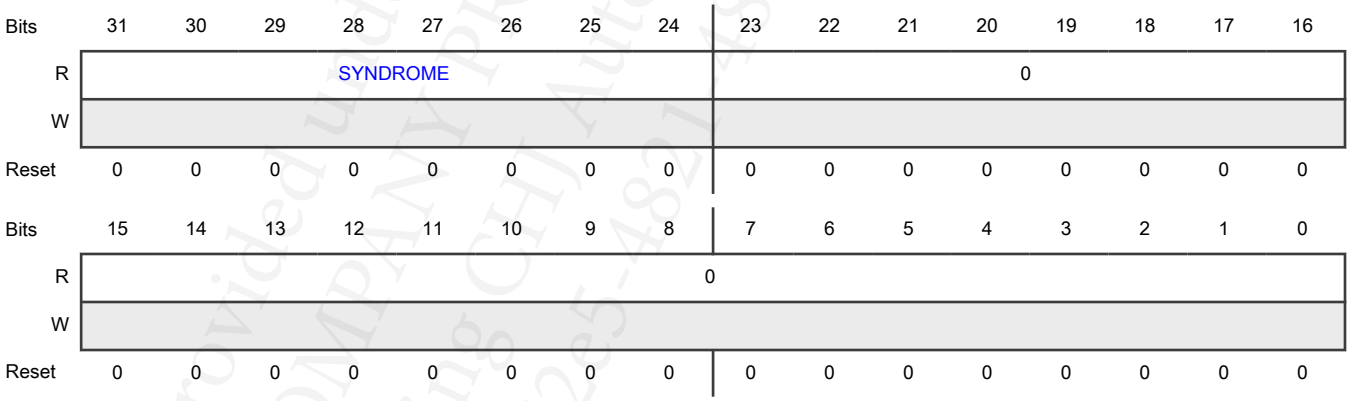
Offset

Register	Offset
SYN0	104h
SYN1	114h
SYN3	134h
SYN4	144h
SYN5	154h
SYN6	164h

Function

The ERM Memory n Syndrome Register is a 32-bit register for capturing the calculated syndrome of the last ECC event on Memory *n*, where *n* denotes the memory channel. Any attempted write to SYN*n* is ignored. The syndrome value identifies the pertinent bit position on a correctable, single-bit data inversion or a non-correctable, single-bit address inversion. The syndrome value does not provide any additional diagnostic information on non-correctable, multi-bit inversions.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-8a8-51390d3ab624c1

Fields

Field	Function
31-24 SYNDROME	SYNDROME Memory *n* Syndrome — This field contains the ECC syndrome associated with the last recorded ECC event on Memory *n*.
23-0 —	Reserved

65.27.8 ERM Memory a Correctable Error Count Register (CORR_ERR_CNT0 - CORR_ERR_CNT8)

Offset

Register	Offset
CORR_ERR_CNT0	108h
CORR_ERR_CNT1	118h
CORR_ERR_CNT2	128h
CORR_ERR_CNT3	138h
CORR_ERR_CNT4	148h
CORR_ERR_CNT5	158h
CORR_ERR_CNT6	168h
CORR_ERR_CNT7	178h
CORR_ERR_CNT8	188h

Function

Each 32-bit ERM Memory n Correctable Error Count Register records the count value of the number of correctable ECC error events for Memory *n*, where *n* denotes the memory channel.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W	0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								COUNT							
W	0								0							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-8 —	Reserved
7-0 COUNT	Memory n Correctable Error Count For each correctable error event, the ERM increments this field's error count value until the counter reaches its maximum value FFh. COUNT field description can be updated as required. COUNT value will stop once it reaches maximum value FFh and will not wrap even if error occurs. Read this field to determine the correctable error count value so far. Write all zeros to this field to reset the counter. Writing non-zero (or partial zero) values has no effect.

65.28 Functional description

65.28.1 Single-bit correction events

When a single-bit correction event on Memory *n* is detected, the ERM:

- Records the event by changing the value of the applicable Status Register bit SR*x*[SBC*n*] to 1.
- Increments the correctable error count value (until the counter reaches its maximum value): CORR_ERR_CNT*n*[COUNT].
- Records the corresponding access address that initiated the event in the Memory *n* Error Address Register: EAR*n* (if this register is present for the channel).
- Stores the corresponding ECC syndrome in the Memory *n* Error Syndrome Register: SYN*n* (if this register is present for the channel). This register identifies the bit position of the corrected data on single-bit data inversion.

The ERM holds event information only for the last reported event.

To clear the record of an event, write 1 to SR*x*[SBC*n*] to change its value to 0.

To reset the correctable error count value, write all zeros to CORR_ERR_CNT*n*[COUNT].

Optional interrupt notification for single-bit correction events

The ERM provides an option to generate an interrupt notification upon the report of a single-bit correction event. This sequence describes how to use the option:

1. To enable interrupt notification for single-bit correction events on Memory *n*, set CR*x*[ESCIE*n*] to 1.
2. Subsequently, when a single-bit correction event on Memory *n* is detected, the ERM:
   - Records the event and address, and stores the ECC syndrome as usual.
   - Additionally sends an interrupt notification corresponding to the event.
3. To clear both the record of an event and the corresponding interrupt notification, write 1 to SR*x*[SBC*n*] to change its value to 0.

65.28.2 Non-correctable error events

When a non-correctable ECC error event on Memory *n* is detected, the ERM:

- Records the event by changing the value of the applicable Status Register bit: SR*x*[NCE*n*] to 1.
- Records the corresponding access address that initiated the event in the Memory *n* Error Address Register: EAR*n* (if this register is present for the channel).

- Stores the corresponding ECC syndrome in the Memory *n* Error Syndrome Register: SYN*n* (if this register is present for the channel).
  - In the event of a non-correctable address bit inversion, SYN*n* identifies the pertinent address bit position.
  - In the event of a non-correctable, multi-bit data inversion, the syndrome value does not provide any additional diagnostic information.

The ERM holds event information only for the last reported event.

To clear the record of an event, write 1 to SR*x*[NCE*n*] to change its value to 0.

Optional interrupt notification for non-correctable error events

The ERM provides an option to generate an interrupt notification upon the report of a non-correctable ECC event. This sequence describes how to use the option:

1. To enable interrupt notifications for non-correctable error events on Memory *n*, set CR*x*[ENCIE*n*] to 1.
2. Subsequently, when a non-correctable error event on Memory *n* is detected, the ERM:
   - Records the event and address and stores the ECC syndrome as usual.
   - Additionally sends an interrupt notification corresponding to the event.
3. To clear both the record of an event and the corresponding interrupt notification, write 1 to SR*x*[NCE*n*] to change its value to 0.

65.29 Initialization

For each ERM channel supporting memory with ECC, prepare the corresponding memory array before enabling ERM interrupts about errors for that memory.

1. Initialize the memory to a known value so that the correct corresponding ECC codeword is stored.
2. During the memory's initialization, if the ERM captures information about any ECC error event, clear the corresponding SR*x*[SBC*n*] or SR*x*[NCE*n*] field that stores the record of the event.
3. Program the applicable CR*x*[ESCIE*n*] and CR*x*[ENCIE*n*] fields to enable ERM interrupts as desired.

65.30 Glossary

ECC	Error correction code

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b2c1-3903ab624c1

Chapter 66
Fault Collection and Control Unit (FCCU)

66.1 Chip-specific FCCU information

66.1.1 Chip-boundary FCCU signals

The EOUT0 and EOUT1 signal are used to notify the internal MCU state to the system.

FCCU signal	Chip signal
EOUT0	FCCU_F0
EOUT1	FCCU_F1

66.1.2 Fault mapping

All the FCCU faults support set or clear injection by FCCU. See the Fault map file attached to this document.

66.1.3 Clock frequency

The FCCU SAFECLK is the RC oscillator. The RC oscillator clock is used to have an independent clock source. The RC oscillator clock frequency is 48 MHz.

66.1.4 FOSU timer interval

The FCCU Output Supervision Unit (FOSU) module timer interval is determined by the value of FOSU_COUNT. On this chip, FOSU_COUNT is 0x69780.

66.1.5 CFG register event bit values by source (C)

For each bit in the FCCU_CFG register, this table specifies the value of *C*, a chip-specific value provided by the bit's associated constant inside the module.

Bit	C
31	0
30	0
29	0
28	0
27	0
26	0
25	0

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482f-4a8-1251-39013ab624c1

Table continued from the previous page...

Bit	C
24	0
23	0
22	0
21	0
20	0
19	0
18	0
17	0
16	0
15	0
14	0
13	0
12	0
11	0
10	0
9	0
8	0
7	0
6	0
5	0
4	0
3	0
2	0
1	0
0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3a6624c1

66.1.6 Supported internal chip reactions

The short functional reset mentioned in this chapter is equivalent to the chip functional reset that is mentioned throughout this document.

66.1.7 Supported FCCU EOUT fault output modes (protocols)

This chip supports only the Bistable fault-output mode of FCCU EOUT. Other EOUT fault-output modes and protocols mentioned in this chapter are not supported on this chip.

66.1.8 Recommended reaction for the fault NCF[21] NoC Mission_Fault_0

The NCF reaction for NoC Mission_Fault_0 should be configured as alarm interrupt only and never as functional reset, and then a software destructive reset inside the interrupt sub-routine must be configured to clean up the chip out of fault state.

66.1.9 NCFs reaction restriction

The fault reaction for the NCFs 47, 48, 49, 59, 60, 61, 62 and 63 should not be configured as functional reset. It should be either alarm interrupt or NMI. In interrupt sub-routine, further reaction to create the necessary destructive reset or PoR or software clearing can be performed to clear up the NCF source.

66.1.10 Clearing FCCU faults

Below are the steps to clear an FCCU fault:

1. NCFx is asserted(through source / fake fault).
2. NCFx status gets set in FCCU.
3. After proper reaction from FCCU, FCCU tries to clear the status. Due to addition of pipes, the NCF takes 2-3 cycles to get deasserted.
4. By writing twice to the NCF status register, the fault is cleared.

66.2 Introduction

The FCCU provides a hardware interface to collect faults and to place the device into a safe state when a failure is detected in the device. No CPU intervention is requested for collection and control operations. FCCU offers a systematic approach to fault collection and control.

66.3 Features

The key features of the FCCU module are these:

- Management of non-critical faults
- HW or SW fault recovery management
- Fault collection from safety relevant modules on the device
- Fault injection (fake faults)
- Collection of test results
- Lockable configuration
  - Changes are only possible after entering the CONFIG state
  - Supports a transient and a permanent lock
  - Configuration changes observed by a watchdog timer
- Configurable fault control

- External reaction (FAULT state): EOUT signaling. Error indication via the pin(s) is controlled by FCCU.
- Internal chip reactions (ALARM state): interrupt request
- Configurable internal chip reactions for each NCF (FAULT state):
  - Short functional reset request pulse
  - NMI
  - No reaction
  - IRQ
- In Bi-Stable operational mode, one of the EOUT signals is high to indicate an OK operational state of FCCU.
- After power on, the EOUT signals have high impedance.[12] They indicate an operational state only after the software configures them.
- In case of a failure event or on software request for error pin indication, the pin(s) are set to faulty state for a minimum time T_{min} (see DELTA_T[DELTA_T]), even if the software tries to release it before (for the case of error pin configured in Bi-Stable mode only).

The self-checking procedure checks the FCCU circuitry at the start up. The FCCU is operational with the default configuration immediately after the completion of the self-checking procedure. Internal (short functional reset request pulse, interrupt request) and external (EOUT signaling) reactions are statically defined or programmable. The default configuration can be modified only in the Configuration (CONFIG) state. FCCU is designed to function when CLKPRIM is faster than the CLKSAFE clocks.

66.4 Block diagram

The following figure represents a top-level diagram of the FCCU module.

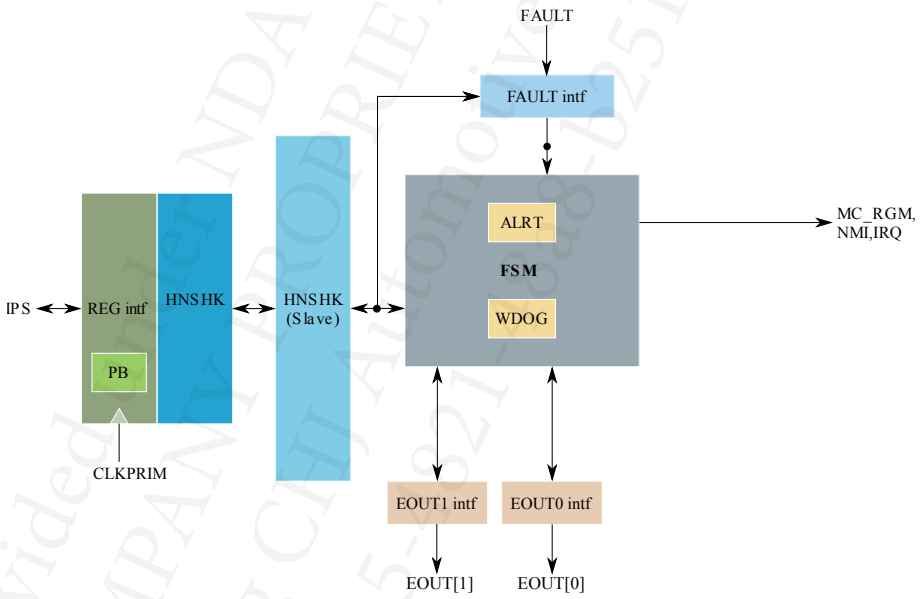


Figure 553. FCCU block diagram

This table describes the FCCU submodules.

[12] Actual value depends on device-specific setting at pad level.

Table 551. FCCU submodules

Submodule	Description
REG intf	Includes the register file, the IPS bus interface, the IRQ interface and the parity block (PB) for the configuration registers
HNSHK blocks (master and slave blocks)	Includes the FSM ability to support the handshake between the REG interface and the FSM unit because of the usage of two asynchronous clocks [CLKPRIM(module clock) and CLKSAFE(RC oscillator clock)]
FSM unit	Implements the main functions of FCCU. The FSM also includes the: • Watchdog timer (WDG) • Alarm timer (ALRT)
FAULT intf	Implements the interface for fault conditioning and management
EOUTx units	Implement the output stage to manage the EOUT interfaces

66.5 Signal description

66.5.1 Signals

This table describes the signals on the boundary of FCCU.

Signal	Direction	Function	Sensitivity	Clock
Clock and reset				
CLKPRIM	Input	Primary Clock—supplies the primary clock.	Rising edge	—
CLKSAFE0	Input	Safe Clock 0—supplies one of two 48 MHz clocks for use by various redundant submodules. This clock and CLKSAFE1 are often referred to as the same clock (CLKSAFE).	Rising edge	—
CLKSAFE1	Input	Safe Clock 1—supplies one of two 48 MHz clocks for use by various redundant submodules. This clock and CLKSAFE0 are often referred to as the same clock (CLKSAFE).	Rising edge	—
Fault interface				
NCF[127:0]	Input	Non-critical Fault Channels—receive indications of non-critical faults from various sources. See the fault map file attached to this document.	Programmable	Asynchronous

Table continues on the next page...

Table continued from the previous page...

Signal	Direction	Function	Sensitivity	Clock
EOUT interface				
EIN[1:0]	Input	Error Inputs—provide the mechanism for FCCU to capture the states that off-chip logic drives on the associated EOUT[1:0] signals. FCCU captures the states in the EINOUT[EIN1] and EINOUT[EIN0] fields.	—	Asynchronous
EOUT[1:0]	Output	Error Outputs—indicate FCCU's condition (faulty, nonfaulty, or configuration) to off-chip logic.	Programmable	CLKSAFE0, CLKSAFE1
FOSU interface				
FIF	Output	FCCU In Fault—indicates to the FOSU module, if present, that FCCU is recognizing a fault. FCCU asserts and then deasserts this signal in two cases: *Case 1* When the EOUT signals are programmed to be always low (CFG[FCCU_SET_CLEAR]), FCCU asserts this signal until the EOUT signals are programmed not to be always low. *Case 2* When the EOUT signals are programmed not to be always low (CFG[FCCU_SET_CLEAR]), FCCU asserts this signal when all of the following are true: • FCCU enters the FAULT state as the result of a fault. • If the fault is a non-critical fault, the associated EOUT_SIG_ENa[EOUTENn] field is set to enabled. FCCU deasserts this signal when FCCU leaves the FAULT state.	High	CLKSAFE0, CLKSAFE1
Interrupt interface				

Table continues on the next page...

Table continued from the previous page...

Signal	Direction	Function	Sensitivity	Clock
NMIOUT	Output	Non-maskable Interrupt Output—Sends a non-maskable interrupt request to the processor core or cores when FCCU enters the FAULT state as the result of a non-critical fault for which NMI is enabled as the reaction (NMI_ENa[NMIENn]).	Low	CLKPRIM

66.6 Functional description

66.6.1 Definitions

In general, the following definitions are applicable for fault management:

- HW recoverable fault: The fault indication is a level-sensitive signal that remains asserted until the fault cause is deasserted. That is, if logical 0 on the fault signal indicates fault, then the status flags are valid as long as the fault line stays at 0. The status is automatically cleared when the fault signal goes to 1. Typically the fault signal is latched external to the FCCU in the module where the fault occurred. The FCCU state transitions are consequently executed on the state changes of the input fault signal. No SW intervention in the FCCU is required to recover the fault condition.
- SW recoverable fault: The fault indication is a signal asserted without a defined time duration. The fault signal is latched in the FCCU. The fault recovery is executed following a SW recovery procedure (status/flag register clearing).

HW recoverable is an option to exclude the handling of error source/s by FCCU management SW, in case it is known that the fault is recoverable by itself when the fault condition is corrected.

These type of chip resets reset the FCCU:

- Power-on
- Destructive

A short functional reset of the chip do not reset the FCCU.

For more information on each type of reset, see the Reset Generation Module (MC_RGM) and reset chapters.

66.6.2 FSM description

The functionality of FCCU is depicted by the FSM state diagram (see Figure 554).

FCCU has four states that are identified with the following meaning:

- CONFIG: Used only to modify the configuration of FCCU from its default. A subset of the FCCU registers, dedicated to define the FCCU configuration (global configuration, reactions to fault, timeout, non-critical fault masking) can be accessed in write mode only in the CONFIG state.

  The CONFIG state is accessible only in the NORMAL state and if the configuration is not locked. A permanent configuration lock can be disabled by a reset that also resets the FCCU. The transient lock register is unlocked by writing BCh into it. FCCU gets transiently locked again if an invalid key is written into TRANS_LOCK[TRANSKEY] (that is, other than BCh). To lock FCCU for configuration, write FFh to PERMNT_LOCK[PERMNTKEY].

  After the release of reset, the state of the transient lock is locked, and the state of the permanent lock is unlocked.

  The locking feature only restricts the FSM movement into CONFIG state. After the user enters the CONFIG state and then tries to lock the configuration, the locking of configuration is effective only after FCCU moves to the NORMAL state; it will not be effective in the current CONFIG state.

The CONFIG to NORMAL state transition can be executed by SW or automatically following a timeout condition of the watchdog. In case the timeout information and the SW request for state change to NORMAL appears at the same time, watchdog timeout has the priority and hence the configuration registers (those that are writable only in the CONFIG state) are reset to their default values. The movement to the NORMAL state is made.

The incoming faults, occurring during the configuration phase (CONFIG state) are latched in order to process them when FCCU is moved to the NORMAL state, according to the new configuration.

All pending faults that occur during the CONFIG state result in both of the following:

— Highest-priority state transition

— Interrupt generation (NMI or alarm IRQ)

If the state transition occurs, it gives the reset reaction corresponding to the worst case based on all the faults (pending or non-pending faults) that occurred during the CONFIG state.

- NORMAL: This is FCCU's operating state when no faults are occurring. It is also the default state on the reset exit. Following state transitions occur on one of the following events:
  — Unmasked non-critical faults with the timeout disabled: FCCU moves to the FAULT state.
  — Unmasked non-critical faults with the timeout enabled: FCCU moves to the ALARM state.
  — Masked non-critical faults: FCCU stays in the NORMAL state.
- ALARM: FCCU moves into the ALARM state when an unmasked non-critical fault occurs and the timeout is enabled. Transition to the ALARM state goes along with an interrupt alarm, if enabled. By definition, this fault may be recovered within a programmable timeout period, before it generates a transition to the FAULT state. The timeout is reinitialized if FCCU enters the NORMAL state. The timeout restarts following the recovery from the FAULT state.
- FAULT: FCCU moves into the FAULT state when one of the following condition occurs:
  — Timeout related to a non-critical fault when FCCU is in the ALARM state
  — Unmasked non-critical faults with the timeout disabled

The transition from the NORMAL or ALARM to the FAULT state goes along with the generation of:

- Internal chip reaction—NMI interrupt (optional)
- External reaction—EOUT signaling (optional)
- Internal chip reaction—SW option: Soft reaction (Short functional reset request pulse if configured)

After moving to the FAULT state, if there is either a previous pending fault or a new fault for which NMI is enabled, NMI generation takes place.

Multiple faults can occur at the same time.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4898-b251-7d3b6b94c1

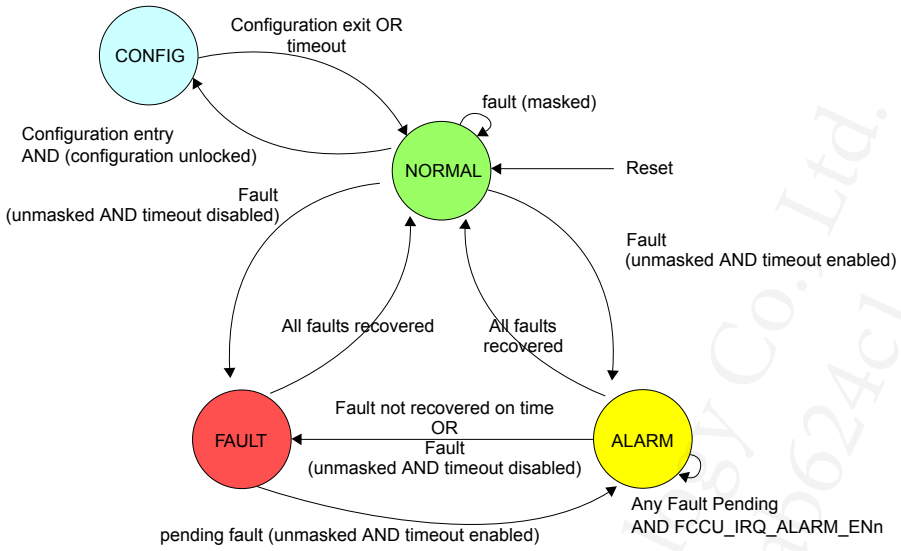


Figure 554. FCCU state diagram

66.6.3 Fault priority scheme and nesting

The FAULT state has a higher priority than the ALARM state in case of concurrent fault events (non-critical) that occur in the NORMAL state.

The ALARM to FAULT state transition occurs if a non-critical fault (unmasked and with timeout disabled) is asserted in the ALARM state.

The ALARM to NORMAL state transition occurs only if all the non-critical faults (including the faults that have been collected after the entry in the ALARM state) have been cleared (SW or HW recovery); otherwise the FCCU remains in the ALARM state.

The FAULT to NORMAL state transition occurs only if all the non-critical faults (including the faults that have been collected after the entry in the FAULT/ALARM state) have been cleared (SW or HW recovery); otherwise the FCCU moves to the ALARM state (if any non-critical fault is still pending and the timeout is not elapsed).

In general, no fault nesting is supported except for the non-critical faults that cause an ALARM to FAULT state transition. In this case, the NCF timer is stopped until the FAULT state is recovered. If FCCU is in the ALARM state and another fault occurs, which has its alarm timeout enabled, then the alarm timer shall not reload and shall not start again.

66.6.4 Fault recovery

The following timing diagrams describe the main use cases of FCCU in terms of fault events and related recovery.

A typical sequence related to non-critical fault management (ALARM state), see Figure 555 and Figure 556, is as follows:

1. Non-critical fault assertion
2. FCCU state transition (automatic): NORMAL to ALARM
   - Alarm interrupt request (if enabled)
   - Timeout running
3. System state: RUN
4. Alarm interrupt management: fault recovery
   - FCCU state transition: ALARM to NORMAL

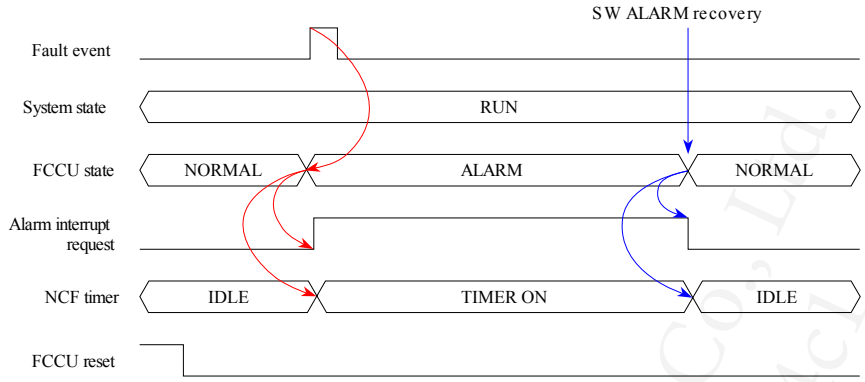


Figure 555. Non-critical fault (ALARM state) SW recovery

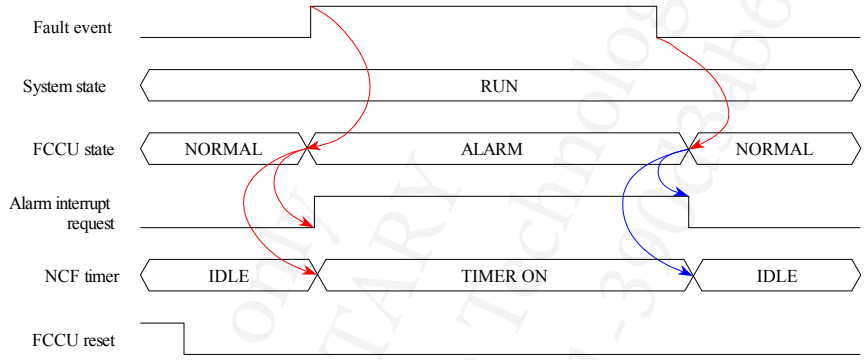


Figure 556. Non-critical fault (ALARM state) HW recovery

A typical sequence related to non-critical fault management (ALARM to FAULT state), see Figure 557, is as follows:

1. Non-critical fault assertion
2. FCCU state transition (automatic): NORMAL to ALARM
   - Alarm interrupt request (if enabled)
   - Timeout running
3. FCCU state transition (following the timeout trigger): ALARM to FAULT
   - NMI assertion (if enabled)
4. NMI interrupt management (if enabled)
   - Fault recovery (by software): FCCU state transition: FAULT to NORMAL

Provided under NXP SLA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4c21-48a8-b151-3903b624c1

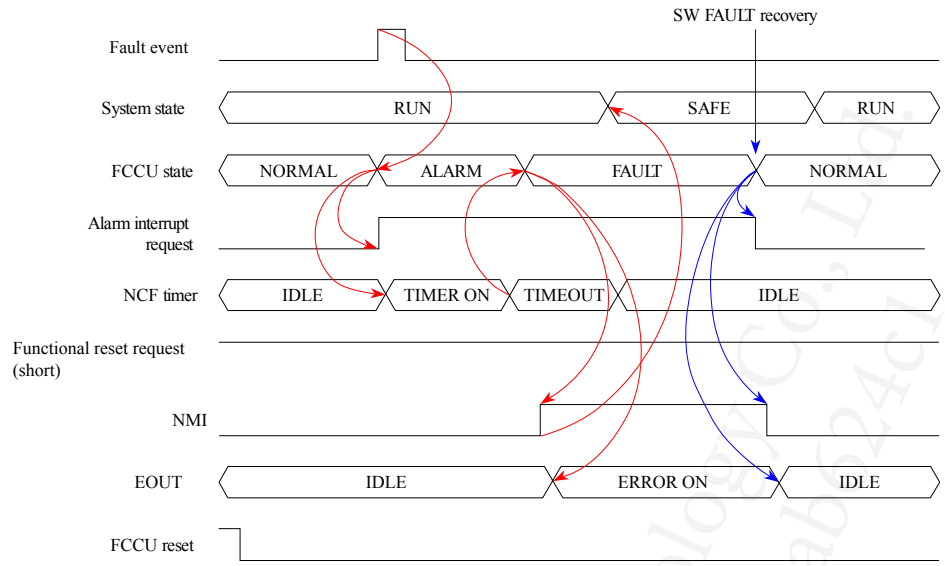


Figure 557. Non-critical fault (ALARM to FAULT state) recovery

66.6.5 EOUT interface

Introduction

You use the EOUT[1:0] signals to indicate FCCU's condition to off-chip logic.

NOTE

For information on the availability and names of these FCCU signals on the boundary of this chip, see the chip-specific FCCU information.

The FCCU conditions

There are three FCCU conditions:

Condition	Description
Faulty	All of the following are true: • The fault-output (EOUT) timer is running (see How the fault-output (EOUT) timer works in Bi-Stable fault-output mode). • FCCU is in FAULT state.
Nonfaulty	All of the following are true: • The fault-output (EOUT) timer is not running. • FCCU is in ALARM or NORMAL state.
Configuration	All of the following are true: • The fault-output (EOUT) timer is not running. • FCCU is in CONFIG state.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482a-48a8-b251-390d3ab624c1

How the fault-output (EOUT) timer works in Bi-Stable fault-output mode

In Bi-Stable fault-output mode (FOM), FCCU starts the fault-output (EOUT) timer when all of the following are true:

- If the EOUT signals are in Bi-Stable FOM, and the EOUT signals are not programmed to be always low (CFG[FCCU_SET_CLEAR]).
- The EOUT timer is not already running.
- FCCU enters the FAULT state as the result of a fault.

When the fault-output (EOUT) timer is already running and a new fault occurs:

- If FCCU is in the CONFIG state: FCCU does not restart the EOUT timer.
- If FCCU is in the NORMAL or ALARM state:
  - And ALARM state is enabled for the fault (non-critical): FCCU enters (or remains in) the ALARM state but does not restart the EOUT timer.
  - And ALARM state is disabled for the fault (non-critical): FCCU enters the FAULT state and restarts the EOUT timer.
- If FCCU is in the FAULT state: FCCU restarts the fault-output (EOUT) timer.

FCCU stops and reinitializes the fault-output (EOUT) timer when all of the following are true:

- If the EOUT signals are in Bi-Stable fault-output mode (CFG[FOM]), and T_{min} (see DELTA_T[DELTA_T]) has expired.
- All faults that caused FCCU to enter or remain in the FAULT state since FCCU started the fault-output (EOUT) timer have been cleared, causing FCCU to return to the NORMAL state.

Prepare the EOUT signals to indicate FCCU's condition

- If the EOUT signals are in Bi-Stable fault-output mode (CFG[FOM]), ensure that the EOUT signals are controlled by FCCU's FSM (CFG[FCCU_SET_CLEAR]).
- Ensure that the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]).

NOTE

If the EOUT signals are in Bi-Stable fault-output mode, you must deactivate and then reactivate the EOUT signals (CFG[FCCU_SET_AFTER_RESET]) to correctly initialize them so they have opposite states.

More about the EOUT interface

Different fault-output modes (protocols) for the fault-output (EOUT) interface are supported (CFG[FOM]):

- Bi-Stable

NOTE

See the chip-specific FCCU information for the fault-output modes supported by this chip.

You can further configure the fault-output modes using the following attributes:

Attribute	Field	Setting used in the example diagrams and tables that follow
Polarity selection	CFG[PS]	For the faulty indication, EOUT1 is high, and EOUT0 is low.

EOUT frequency: This frequency is generated by dividing the CLKSAFE frequency by a fixed factor of 2^{18}.

$$EOUT_{freq} = \frac{CLKSAFE_{freq}}{2^{18}}$$

For example, with a CLKSAFE frequency of 16 MHz, this drives a signal of 61 Hz on EOUT.

In case of a failure event or on software request for EOUT indication, the signal(s) are set to the faulty state for a minimum time (T_{min}), even if software tries to release it before. If software configures the error pins to OK(1), and if a fault comes trying to drive the pin to NOK(0), then priority is given to the fault indication and the error signals indicate NOK, such as an incoming fault is not masked even when software has set the error signal to high. Also, if the error signals are forced to low by software by writing to CFG[FCCU_SET_CLEAR], then the signals shall remain low (or high) for the entire duration of T_{min}. During the T_{min} by a non-software fault, the FCCU FSM moves independently of this signal state (low), and as soon as the timer expires, the pin behavior is dictated by the state in which the FSM finds itself in, and it is not possible to set the signals to OK by software moving FCCU to the CONFIG state, as long as this timer is running. No software intervention is needed to bring the signal from the low state.

Software can bring the pin back to OK state by clearing the faults and waiting for the T_{min} interval to expire, after which the FCCU automatically enters the NORMAL state and the error signal indicates OK.

In case another failure event happens within T_{min} after a first one, the T_{min} counter is restarted.

66.6.5.1 Bi-Stable protocol

The encoding scheme is provided in Table 552 and the related timing diagram is shown in Figure 558.

Table 552. Bi-Stable encoding

Condition	EOUT[1:0] (CFG[PS] is 0)	EOUT[1:0] (CFG[PS] is 1)
Nonfaulty	Static 01	Static 10
Faulty	Static 10	Static 01
Reset	High-impedance (no toggling)[1]	High-impedance (no toggling)
Configuration	Static 01	Static 10

1. Final value depends on device specific settings at pad level.

NOTE

Figure 558 is formatted to display the behavior in all four conditions (reset, nonfaulty, faulty, and configuration), not to imply transitions between one condition and another. In particular, a transition from the faulty condition to the configuration condition is not possible.

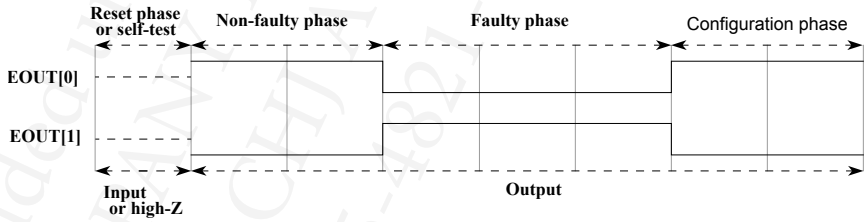


Figure 558. Bi-Stable protocol

66.7 Prepare FCCU for configuration

66.7.1 Introduction

You prepare FCCU for configuration by first configuring the CONFIG state and then putting FCCU in that state.

66.7.2 About preparing FCCU for configuration

When preparing FCCU for configuration, keep the following in mind:

- To put FCCU in CONFIG state, FCCU must be in NORMAL state.
- After FCCU is reset, the configuration is temporarily locked. You must temporarily unlock the configuration before you can put FCCU in the CONFIG state.
- When FCCU is in the CONFIG state, FCCU does not actually save the changes you make to the configuration. To save changes to the configuration, you must manually put FCCU in the NORMAL state. If FCCU automatically leaves the CONFIG state and enters the NORMAL state because the Configuration-state timeout interval (CFG_TO[TO]) expires (called a Configuration-state timeout), FCCU changes the value of the Configuration (CFG) register to its Configuration-state-timeout value and the value of each of the other configuration registers to its reset value. FCCU also changes the value of the Configuration-State Timeout Interval (CFG_TO) register to its reset value. For information on the Configuration-state timeout value, see CFG register bit value at different events. For a list of configuration registers, see Configuration registers.

66.7.3 Configure the CONFIG state

1. Set the Configuration-state timeout interval (CFG_TO[TO]).
2. Enable the Configuration-state-timeout interrupt signal (IRQ_EN[CFG_TO_IEN]), if you want FCCU to request an interrupt when a Configuration-state timeout occurs.

66.7.4 Put FCCU in the CONFIG state

1. Unlock the configuration temporarily (TRANS_LOCK[TRANSKEY]) in Supervisor mode.
2. Check the FCCU status (STAT[STATUS]).
   - If the FCCU status is CONFIG, FCCU is in the CONFIG state. Stop.
   - If the FCCU status is NORMAL, go to step 3.
   - If the FCCU status is ALARM or FAULT, go to step 5.
3. Run the OP1 operation (see Run an operation).
4. Check the operation status (CTRL[OPS]).
   - If the operation status is Successful, FCCU is in the CONFIG state. Stop.
   - If the operation status is Aborted, the configuration is probably permanently locked. Go to step 7.
5. Recover all faults (see Fault recovery).
6. Go to step 2.
7. Reset FCCU.

66.8 Configure FCCU

66.8.1 Introduction

You configure FCCU so it functions according to the needs of your particular application.

66.8.2 About configuring FCCU

When configuring FCCU:

- If you enable a non-critical fault channel but disable all reactions for that channel, FCCU changes state when necessary but does not perform any reaction because reactions are disabled. If you enable reactions for a non-critical fault channel but disable that channel, and FCCU is in the NORMAL state when a fault occurs on the channel, FCCU does not enter the ALARM or FAULT state and therefore does not perform any reaction.

66.8.3 Configure the non-critical fault channels

For each non-critical fault channel that you want FCCU to monitor:

1. Set the recovery type (NCF_CFGa[NCFCn]).
2. Enable at least one type of Fault-state reaction:
   - Chip functional reset (NCFS_CFGa[NCFSCn])
   - Non-maskable interrupt (NMI_ENa[NMIENn])
   - EOUT signaling (EOUT_SIG_ENa[EOUTENn])

   NOTE

   If you enable the chip functional reset as the type of Fault-state reaction for a channel, enable at least one other type of Fault-state reaction for the channel or enable the ALARM state (step 4) for the channel.
3. Set the Alarm-state timeout interval (NCF_TO[TO]), if you plan to enable the ALARM state for any non-critical fault channel.

   NOTE

   Ensure that the Alarm-state timeout interval is less than the FOSU module's timeout interval; otherwise, FOSU generates a chip reset every time a fault occurs on the channel. The FOSU timeout interval (FOSU_COUNT) is a chip-specific value. See the chip-specific FCCU information.
4. Enable the ALARM state (NCF_TOEa[NCFTOEn]) for any non-critical fault channel for which you want FCCU to enter the ALARM state before entering the FAULT state.
5. Enable the Alarm-state reaction (IRQ_ALARM_ENa[IRQENn]) for each non-critical fault channel for which you enabled the ALARM state.
6. Enable the corresponding NCF_Ea[NCFEn] field for each non-critical fault channel that you want FCCU to monitor.

66.9 Put FCCU in the NORMAL state

66.9.1 Introduction

You put FCCU in the NORMAL state to save changes to the configuration, and to allow FCCU to enter the ALARM or FAULT state when a fault occurs on an enabled fault channel.

66.9.2 About putting FCCU in NORMAL state

When putting FCCU in the NORMAL state:

- If you attempt to lock the configuration while FCCU is in CONFIG state, FCCU does not actually lock the configuration until FCCU leaves CONFIG state—that is, either you put FCCU in the NORMAL state, or FCCU puts itself in the NORMAL state because the Configuration-state timeout interval (CFG_TO[TO]) expires.
- After you permanently lock the configuration, you must reset FCCU before you can put FCCU in the CONFIG state.

66.9.3 Put FCCU in the NORMAL state

1. Check the FCCU status (STAT[STATUS]).
   - If the FCCU status is NORMAL, go to step 6.
   - If the FCCU status is CONFIG, go to step 2.
   - If the FCCU status is ALARM or FAULT, go to step 4.
2. Run the OP2 operation (see Run an operation).
3. Check the operation status (CTRL[OPS]).
   - If the operation status is Successful, FCCU is in the NORMAL state. Go to step 6.

- If the operation status is Aborted, go to step 2.

4. Recover all faults (see Fault recovery).
5. Go to step 1.
6. Lock the configuration if you want to prevent any changes to it:
   - To require a key to unlock the configuration, from Supervisor mode, temporarily lock the configuration (TRANS_LOCK[TRANSKEY]).
   - To require a reset of FCCU to unlock the configuration, from Supervisor mode, permanently lock the configuration (PERMNT_LOCK[PERMNTKEY]).

     The configuration is permanently locked until FCCU is reset.

66.10 Manage faults

66.10.1 Introduction

After saving changes to the configuration, you are ready to use FCCU to manage faults.

66.10.2 Determine if there are any unrecovered non-critical faults

Check the unrecovered-fault indicators for the non-critical faults (NCF_Sa[NCFSn]).

66.10.3 Recover a software-recoverable non-critical fault

1. Resolve the source of the software-recoverable non-critical fault.
2. Unlock the NCF_Sa registers (NCFK[NCFK]) using a 32-bit write.
3. Initiate clearing of the unrecovered-fault indicator for the software-recoverable non-critical fault (NCF_Sa[NCFSn]) using a 32-bit write.

   FCCU initiates the OP12 operation.

   NOTE

   If you want to clear multiple unrecovered-fault indicators and those indicators reside in different NCF_Sa registers, you must perform steps 2 and 3 for each individual register.

4. Check the operation status (CTRL[OPS]).
   - If the operation status is In Progress, go to step 4.
   - If the operation status is Successful, go to step 5.
   - If the operation status is Aborted, go to step 2.
5. Check the unrecovered-fault indicator for the software-recoverable non-critical fault (NCF_Sa[NCFSn]).
   - If the indicator indicates no unrecovered fault, the fault has been recovered. Stop.
   - If the indicator still indicates an unrecovered fault, go to step 2.

66.10.4 Clear the freeze-status indicators

1. Run the OP13 operation (see Run an operation).
2. Check the operation status (CTRL[OPS]).
   - If the operation status is Successful, stop .
   - If the operation status is Aborted, go to step 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
ec42e482-1-48a8-b251-390d3ab624c1

66.11 Run operations

66.11.1 Introduction

You run operations to perform actions such as putting FCCU in the CONFIG state or setting the operation status to Idle. For a complete list of operations you can run, see CTRL[OPR].

66.11.2 About running operations

When running operations:

- FCCU ignores any operations initiated while the operation status is In Progress.
- Certain operations must be unlocked before you can initiate them. After you initiate them, they are locked again.

66.11.3 Run an operation

1. Check the operation status (CTRL[OPS]).
2. Go to step 1 if the operation status is In Progress.
3. Unlock the operation (CTRLK[CTRLK]) using a 32-bit write, if the operation must be unlocked before you can initiate it.

NOTE

The Control Key (CTRLK) register used in this step and the Control (CTRL) register used in the next step must be written with consecutive instructions. Do not use read-modify-write instructions, such as bit-field instructions, to modify these registers.

4. Initiate the operation (CTRL[OPR]) using a 32-bit write.
5. Check the operation status (CTRL[OPS]).
6. Go to step 5 if the operation status is In Progress.

The operation status is now Idle (OP15 only), Aborted, or Successful.

66.12 Register descriptions

66.12.1 FCCU register descriptions

The FCCU registers are listed in the table below. Any address offset not explicitly mentioned in this table is reserved.

The FCCU supports word (32-bit), half-word (16-bit), and byte (8-bit) read and write accesses.

Follow these register-access guidelines:

- Do not read from or write to any addresses that are not shown in the following table. Doing so may or may not result in a transfer error.
- Do not write to any of the configuration registers unless FCCU is in the CONFIG state. Doing so results in a transfer error.
- Do not write to the Transient Configuration Lock (TRANS_LOCK) or Permanent Configuration Lock (PERMNT_LOCK) registers unless your code runs in the Supervisor mode. Doing so results in a transfer error.

For each possible NCF failure source, a different reaction—including no reaction—is configurable through the use of NMI, IRQ, and short reset selection registers. It is not possible for a single event upset to switch off all reactions on failures as implementation is per fault source (but it will be possible to switch them all off by SW if intended). Failures themselves are not able to disable all reactions and indications.

The FCCU is not reset by short functional resets.

66.12.1.1 FCCU memory map

FCCU base address: 4030_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Control (CTRL)	32	RW	0000_00C0h
4h	Control Key (CTRLK)	32	WO	0000_0000h
8h	Configuration (CFG)	32	RW	0000_0000h
1Ch - 28h	Non-critical Fault Configuration (NCF_CFG0 - NCF_CFG3)	32	RW	FFFF_FFFFh
4Ch - 68h	Non-critical Fault-State Configuration (NCFS_CFG0 - NCFS_CFG7)	32	RW	0000_0000h
80h - 8Ch	Non-critical Fault Status (NCF_S0 - NCF_S3)	32	W1C	0000_0000h
90h	Non-critical Fault Key (NCFK)	32	WORZ	0000_0000h
94h - A0h	Non-critical Fault Enable (NCF_E0 - NCF_E3)	32	RW	0000_0000h
A4h - B0h	Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0 - NCF_TOE3)	32	RW	FFFF_FFFFh
B4h	Non-critical-Fault Alarm-State Timeout Interval (NCF_TO)	32	RW	0003_A980h
B8h	Configuration-State Timeout Interval (CFG_TO)	32	RW	0000_0005h
BCh	IO Control (EINOUT)	32	RW	See description
C0h	Status (STAT)	32	RO	0000_0010h
C4h	Normal-to-Alarm Freeze Status (N2AF_STATUS)	32	RO	0000_0000h
C8h	Alarm-to-Fault Freeze Status (A2FF_STATUS)	32	RO	0000_0000h
CCh	Normal-to-Fault Freeze Status (N2FF_STATUS)	32	RO	0000_0000h
D0h	Fault-to-Alarm Freeze Status (F2AF_STATUS)	32	RO	0000_0000h
DCh	Non-critical Fault Fake (NCFF)	32	WO	0000_0000h
E0h	IRQ Status (IRQ_STAT)	32	W1C	0000_0000h
E4h	IRQ Enable (IRQ_EN)	32	RW	0000_0000h
F0h	Transient Configuration Lock (TRANS_LOCK)	32	WO	0000_0000h
F4h	Permanent Configuration Lock (PERMNT_LOCK)	32	WO	0000_0000h
F8h	Delta T (DELTA_T)	32	RW	0000_0000h
FCh - 108h	Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0 - IRQ_ALARM_EN3)	32	RW	0000_0000h
10Ch - 118h	Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0 - NMI_EN3)	32	RW	0000_0000h
11Ch - 128h	Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0 - EOUT_SIG_EN3)	32	RW	0000_0000h
12Ch	Alarm-State Timer (TMR_ALARM)	32	RO	0003_A980h
134h	Configuration-State Timer (TMR_CFG)	32	RO	000F_FFFFh
138h	Fault-Output Timer (TMR_ETMR)	32	RO	0000_0000h

66.12.1.2 Control (CTRL)

Offset

Register	Offset
CTRL	0h

Function

Initiates and indicates the status of operations—and enables the Debug mode.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0							0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						DEBUG	0	OPS		0	OPR				
W																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

Fields

Field	Function
31 —	Reserved
30-29 —	Reserved
28-10 —	Reserved
9 DEBUG	Debug Mode Enable Specifies whether the Debug mode is enabled. If so, FCCU enters the Debug mode when the Debug signal is asserted. When FCCU enters the Debug mode, it halts operation and remains in the state it was in before it entered this mode. **NOTE** FOSU does not halt when FCCU enters the Debug mode. Therefore, FOSU can still cause a reset if a fault occurs while FCCU is in the Debug mode. 0b - Disabled

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Enabled
8 —	Reserved
7-6 OPS	Operation Status This field can be read and cleared (via OP15 operation) by the software. 00b - Idle 01b - In progress 10b - Aborted 11b - Successful
5 —	Reserved
4-0 OPR	Operation Run Initiates operations that perform actions such as putting FCCU in the CONFIG state or setting the operation status to Idle. For information on how to run operations, see Run operations. FCCU ignores any write to this field while the operation status (CTRL[OPS]) is "In progress". After completion of an operation, FCCU sets this field to OP0. The following events result in an operation status (CTRL[OPS]) of "Aborted": • Writing to a NCF_Sa register (which automatically initiates the OP12 operation) without first successfully unlocking the register (NCFK[NCFK]) • Initiating an OP1 operation when FCCU is not in the NORMAL state or the configuration is locked • Initiating an OP1, or OP2, operation without first unlocking the operation (CTRLK[CTRLK]) 00000 OP0—No operation 00001 OP1—Applies only when the configuration is unlocked, when FCCU is in the NORMAL state, and immediately after you unlock the operation (CTRLK[CTRLK]). Put FCCU in the CONFIG state. 00010 OP2—Applies only immediately after you unlock the operation (CTRLK[CTRLK]). Put FCCU in the NORMAL state. 00011 Reserved 00100 Reserved 00101 Reserved 00110 Reserved 00111 Reserved 01000 Reserved 01001 Reserved 01010 Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
	01011 Reserved 01100 OP12—Do not initiate this operation; it is automatically initiated by the FCCU. A NCF_Sa register status clear operation is in progress. 01101 OP13—Clear the freeze status registers. 01110 OP14—Do not initiate this operation; it is automatically initiated by the FCCU. A Configuration-state timeout is in progress. For more information, see Configuration registers. 01111 OP15—Set the operation status (CTRL[OPS]) to Idle. 10000 Reserved 10001 Reserved 10010 Reserved 10011 Reserved 10100 Reserved 10101—11110 Forbidden. Writing any of these values returns an operation status (CTRL[OPS]) of "Aborted" with no side effect. 11111 Reserved

66.12.1.3 Control Key (CTRLK)

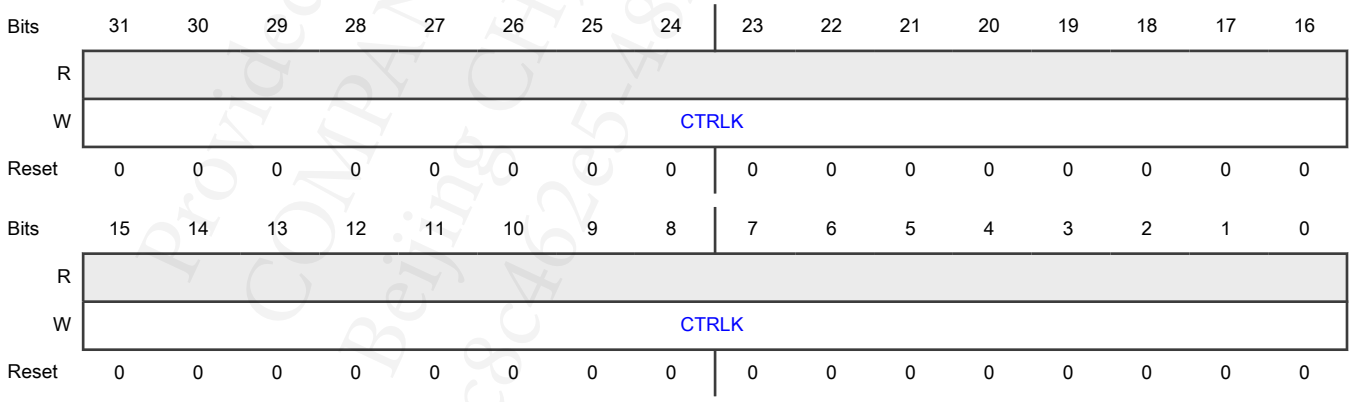
Offset

Register	Offset
CTRLK	4h

Function

See CTRLK[CTRLK].

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CTU Automotive Technology Co., Ltd.
c8c46e25-4821-48a8-b251-390d3ab624c7

Fields

Field	Function
31-0 CTRLK	Locked-Operation Control Key Writable only with a 32-bit write. Unlocks locked operations (CTRL[OPR]) so you can initiate them. For information on how to unlock locked operations before you initiate them, see Run an operation. **NOTE** • You must initiate an operation in the FCCU register access that immediately follows the one that unlocks it; otherwise, the operation is again locked. • Reading from this register always returns the value 0000_0000h. Operations not listed here are not locked and do not need to be unlocked. 9137_56AFh: Unlock OP1. 825A_132Bh: Unlock OP2. Any other value: Do nothing.

66.12.1.4 Configuration (CFG)

Offset

Register	Offset
CFG	8h

Function

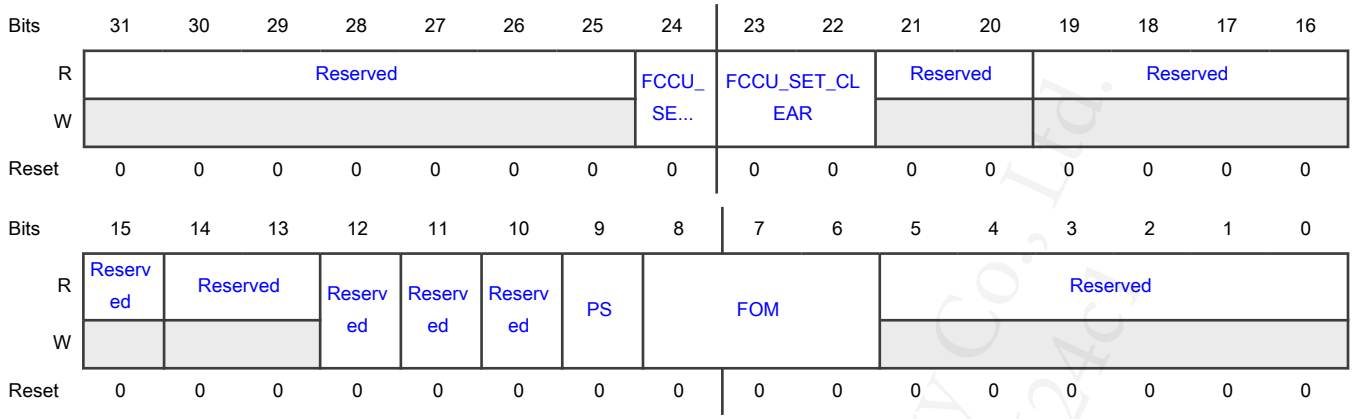
Writable only when FCCU is in the CONFIG state. Changed by FCCU to another value when the chip resets FCCU, a Configuration-state timeout occurs, or you run an OP31 operation. See CFG register bit value at different events for more information. Specifies the global configuration for FCCU.

NOTE

If you specify a new value for any of the fields in this register that affect the EOUT signals while the fault-output (EOUT) timer is running (FCCU is indicating a fault on the EOUT signals), FCCU does not use the new settings you specified until after the fault-output (EOUT) timer expires (FCCU stops indicating a fault on the EOUT signals).

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4824-48a8-b251-390d3ab624c3

Diagram



Fields

Field	Function
31-25 —	Reserved
24 FCCU_SET_AFTER_RESET	Fault-Output (EOUT) Activate For fault-output (EOUT) signaling, controls whether the EOUT signals are active. 0b - Inactive (the EOUT signals are in a high-impedance state) 1b - Active (the EOUT signals indicate FCCU's condition)
23-22 FCCU_SET_CLEAR	Fault-Output (EOUT) Control Applies only to Bi-Stable fault-output mode (CFG[FOM]) and when the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]). Controls whether the fault-output (EOUT) signals are managed by FCCU's FSM. 00b - Controlled by the FSM 01b - Always low 10b - Controlled by the FSM 11b - High until a fault occurs on a channel, regardless of whether that fault is disabled; thereafter, controlled by the FSM. Note: FCCU ignores an attempt to write this value if the fault-output (EOUT) timer is already running.
21-20 —	Reserved
19-16 —	Reserved
15 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
14-13 —	Reserved
12 —	Reserved Always write the reset value to this field.
11 —	Reserved Always write the reset value to this field.
10 —	Reserved
9 PS	Fault-Output (EOUT) Polarity Selection Applies to fault-output (EOUT) signaling and controls the polarity of the signals for fault-output mode indications that hold the signals low or high (versus toggling them or placing them in a high-impedance state). Applies only to Bi-Stable fault-output mode (for all indications). 0b - For the faulty indication, EOUT1 is high, and EOUT0 is low. 1b - For the faulty indication, EOUT1 is low, and EOUT0 is high.
8-6 FOM	Fault-Output (EOUT) Mode For fault-output (EOUT) signaling, controls the protocol of the signaling. 000b - Reserved 001b - Reserved 010b - Bi-Stable 011b - Reserved 100b - Reserved 101b - Test 0 (controlled by the EINOUT register; EOUT1 is an output; EOUT0 is an input) 110b - Test 1 (controlled by the EINOUT register; EOUT1 and EOUT0 are both outputs) 111b - Test 2 (controlled by the EINOUT register; EOUT1 is an input; EOUT0 is an output)
5-0 —	Reserved

66.12.1.5 Non-critical Fault Configuration (NCF_CFG0 - NCF_CFG3)

Offset

Register	Offset
NCF_CFG0	1Ch

Table continues on the next page...

Table continued from the previous page...

Register	Offset
NCF_CFG1	20h
NCF_CFG2	24h
NCF_CFG3	28h

Function

See NCF_CFGa[NCFCn].

Non-critical Fault Configuration (NCF_CFG0 - NCF_CFG3) register	Offset	NCF_CFGa[NCFCn] fields (value of n)	
		Most significant (leftmost) bit	Least significant (rightmost) bit
0	1Ch	31	0
1	20h	63	32
2	24h	95	64
3	28h	127	96

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NCFC 31	NCFC 30	NCFC 29	NCFC 28	NCFC 27	NCFC 26	NCFC 25	NCFC 24	NCFC 23	NCFC 22	NCFC 21	NCFC 20	NCFC 19	NCFC 18	NCFC 17	NCFC 16
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NCFC 15	NCFC 14	NCFC 13	NCFC 12	NCFC 11	NCFC 10	NCFC 9	NCFC 8	NCFC 7	NCFC 6	NCFC 5	NCFC 4	NCFC 3	NCFC 2	NCFC 1	NCFC 0
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 NCFCn	Non-critical Fault Configuration n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls the recovery type (HW or SW) of the associated non-critical

Table continues on the next page...

Field	Function
	fault channel (n). For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. **NOTE** Configure a non-critical fault channel as hardware-recoverable only if the source continues to indicate a fault on the fault channel's input (NCFn) until the condition that caused the fault is no longer true; otherwise, configure the non-critical fault channel as software-recoverable. For the mapping of the NCF_CFGa[NCFCn] fields among registers, see the earlier table in this register description. 0b - Hardware-recoverable 1b - Software-recoverable

66.12.1.6 Non-critical Fault-State Configuration (NCFS_CFG0 - NCFS_CFG7)

Offset

For a = 0 to 7:

Register	Offset
NCFS_CFGa	4Ch + (a × 4h)

Function

See NCFS_CFGa[NCFSCn].

Non-critical Fault-State Configuration (NCFS_CFG0 - NCFS_CFG7) register	Offset	NCFS_CFGa[NCFSCn] fields (value of n)	
		Most significant (leftmost) bit	Least significant (rightmost) bit
0	4Ch	15	0
1	50h	31	16
2	54h	47	32
3	58h	63	48
4	5Ch	79	64
5	60h	95	80
6	64h	111	96
7	68h	127	112

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482c-4a28-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NCFSC15		NCFSC14		NCFSC13		NCFSC12		NCFSC11		NCFSC10		NCFSC9		NCFSC8	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NCFSC7		NCFSC6		NCFSC5		NCFSC4		NCFSC3		NCFSC2		NCFSC1		NCFSC0	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30: NCFSC15 29-28: NCFSC14 27-26: NCFSC13 25-24: NCFSC12 23-22: NCFSC11 21-20: NCFSC10 19-18: NCFSC9 17-16: NCFSC8 15-14: NCFSC7 13-12: NCFSC6 11-10: NCFSC5 9-8: NCFSC4 7-6: NCFSC3 5-4: NCFSC2 3-2: NCFSC1 1-0: NCFSC0	Non-critical Fault-State Configuration n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the chip functional reset is enabled as a Fault-state reaction for the associated non-critical fault channel (n). When the chip functional reset is enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to assert the rst_sfunc_b signal when FCCU enters the FAULT state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. For the mapping of the NCFS_CFGa[NCFSCn] fields among registers, see the earlier table in this register description. 00b - Disabled 01b - Enabled (rst_sfunc_b) (short) 10b - Reserved 11b - Disabled

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-d3ab624c

66.12.1.7 Non-critical Fault Status (NCF_S0 - NCF_S3)

Offset

Register	Offset
NCF_S0	80h
NCF_S1	84h
NCF_S2	88h
NCF_S3	8Ch

Function

See NCF_Sa[NCFSn].

Non-critical Fault Status (NCF_S0 - NCF_S3) register	Offset	NCF_Sa[NCFSn] fields (value of n)	
		Most significant (leftmost) bit	Least significant (rightmost) bit
0	80h	31	0
1	84h	63	32
2	88h	95	64
3	8Ch	127	96

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NCFS 31	NCFS 30	NCFS 29	NCFS 28	NCFS 27	NCFS 26	NCFS 25	NCFS 24	NCFS 23	NCFS 22	NCFS 21	NCFS 20	NCFS 19	NCFS 18	NCFS 17	NCFS 16
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NCFS 15	NCFS 14	NCFS 13	NCFS 12	NCFS 11	NCFS 10	NCFS 9	NCFS 8	NCFS 7	NCFS 6	NCFS 5	NCFS 4	NCFS 3	NCFS 2	NCFS 1	NCFS 0
W	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0	Non-critical Fault Status n

Table continues on the next page...

Field	Function
NCFSn	Indicates whether there is an unrecovered fault on the associated non-critical fault channel (n). **NOTE** To recover a software-recoverable non-critical fault, which includes clearing its unrecovered-fault indicator, see Recover a software-recoverable non-critical fault. FCCU clears the unrecovered-fault indicator for a hardware-recoverable non-critical fault automatically when the source no longer indicates a fault on the fault channel's input signal; if you attempt to clear the unrecovered-fault indicator for a hardware-recoverable non-critical fault, FCCU does not clear the indicator and does not indicate an error. For the mapping of the NCF_Sa[NCFSn] fields among registers, see the earlier table in this register description. 0b - No unrecovered fault 1b - Unrecovered fault

66.12.1.8 Non-critical Fault Key (NCFK)

Offset

Register	Offset
NCFK	90h

Function

See NCFK[NCFK].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W	NCFK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W	NCFK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0	Non-critical Fault Key

Table continues on the next page...

Field	Function
NCFK	Writable only with a 32-bit write. Unlocks the NCF_Sa registers so you can write to them while recovering a software-recoverable non-critical fault. For information on how to unlock the NCF_Sa registers before writing to them, see Recover a software-recoverable non-critical fault. **NOTE** • You must write to one of the NCF_Sa registers immediately after unlocking it (that is, in the FCCU register access that immediately follows the one that unlocks them); otherwise the registers are again locked. If you want to write to multiple NCF_Sa registers, you must unlock each register immediately before you write to it. • Reading from this register always returns the value 0000_0000h. AB34_98FEh: Unlock. Any other value: Do nothing.

66.12.1.9 Non-critical Fault Enable (NCF_E0 - NCF_E3)

Offset

Register	Offset
NCF_E0	94h
NCF_E1	98h
NCF_E2	9Ch
NCF_E3	A0h

Function

See NCF_Ea[NCFEn].

Non-critical Fault Enable (NCF_E0 - NCF_E3) register	Offset	NCF_Ea[NCFEn] fields (value of n)	
		Most significant (leftmost) bit	Least significant (rightmost) bit
0	94h	31	0
1	98h	63	32
2	9Ch	95	64
3	A0h	127	96

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-48c4-48a8-b251-390d3ab624c3

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE
W	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE	NCFE
W	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 NCFEn	Non-critical Fault Enable n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the associated non-critical fault channel (n) is enabled. When a non-critical fault channel is enabled, a fault on that channel causes FCCU to leave the NORMAL state and enter the FAULT state (or ALARM state if enabled for the channel). For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. For the mapping of the NCF_Ea[NCFEn] fields among registers, see the earlier table in this register description. 0b - Disabled 1b - Enabled

66.12.1.10 Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0 - NCF_TOE3)

Offset

Register	Offset
NCF_TOE0	A4h
NCF_TOE1	A8h
NCF_TOE2	ACh
NCF_TOE3	B0h

Function

See NCF_TOEa[NCFTOEn].

Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0 - NCF_TOE3) register	Offset	NCF_TOEa[NCFTOEn] fields (value of n)	
		Most significant (leftmost) bit	Least significant (rightmost) bit
0	A4h	31	0
1	A8h	63	32
2	ACh	95	64
3	B0h	127	96

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NCFTOE31	NCFTOE30	NCFTOE29	NCFTOE28	NCFTOE27	NCFTOE26	NCFTOE25	NCFTOE24	NCFTOE23	NCFTOE22	NCFTOE21	NCFTOE20	NCFTOE19	NCFTOE18	NCFTOE17	NCFTOE16
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NCFTOE15	NCFTOE14	NCFTOE13	NCFTOE12	NCFTOE11	NCFTOE10	NCFTOE9	NCFTOE8	NCFTOE7	NCFTOE6	NCFTOE5	NCFTOE4	NCFTOE3	NCFTOE2	NCFTOE1	NCFTOE0
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 NCFTOEn	Non-critical-Fault Alarm-State Timeout Enable n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the ALARM state is enabled for the associated non-critical fault channel (n). When the ALARM state is enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to leave the NORMAL state and enter the ALARM state instead of FAULT state. If the fault is not recovered within the Alarm-state timeout interval, then FCCU leaves the ALARM state and enters the FAULT state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. For the mapping of the NCF_TOEa[NCFTOEn] fields among registers, see the earlier table in this register description. 0b - Disabled 1b - Enabled

66.12.1.11 Non-critical-Fault Alarm-State Timeout Interval (NCF_TO)

Offset

Register	Offset
NCF_TO	B4h

Function

See NCF_TO[TO]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TO															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TO															
W																
Reset	1	0	1	0	1	0	0	1	1	0	0	0	0	0	0	0

Fields

Field	Function
31-0 TO	Non-critical-Fault Alarm-State Timeout Interval Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls the maximum amount of time that FCCU can be in the ALARM state ($T_{Max\ Alarm}$) according to this equation: $T_{Max\ Alarm} = TO \times T_{CLKSAFE}$ where $T_{CLKSAFE}$ is the safe-clock period. If FCCU enters the ALARM state (because a fault occurs on an enabled non-critical fault channel for which the ALARM state is enabled) and this timeout interval expires (called an Alarm-state timeout), then FCCU leaves the ALARM state and enters the FAULT state. **NOTE** Make sure the Alarm-state timeout interval is less than the FOSU module's timeout interval; otherwise, a fault that occurs while FCCU is in the ALARM state can cause FOSU to generate a chip reset. The FOSU timeout interval (FOSU_COUNT) is chip-specific. See the chip-specific FCCU information.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CUAutomotive Technology Co., Ltd.
c8c4f5-21-3838-b251-390da3ab624c1

66.12.1.12 Configuration-State Timeout Interval (CFG_TO)

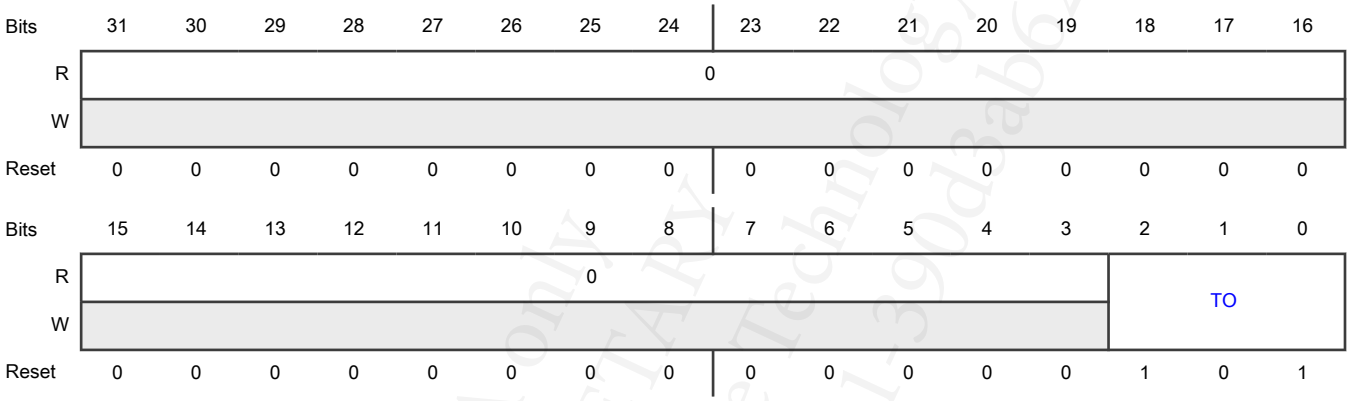
Offset

Register	Offset
CFG_TO	B8h

Function

See CFG_TO[TO]

Diagram



Fields

Field	Function
31-3 —	Reserved
2-0 TO	Configuration-State Timeout Interval Writable only when FCCU is in the NORMAL, ALARM, or FAULT state (not in the CONFIG state). Changed by FCCU to its reset value when a Configuration-state timeout occurs. Not accessible while a Configuration-state timeout (OP14 operation) is in progress. Controls the maximum amount of time that FCCU can be in the CONFIG state ($T_{Max\ configuration}$) according to this equation: $T_{Max\ configuration} = T_{CLKSAFE} \times 2^{(TO + 13)}$ where $T_{CLKSAFE}$ is the safe-clock period. If you put FCCU in the CONFIG state and this timeout interval expires (called a Configuration-state timeout), then FCCU: • Automatically leaves the CONFIG state and enters the NORMAL state • Changes the value of the Configuration (CFG) register to its Configuration-state-timeout value and the value of each of the other configuration registers to its reset value. For information on the Configuration (CFG) register's Configuration-state-timeout value, see CFG register bit value at different events. For a list of configuration registers, see Configuration registers.

Provided under NDA only
COMPANY PROPRIETARY
Copyright 2015 Automotive Technology Co., Ltd.
Information in this document is 6285 4a4-48a8-b251-390d3ab624c1

66.12.1.13 IO Control (EINOUT)

Offset

Register	Offset
EINOUT	BCh

Function

The EINOUT register allows the following operations typically in the NORMAL state:

- To control the EOUT[1] output level when the FCCU is configured in "Test1" or "Test0" fault output mode (CFG[FOM])
- To control the EOUT[0] output level when the FCCU is configured in "Test1" or "Test2" fault output mode (CFG[FOM])
- to observe the state of signals at EIN[1:0] pins

The following table shows Bi-Stable encoding.

Table 553. Bi-Stable encoding

Mode = CFG[FOM]	EOUT[0]	EOUT[1]
Test1	output	output
Test2	output	input
Test0	input	output

NOTE

Because of the resynchronization stage of the EOUT interface, there is a latency of a few CLKSAFE cycles following a write/read operation of the EINOUT register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										EIN1	EIN0	0		EOUT1	EOUT0
W																
Reset	0	0	0	0	0	0	0	0	0	0	u[1]	u	0	0	0	0

1. Reset value varies as per the corresponding EIN signal.

Fields

Field	Function
31-6 —	Reserved
5 EIN1	Error Input 1 Applies only when the EOUT signals are active (FCCU_SET_AFTER_RESET). Indicates the state of the EIN1 signal. 0b - Low 1b - High
4 EIN0	Error Input 0 Applies only when the EOUT signals are active (FCCU_SET_AFTER_RESET). Indicates the state of the EIN0 signal. 0b - Low 1b - High
3-2 —	Reserved
1 EOUT1	EOUT1 Error out 1 (significant only if the CFG.FOM = Test1 or Test0 => EOUT[1] configured in output mode). The EOUT1 set/clear the respective EOUT[1] output signal if CFG.FOM = 110 or 101, otherwise it is a "don't-care" value. **NOTE** When the configuration watchdog timer expires, FCCU changes the value of this field to its reset value. 0b - force EOUT[1] = 0 1b - force EOUT[1] = 1
0 EOUT0	EOUT0 Error out 0 (significative only if the CFG.FOM = Test1 or Test2 => EOUT[0] configured in output mode). The EOUT0 set/clear the respective EOUT[0] output signal if CFG.FOM = 110 or 111, otherwise it is a "don't care" value. **NOTE** When the configuration watchdog timer expires, FCCU changes the value of this field to its reset value. 0b - force EOUT[0] = 0 1b - force EOUT[0] = 1

Provided under NDA only
COMPANY PROPRIETARY
Beijing CTI Automotive Technology Co., Ltd.
c84625_421_48ab_b261_390d3ab624c1

66.12.1.14 Status (STAT)

Offset

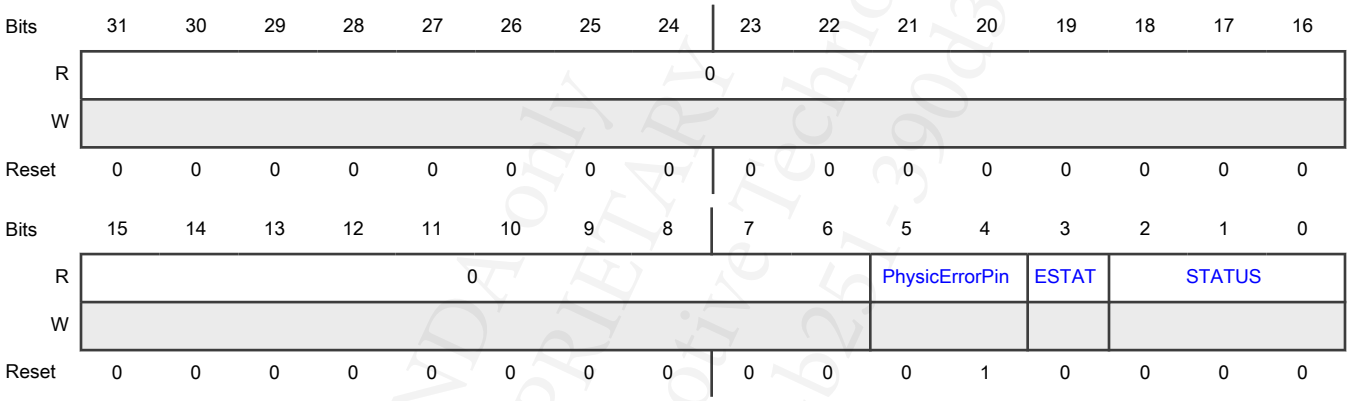
Register	Offset
STAT	C0h

Function

This register indicates the following:

- States that FCCU is driving on the EOUT signals
- Whether FCCU is in a faulty condition
- Current state of FCCU

Diagram



Fields

Field	Function
31-6 —	Reserved
5-4 PhysicErrorPin	EOUT Signal States Applies only when the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]). Indicates the states that FCCU is driving on the EOUT signals. 00b - EOUT1 is low; EOUT0 is low. 01b - EOUT1 is low; EOUT0 is high. 10b - EOUT1 is high; EOUT0 is low. 11b - EOUT1 is high; EOUT0 is high.
3 ESTAT	FCCU Faulty Condition

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Indicates whether FCCU is in faulty condition (as indicated by the EOUT signals). For more information, see The FCCU conditions. 0b - Not in faulty condition (in nonfaulty or configuration condition) 1b - In faulty condition
2-0 STATUS	FCCU State Indicates the current state of FCCU 000b - NORMAL 001b - CONFIG 010b - ALARM 011b - FAULT 100b - Reserved 101b - Reserved 110b - Reserved 111b - Reserved

66.12.1.15 Normal-to-Alarm Freeze Status (N2AF_STATUS)

Offset

Register	Offset
N2AF_STATUS	C4h

Function

See N2AF_STATUS[NAFS].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								NAFS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-8 —	Reserved
7-0 NAFS	Normal-to-Alarm Freeze Status Used only for testing and debugging. Indicates whether FCCU left the NORMAL state and entered the ALARM state since the last time this register was cleared and, if so, which non-critical fault caused FCCU to do so. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00h: No Normal-to-Alarm-state transition (cleared) 01h: NCF0 10h: NCF1 ... 7Fh: NCF126 80h: NCF127 ... FFh: Multiple Normal-to-Alarm-state transitions

66.12.1.16 Alarm-to-Fault Freeze Status (A2FF_STATUS)

Offset

Register	Offset
A2FF_STATUS	C8h

Function

Used only for testing and debugging. Indicates whether FCCU left the ALARM state and entered the FAULT state since the last time this register was cleared and, if so, which type of fault caused FCCU to do so.

NOTE

To clear this register and the other freeze status registers, see Clear the freeze-status indicators.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CTU Automotive Technology Co., Ltd.
c8c462e5-4221-48a8-b251-390d3ab624c4

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						AF_SRC		AFFS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9-8 AF_SRC	Alarm-to-Fault Source Used only for testing and debugging. Indicates the type of fault that caused FCCU to leave the ALARM state and enter the FAULT state since the last time this register was cleared. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00b - No Alarm-to-Fault-state fault 01b - Reserved 10b - Non-critical fault 11b - Multiple Alarm-to-Fault-state faults
7-0 AFFS	Alarm-to-Fault Freeze Status Used only for testing and debugging. Indicates whether FCCU left the ALARM state and entered the FAULT state since the last time this register was cleared and, if so, which fault caused FCCU to do so. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00h: No Alarm-to-Fault-state transition (cleared) 01h: NCF0 (due to an Alarm-state timeout) 10h: NCF1 (due to an Alarm-state timeout) ... 7Fh: NCF126 (due to an Alarm-state timeout)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	80h: NCF127 (due to an Alarm-state timeout) ... FFh: Multiple Alarm-to-Fault-state transitions

66.12.1.17 Normal-to-Fault Freeze Status (N2FF_STATUS)

Offset

Register	Offset
N2FF_STATUS	CCh

Function

Used only for testing and debugging. Indicates whether FCCU left the NORMAL state and entered the FAULT state since the last time this register was cleared and, if so, which type of fault caused FCCU to do so.

NOTE

To clear this register and the other freeze status registers, see Clear the freeze-status indicators.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						NF_SRC		NFFS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9-8 NF_SRC	Normal-to-Fault Source

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Used only for testing and debugging. Indicates the type of fault that caused FCCU to leave the NORMAL state and enter the FAULT state since the last time this register was cleared. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00b - No Normal-to-Fault-state fault 01b - Reserved 10b - Non-critical fault 11b - Multiple Normal-to-Fault-state faults
7-0 NFFS	Normal-to-Fault Freeze Status Used only for testing and debugging. Indicates whether FCCU left the NORMAL state and entered the FAULT state since the last time this register was cleared and, if so, which fault caused FCCU to do so. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00h: No Normal-to-Fault-state transition (cleared) 01h: NCF0 10h: NCF1 ... 7Fh: NCF126 80h: NCF127 ... FFh: Multiple Normal-to-Fault-state transitions

66.12.1.18 Fault-to-Alarm Freeze Status (F2AF_STATUS)

Offset

Register	Offset
F2AF_STATUS	D0h

Function

See F2AF_STATUS[FAFS].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
c8c462e5-4821-48a8-b251-3903ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							FAFS								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved
8-0 FAFS	Fault-to-Alarm Freeze Status Used only for testing and debugging. Indicates whether FCCU left the FAULT state and entered the ALARM state since the last time this register was cleared and, if so, which non-critical fault caused FCCU to do so. **NOTE** To clear this register and the other freeze status registers, see Clear the freeze-status indicators. 00h: No Fault-to-Alarm-state transition (cleared) 01h: NCF0 10h: NCF1 ... 7Fh: NCF126 80h: NCF127 ... FFh: Multiple Fault-to-Alarm-state transitions

66.12.1.19 Non-critical Fault Fake (NCFF)

Offset

Register	Offset
NCFF	DCh

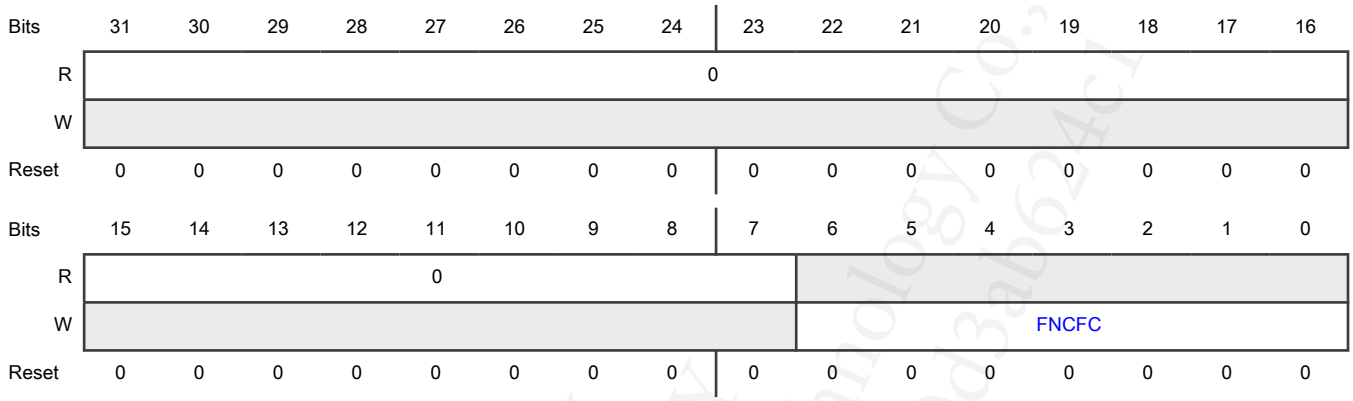
Provided under NDA only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d8ab624c1

Function

This register contains a unique code to set a non-critical fault in mutually exclusive mode by the external FAULT interface (signal setting). It allows the SW emulation of the non-critical faults, by injecting the fault directly in the FAULT root, to verify the entire path and reaction. The reaction following a fake non-critical fault cannot be masked.

This is a write-only register with a set of codes corresponding to each non-critical fault injection.

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 FNCFC	FNCFC Fake non-critical fault code **NOTE** Writing to this field injects fake faults; writing 00 and the default value being 0 renders different results. 00h: Fake non-critical fault injection at non-critical fault source 0 01h: Fake non-critical fault injection at non-critical fault source 1 02h: Fake non-critical fault injection at non-critical fault source 2 ... 7Fh: Fake non-critical fault injection at non-critical fault source 127

66.12.1.20 IRQ Status (IRQ_STAT)

Offset

Register	Offset
IRQ_STAT	E0h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e54821-4c8-b251-3903ab624c1

Function

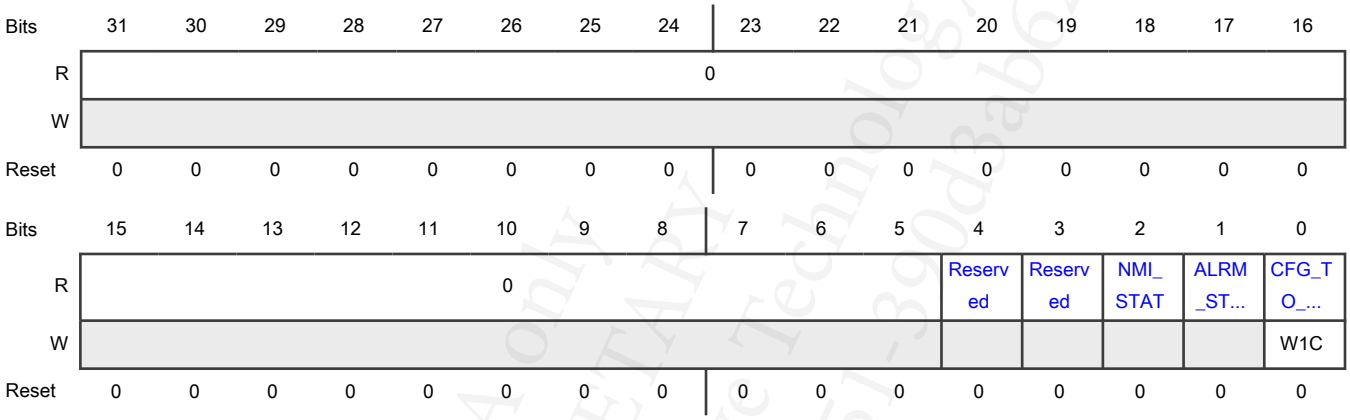
This register provides the FCCU interrupt status related to the following events:

- Configuration-state timeout error
- Alarm interrupt
- NMI interrupt

The configuration-state timeout interrupt is asserted if both IRQ_STAT[CFG_TO_STAT] and IRQ_EN[CFG_TO_IEN] bits are asserted. It is cleared when a 1 is written to the IRQ_STAT[CFG_TO_STAT] bit.

The NMI and ALARM interrupts are asserted and cleared according to the FCCU state. The status bits of the IRQ_STAT trace the status of the related interrupt lines.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 —	Reserved
3 —	Reserved
2 NMI_STAT	NMI Interrupt Status 0b - NMI interrupt is OFF 1b - NMI interrupt is ON
1 ALRM_STAT	Alarm Interrupt Status 0b - Alarm interrupt is OFF 1b - Alarm interrupt is ON
0	Configuration-State Timeout Status

Table continues on the next page...

Table continued from the previous page...

Field	Function
CFG_TO_STAT	0b - No configuration-stat timeout error 1b - Configuration-state timeout error

66.12.1.21 IRQ Enable (IRQ_EN)

Offset

Register	Offset
IRQ_EN	E4h

Function

This register is used to configure enabling of interrupt related to the "Configuration-state timeout error".

The configuration-state timeout interrupt is asserted if both the IRQ_STAT[CFG_TO_STAT] and IRQ_EN[CFG_TO_IEN] fields are set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	0		CFG_TO_...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-4 —	Reserved
3 —	Reserved
2-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 CFG_TO_IEN	Configuration-State Timeout Interrupt Enable 0b - Configuration-state timeout interrupt disabled 1b - Configuration-state timeout interrupt enabled

66.12.1.22 Transient Configuration Lock (TRANS_LOCK)

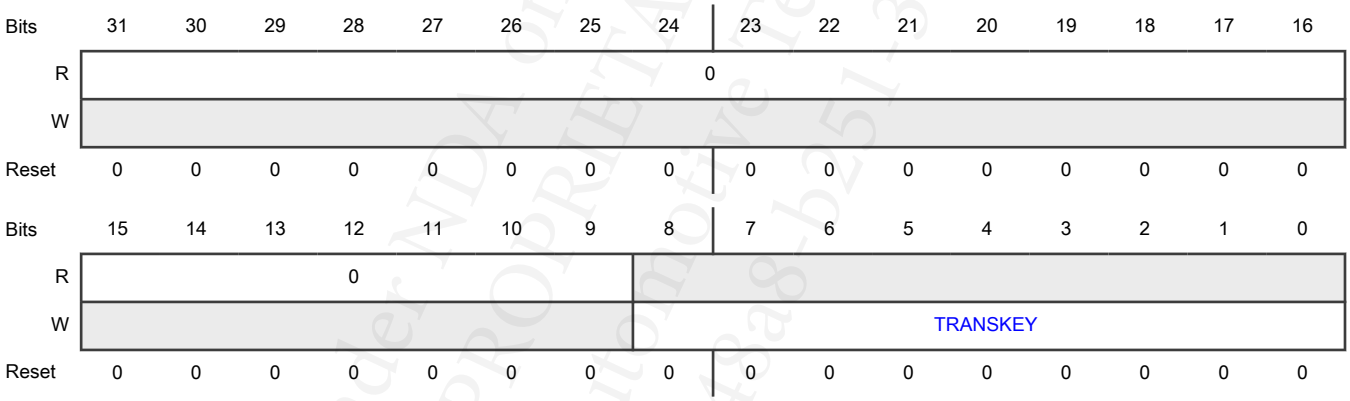
Offset

Register	Offset
TRANS_LOCK	F0h

Function

See TRANS_LOCK[TRANSKEY]

Diagram



Fields

Field	Function
31-9 —	Reserved
8-0 TRANSKEY	Transient Configuration Lock Writable only by code running in Supervisor mode. Temporarily locks and unlocks the configuration. Locking the configuration prevents FCCU from entering the CONFIG state. For information about putting FCCU in configuration, see Prepare FCCU for configuration and Configure FCCU.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE You can write to this field when FCCU is in any state, but the lock will not get into effect until FCCU is in the NORMAL state. BCh: Unlock. Any other value: Lock.

66.12.1.23 Permanent Configuration Lock (PERMNT_LOCK)

Offset

Register	Offset
PERMNT_LOCK	F4h

Function

See PERMNT_LOCK[PERMNTKEY]

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W								PERMNTKEY								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9 —	Reserved
8-0 PERMNTKEY	Permanent Configuration Lock

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-482f-4ba8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
	Writable only by code running in the Supervisor mode. Permanently locks the configuration, which prevents FCCU from entering the CONFIG state until FCCU is reset. For information about putting FCCU in configuration, see Prepare FCCU for configuration and Configure FCCU. **NOTE** You can write to this field when FCCU is in any state, but the lock will not get into effect until FCCU is in NORMAL state. FFh: Lock. Any other value: Do nothing.

66.12.1.24 Delta T (DELTA_T)

Offset

Register	Offset
DELTA_T	F8h

Function

The DELTA_T register is used for programming the value of delta_T constant (see DELTA_T), in microseconds.

NOTE

This register can be written only when the FCCU is in the CONFIG state.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		Reserved													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		DELTA_T													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-30	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
29-16 —	Reserved
15-14 —	Reserved
13-0 DELTA_T	Minimum Fault-Output (EOUT) Timer Interval Applies only to Bi-Stable mode (CFG[FOM]). Controls the minimum amount of time (T_{min}) that the fault-output (EOUT) timer runs according to this equation: T_{min} = 250 µs + DELTA_T **NOTE** The durations shown for the DELTA_T values are for CLKSAFE clock signals running at exactly 48 MHz. However, the CLKSAFE signals are sourced from an internal chip clock signal (for example, IRC)—or an integer division of it (see the chip-specific FCCU information)—that has a trimmed frequency variation. Therefore, the DELTA_T durations may vary. See your chip's data sheet for the trimmed frequency variation (for example, δF_{var}).

66.12.1.25 Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0 - IRQ_ALARM_EN3)

Offset

Register	Offset
IRQ_ALARM_EN0	FCh
IRQ_ALARM_EN1	100h
IRQ_ALARM_EN2	104h
IRQ_ALARM_EN3	108h

Function

See IRQ_ALARM_ENa[IRQENn].

Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0 - IRQ_ALARM_EN3) register	Offset	IRQ_ALARM_ENa[IRQENn] fields (value of n)	
		Most significant (leftmost) bit	Least significant (rightmost) bit
0	0FCh	31	0

Table continues on the next page...

Table continued from the previous page...

Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0 - IRQ_ALARM_EN3) register	Offset	IRQ_ALARM_ENa[IRQENn] fields (value of n)	
		Most significant (leftmost) bit	Least significant (rightmost) bit
1	100h	63	32
2	104h	95	64
3	108h	127	96

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IRQEN 31	IRQEN 30	IRQEN 29	IRQEN 28	IRQEN 27	IRQEN 26	IRQEN 25	IRQEN 24	IRQEN 23	IRQEN 22	IRQEN 21	IRQEN 20	IRQEN 19	IRQEN 18	IRQEN 17	IRQEN 16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IRQEN 15	IRQEN 14	IRQEN 13	IRQEN 12	IRQEN 11	IRQEN 10	IRQEN 9	IRQEN 8	IRQEN 7	IRQEN 6	IRQEN 5	IRQEN 4	IRQEN 3	IRQEN 2	IRQEN 1	IRQEN 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 IRQENn	Non-critical Alarm-State Interrupt-Request Enable n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the interrupt request is enabled as the Alarm-state reaction for the associated non-critical fault channel (n). When the ALARM state and the Alarm-state interrupt request are enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to assert the irq_alarm signal when FCCU enters the ALARM state; irq_alarm remains asserted until FCCU is in the NORMAL state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. For the mapping of the IRQ_ALARM_ENa[IRQENn] fields among registers, see the earlier table in this register description. 0b - Disabled 1b - Enabled

66.12.1.26 Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0 - NMI_EN3)

Offset

Register	Offset
NMI_EN0	10Ch
NMI_EN1	110h
NMI_EN2	114h
NMI_EN3	118h

Function

See NMI_ENa[NMIENn].

Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0 - NMI_EN3) register	Offset	NMI_ENa[NMIENn] fields (value of n)	
		Most significant (leftmost) bit	Least significant (rightmost) bit
0	10Ch	31	0
1	110h	63	32
2	114h	95	64
3	118h	127	96

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	NMIEN31	NMIEN30	NMIEN29	NMIEN28	NMIEN27	NMIEN26	NMIEN25	NMIEN24	NMIEN23	NMIEN22	NMIEN21	NMIEN20	NMIEN19	NMIEN18	NMIEN17	NMIEN16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	NMIEN15	NMIEN14	NMIEN13	NMIEN12	NMIEN11	NMIEN10	NMIEN9	NMIEN8	NMIEN7	NMIEN6	NMIEN5	NMIEN4	NMIEN3	NMIEN2	NMIEN1	NMIEN0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0	Non-critical Fault-State Non-maskable-Interrupt-Request Enable n

Table continues on the next page...

Provided under NDA only
CONTAINS PROPRIETARY
Beijing Autom
c5c462e5-9

Field	Function
NMIENn	Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Controls whether the non-maskable interrupt request is enabled as a Fault-state reaction for the associated non-critical fault channel (n). When the non-maskable interrupt request is enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to assert the NMIOUT signal when FCCU enters the FAULT state; NMIOUT remains asserted until FCCU exits FAULT state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. For the mapping of the NMI_ENa[NMIENn] fields among registers, see the earlier table in this register description. 0b - Disabled 1b - Enabled

66.12.1.27 Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0 - EOUT_SIG_EN3)

Offset

Register	Offset
EOUT_SIG_EN0	11Ch
EOUT_SIG_EN1	120h
EOUT_SIG_EN2	124h
EOUT_SIG_EN3	128h

Function

See EOUT_SIG_ENa[EOUTENn].

Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0 - EOUT_SIG_EN3) register	Offset	EOUT_SIG_ENa[EOUTENn] fields (value of n)	
		Most significant (leftmost) bit	Least significant (rightmost) bit
0	11Ch	31	0
1	120h	63	32
2	124h	95	64
3	128h	127	96

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-390d3b6240d1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EOUTEN31	EOUTEN30	EOUTEN29	EOUTEN28	EOUTEN27	EOUTEN26	EOUTEN25	EOUTEN24	EOUTEN23	EOUTEN22	EOUTEN21	EOUTEN20	EOUTEN19	EOUTEN18	EOUTEN17	EOUTEN16
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EOUTEN15	EOUTEN14	EOUTEN13	EOUTEN12	EOUTEN11	EOUTEN10	EOUTEN9	EOUTEN8	EOUTEN7	EOUTEN6	EOUTEN5	EOUTEN4	EOUTEN3	EOUTEN2	EOUTEN1	EOUTEN0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 EOUTENn	Non-critical Fault-State EOUT Signaling Enable n Writable only when FCCU is in the CONFIG state. Changed by FCCU to its reset value when a Configuration-state timeout occurs. Applies only when the EOUT signals are active (CFG[FCCU_SET_AFTER_RESET]). When FCCU is configured for Bi-Stable fault-output mode (CFG[FOM]), controls whether EOUT signaling is enabled as a Fault-state reaction for the associated non-critical fault channel (n). (For other fault-output modes, EOUT signaling is always enabled, regardless of the value of this field.) When EOUT signaling is enabled for an enabled non-critical fault channel, a fault on that channel causes FCCU to indicate the faulty condition on the EOUT[1:0] signals when FCCU enters the FAULT state. For information on how to configure the non-critical fault channels, see Configure the non-critical fault channels. For all fault-output modes, also controls whether FCCU asserts the FIF signal when a fault on the associated non-critical fault channel (n) causes FCCU to enter the FAULT state. **NOTE** For all fault-output modes, you must set this field to enabled to ensure that FCCU asserts the FIF signal when FCCU enters FAULT state as the result of a fault on the associated non-critical fault channel (n) so the FOSU module does not mistakenly generate a destructive chip reset. For the mapping of the EOUT_SIG_ENa[EOUTENn] fields among registers, see the earlier table in this register description. 0b - In Bi-Stable fault-output mode, both EOUT signaling and FIF assertion are disabled; in other fault-output modes, EOUT signaling is enabled and FIF assertion is disabled. 1b - Both EOUT signaling and FIF assertion are enabled in all fault-output modes.

66.12.1.28 Alarm-State Timer (TMR_ALARM)

Offset

Register	Offset
TMR_ALARM	12Ch

Function

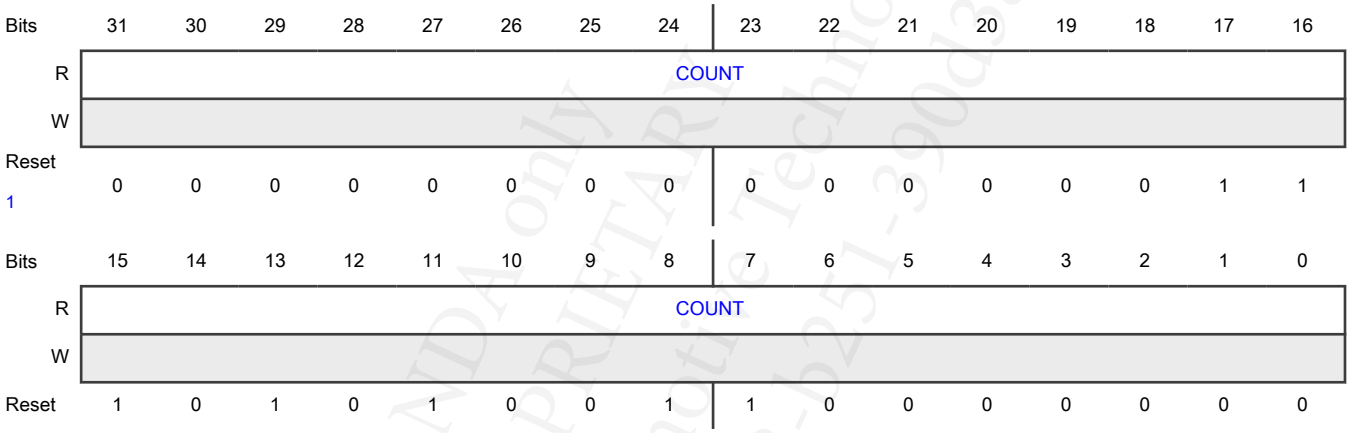
See TMR_ALARM[COUNT].

This table shows how the Alarm-state timer's state and value vary by FCCU state:

Table 554. TMR_ALARM reset value

FCCU state	Timer state	Timer value
CONFIG	Idle	0000_0000h
NORMAL	Idle	Initial value: NCF_TO[TO]
ALARM	Running	Value when read
FAULT	Idle	End of count

Diagram



1. The default reset value is provided by NCF_TO[TO]. See Table 554 for the reset value at different FCCU states.

Fields

Field	Function
31-0 COUNT	Alarm-State Timer Count Specifies the value of the Alarm-state timer in CLKSAFE periods.

66.12.1.29 Configuration-State Timer (TMR_CFG)

Offset

Register	Offset
TMR_CFG	134h

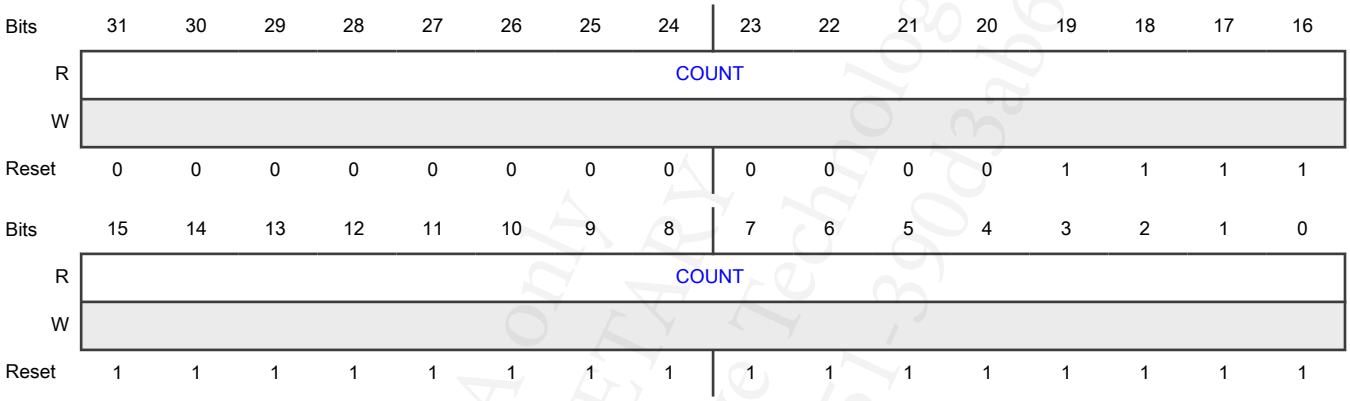
Function

See TMR_CFG[COUNT].

This table shows how the Configuration-state timer's state and value vary by FCCU state:

FCCU state	Timer state	Timer value
CONFIG	Running	Value when read
NORMAL	Idle	000F_FFFFh
ALARM	Idle	000F_FFFFh
FAULT	Idle	000F_FFFFh

Diagram



Fields

Field	Function
31-0 COUNT	Configuration-State Timer Count Specifies the value of the Configuration-state timer in CLKSAFE periods.

66.12.1.30 Fault-Output Timer (TMR_ETMR)

Offset

Register	Offset
TMR_ETMR	138h

Function

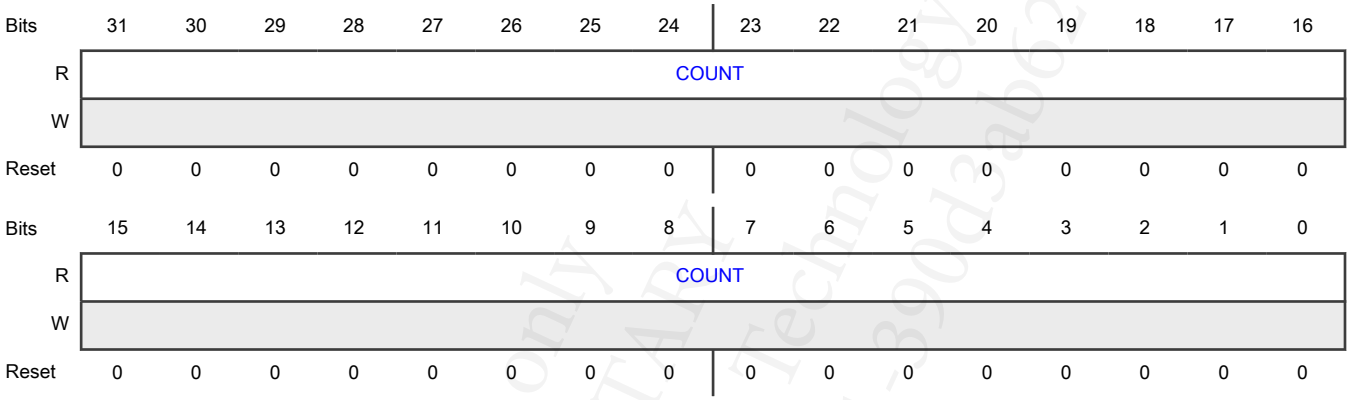
See TMR_ETMR[COUNT].

This table shows how the fault-output timer's state and value vary by FCCU state:

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4ca8-b251-39003ab624c1

FCCU state	Timer state (value)
CONFIG	Idle (0000_0000h)
NORMAL	Idle (0000_0000h)
ALARM	Idle (0000_0000h)
FAULT	Running (value when read)

Diagram



Fields

Field	Function
31-0 COUNT	Fault-Output Timer Count Specifies the value of the fault-output timer in CLKSAFE periods.

66.12.2 Configuration registers

66.12.2.1 Definition

Configuration registers are registers that:

- Let you configure FCCU's Alarm-state timer interval, fault channels, and fault-output (EOUT) signals.
- You can write to only when the configuration is not locked and FCCU is in CONFIG state (see Put FCCU in the CONFIG state).
- Save the values you write to them while FCCU is in CONFIG state only after you manually put FCCU in NORMAL state. If FCCU automatically leaves CONFIG state and enters NORMAL state because the configuration-timer interval (CFG_TO[TO]) expires (called a Configuration-state timeout), FCCU changes the value of the Configuration (CFG) register to its Configuration-state-timeout value and the value of each of the other configuration registers to its reset value; FCCU also changes the value of the Configuration-State Timeout Interval (CFG_TO) register to its reset value. For information on the Configuration-state timeout value, see CFG register bit value at different events.

66.12.2.2 Configuration registers

Following is the list of configuration registers in this module. They are listed in an offset order from lowest to highest.

- Configuration (CFG)

- Non-critical Fault Configuration (NCF_CFGa)
- Non-critical Fault-State Configuration (NCFS_CFG0 - NCFS_CFG7)
- Non-critical Fault Enable (NCF_E0 - NCF_E3)
- Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0 - NCF_TOE3)
- Non-critical-Fault Alarm-State Timeout Interval (NCF_TO)
- Delta T (DELTA_T)
- Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0 - IRQ_ALARM_EN3)
- Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0 - NMI_EN3)
- Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0 - EOUT_SIG_EN3)

66.12.3 CFG register bit value at different events

In this chip, there are no events that affect the bits in the configuration register.

66.13 FOSU

The FOSU provides a supervision of the primary fault notification path by analyzing FCCU's behavior for correctness. It waits for any reaction of the FCCU in a fixed time window after a fault is signaled.

The intention of the FOSU is to provide a secondary fault reaction path in most cases when the FCCU fails but not to needlessly propagate a fault which is already handled by the FCCU in a full chip reset. Only a failed primary fault reaction (that is, FCCU's failure) is a reason for the secondary reaction to take over (and generate a destructive reset request).

There is a 'do nothing' input coming from FCCU which indicates that the FCCU is programmed for no reaction for ALL FAULTS. It is a "static" input in the sense that it does not change after FCCU configuration. The FOSU masks the incoming faults with the 'do nothing' control from the FCCU, meaning that a fault is not captured by the FOSU if the 'do nothing' signal is asserted (that is, a disabled fault). There is no minimum pulse width requirement on the fault indication other than what is required by the technology, which is the same as that of the FCCU. FOSU does not monitor FCCU for the case of faults occurring during the CONFIG state.

The FOSU contains a timer with a duration of FOSU_COUNT, driven by CLKSAFE. The timer is initialized and started on any captured, enabled fault. While the timer is running, any subsequent captured fault will neither restart nor reinitialize the timer. The timer is stopped when the FCCU shows any of the following reactions (the FOSU does not check whether the reaction is the configured one for the faults which occurred):

- Reset: short functional reset
- IRQ (triggered by ALARM state)
- NMI
- Error out triggered (by FCCU or by SW)

When the timer is stopped, the fault capture logic is cleared to ensure that the timer is not restarted because of faults still 'stuck' in the capture logic. The timer is then restarted by the next new failure indication. When the timer expires, the FOSU's failure indicator output is asserted after it ensures that the fault is enabled and the static "fccu program to do nothing" signal is deasserted. This is because FCCU uses settings after it exits CONFIG state, even if fault captured before the exit.

The FOSU's failure indicator output is connected to one of the MC_RGM's 'destructive' reset inputs, so its assertion will cause a reset sequence to be initiated starting at DEST0. The FOSU module is reset with the same reset as is used by the FCCU. When this reset is asserted, the FOSU's capture logic is cleared, its timer is kept stopped and in a non-expired state, and its failure indicator output is deasserted.

NOTE

FOSU is triggered on assertion of enabled fault. In case the triggering fault is disabled, FOSU times out without reaction.

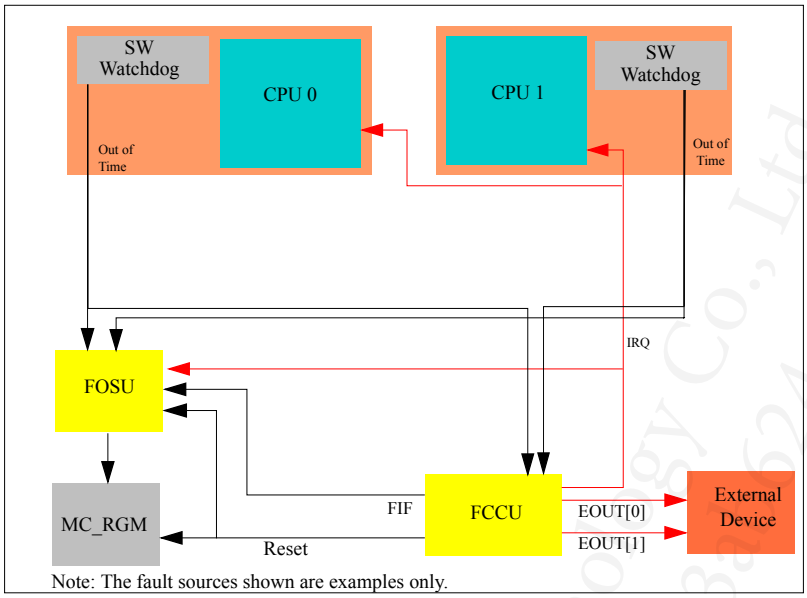


Figure 559. FOSU connections to the FCCU and MC_RGM

66.14 Use cases and limitations

Configuration guidelines

Follow these guidelines to configure FCCU:

- If you want FCCU to react to a fault on a non-critical fault channel:
  - Enable the channel (Non-critical Fault Enable (NCF_E0 - NCF_E3)).
  - Enable at least one type of Fault-state reaction for the channel: chip reset (Non-critical Fault-State Configuration (NCFS_CFG0 - NCFS_CFG7)), fault-output (EOUT) signaling (Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0 - EOUT_SIG_EN3)), or non-maskable interrupt (Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0 - NMI_EN3)).
  - If you enable chip reset as the type of Fault-state reaction for the channel (Non-critical Fault-State Configuration (NCFS_CFG0 - NCFS_CFG7)), enable either ALARM state (Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0 - NCF_TOE3)) or at least one other type of Fault-state reaction for the channel: fault-output (EOUT) signaling (Non-critical Fault-State EOUT Signaling Enable (EOUT_SIG_EN0 - EOUT_SIG_EN3)) or non-maskable interrupt (Non-critical Fault-State Non-maskable-Interrupt-Request Enable (NMI_EN0 - NMI_EN3)).
  - If you enable ALARM state for the channel (Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0 - NCF_TOE3)), enable the Alarm-state reaction (Non-critical Alarm-State Interrupt-Request Enable (IRQ_ALARM_EN0 - IRQ_ALARM_EN3)).
  - If you enable ALARM state for the channel (Non-critical-Fault Alarm-State Timeout Enable (NCF_TOE0 - NCF_TOE3)), make sure the Alarm-state timer interval (Non-critical-Fault Alarm-State Timeout Interval (NCF_TO)) is less than the FOSU module's timer interval; otherwise, FOSU generates a chip reset every time a fault occurs on the channel. The FOSU timer interval (FOSU_COUNT) is chip-specific. See the chip-specific FCCU information.

NOTE

If you enable a non-critical fault channel but disable all reactions for that channel, FCCU changes state if necessary but doesn't perform any reaction because reactions are disabled. If you enable reactions for a non-critical fault channel but disable that channel, and FCCU is in NORMAL state when a fault occurs on the channel, FCCU doesn't enter ALARM or FAULT state and therefore doesn't perform any reaction.

Recommendations to configure FCCU

1. After a power on, or 'destructive' reset (when initiated by the assertion of the chip reset pin, RESET_B), where both system and FCCU are reset, the following steps could be followed to configure FCCU:
    a. Check and clear any pending fault status
    b. Verify FCCU is in NORMAL state, else repeat step(a) above
    c. Configure FCCU
2. After any 'functional' reset of the system, arising out of a reset request from FCCU or other sources, the following steps could be followed to reconfigure FCCU:
    a. If active, wait for the Error out T_{min} to expire
    b. Check and clear fault status
    c. Error pin moves to "non faulty" state, once fault status is cleared and T_{min} expires
    d. Verify FCCU is in NORMAL state, else repeat step(a) above
    e. Read and verify value in NCF_En
    f. Reconfigure FCCU, if necessary

66.15 Glossary

CF	Critical fault
EOUT	Error out
FOSU	FCCU output supervision unit
FSM	Finite state machine
intf	Interface
IRQ	Interrupt request
NCF	Non-critical fault
NMI	Non-maskable interrupt

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 67
Clock Monitoring Unit-Frequency Check (CMU_FC)

67.1 Chip-specific CMU_FC information

67.1.1 Overview

The CMU module consists of both Frequency Check (CMU_FC) and Frequency Meter (CMU_FM) submodules. This device instantiates both of these submodules.

For details on clocks monitored by each CMU instance, see the "Clock input sources" table in the Clocking chapter.

CMU interrupts

CMU_FC and CMU_FM instances each have an Interrupt Enable Register (IER) for enabling module interrupts. However, enabled CMU_FC and CMU_FM interrupt requests do not map to system-level interrupts.

- CMU_FC interrupt requests map to system-level FCCU faults as events.
- CMU_FM interrupt requests do not map to the system level at all (not as interrupts or faults).

67.2 Introduction

CMU_FC ensures the clock integrity of selected clocks and allows continuous clock integrity verification.

CMU_FC checks if the frequency of a monitored clock (*monitored_clock*) is within a programmable frequency range specified by the user. If the frequency is outside the specified limits, it could lead to performance, protocol, and timing failures. CMU_FC requires a clock used as a base reference for the frequency check operation. This reference clock (*reference_clock*) must be within documented limits for correct CMU_FC operation.

67.2.1 Basic operation

CMU_FC counts clock cycles of the monitored clock during a user programmable time duration of *n* clock cycles of the reference clock. There is a comparison between the final count of the monitored clock and the user programmable upper and lower threshold count limits. The control logic determines whether the cycle count is within the programmed limits. If not it sets the Frequency Higher than High frequency reference threshold event status (FHH) field or the Frequency Lower than Low frequency reference threshold event status (FLL) field in the Status Register (SR). If you enable interrupts, an interrupt asserts when the status flag sets.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a1b-b251-390d8ab624c1

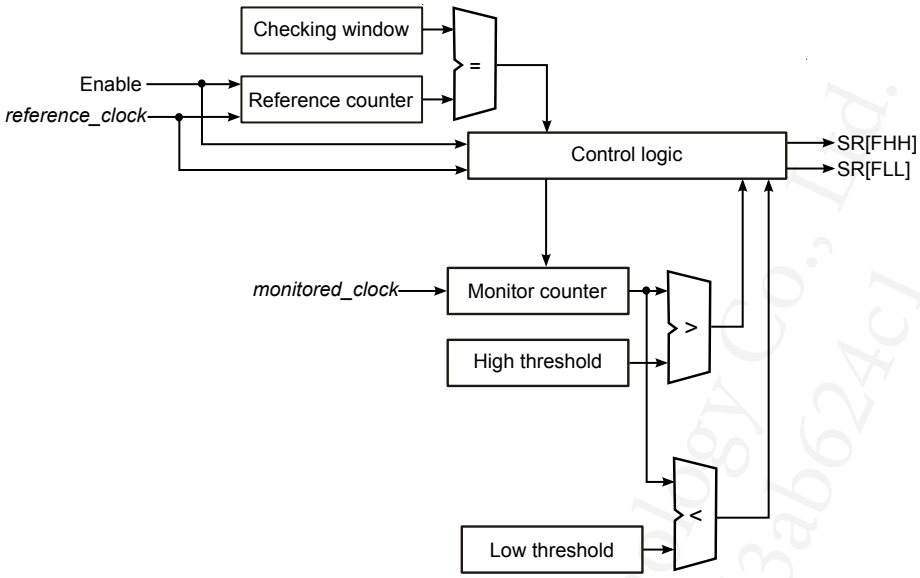


Figure 560. CMU_FC block diagram

CMU_FC cannot detect the following:

- Monitored clock duty cycle variations
- Instantaneous monitored clock frequency variations that do not cross upper or lower threshold limits

NOTE

CMU_FC has internal reset synchronizer logic for reference clock and monitored clock domains. Synchronization may take up to 10 destination clock cycles to complete.

67.2.2 Features

The CMU_FC features are:

- Programmable duration of reference clock cycles
- Initiates an event if a monitored clock frequency is higher than high frequency reference (FHH)
- Initiates an event if a monitored clock frequency is lower than low frequency reference (FLL)
- Timeout functionality generates an FLL event if a monitored clock stops functioning
- Masking of FHH event interrupt
- Masking of FLL event interrupt

Table 555. Clock description

Clock	I/O	Description
reference_clock	I	Clock signal that the module uses as a reference to evaluate *monitored_clock*
monitored_clock	I	Clock signal on which frequency check is performed
bus_clock	I	Clock signal on which register read/write operation is performed

NOTE

See the Clocking chapter for details of monitored and reference clocks used on this chip.

67.3 CMU_FC register descriptions

67.3.1 CMU_FC memory map

This section describes the address order of all the CMU_FC registers. Each description includes a standard register diagram and associated field descriptions.

CMU_FC_0 base address: 4005_C000h

CMU_FC_5 base address: 4005_C0A0h

CMU_FC_6 base address: 4005_C0C0h

CMU_FC_7 base address: 4005_C0E0h

CMU_FC_8 base address: 4005_C100h

CMU_FC_9 base address: 4005_C120h

CMU_FC_10 base address: 4005_C140h

CMU_FC_11 base address: 4005_C160h

CMU_FC_12 base address: 4005_C180h

CMU_FC_13 base address: 4005_C1A0h

CMU_FC_14 base address: 4005_C1C0h

CMU_FC_15 base address: 4005_C1E0h

CMU_FC_16 base address: 4005_C200h

CMU_FC_17 base address: 4005_C220h

CMU_FC_18 base address: 4005_C240h

CMU_FC_20 base address: 4005_C280h

CMU_FC_21 base address: 4005_C2A0h

CMU_FC_22 base address: 4005_C2C0h

CMU_FC_27 base address: 4005_C360h

CMU_FC_28 base address: 4005_C380h

CMU_FC_39 base address: 4005_C4E0h

CMU_FC_46 base address: 4005_C5C0h

CMU_FC_47 base address: 4005_C5E0h

CMU_FC_48 base address: 4005_C600h

CMU_FC_49 base address: 4005_C620h

CMU_FC_50 base address: 4005_C640h

CMU_FC_51 base address: 4005_C660h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Offset	Register	Width (In bits)	Access	Reset value
0h	Global Configuration Register (GCR)	32	RW	0000_0000h
4h	Reference Count Configuration Register (RCCR)	32	RW	0000_0000h
8h	High Threshold Configuration Register (HTCR)	32	RW	00FF_FFFFh
Ch	Low Threshold Configuration Register (LTCR)	32	RW	0000_0000h
10h	Status Register (SR)	32	W1C	0000_0000h
14h	Interrupt Enable Register (IER)	32	RW	0000_0000h

67.3.2 Global Configuration Register (GCR)

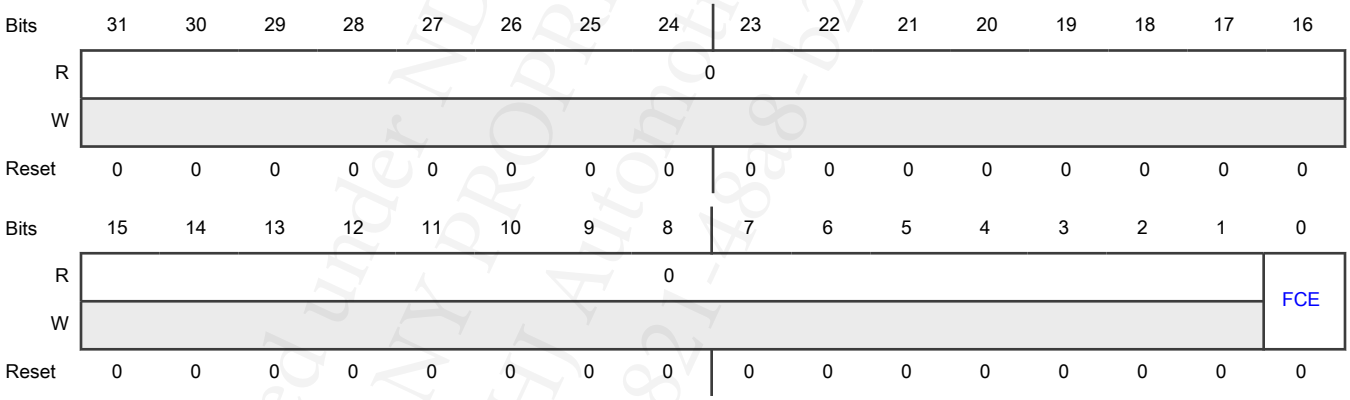
Offset

Register	Offset
GCR	0h

Function

Controls module level configurations such as enabling frequency check.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 FCE	Frequency Check Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Starts or stops frequency checking. FCE is disabled by default. Software can enable FCE at any time. To stop the ongoing operation, write 0 to FCE only when SR[RS] = 1. 0b - Stops frequency checking 1b - Starts frequency checking

67.3.3 Reference Count Configuration Register (RCCR)

Offset

Register	Offset
RCCR	4h

Function

Programs reference count duration of the frequency check window.

NOTE

Write to RCCR only when GCR[FCE] = 0. A bus transfer error results if software writes RCCR when GCR[FCE] = 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REF_CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 REF_CNT	Reference clock count Total number of counts of *reference_clock* for which frequency check runs. This field defines the duration of one frequency check window. See Programming RCCR[REF_CNT] for RCCR calculation.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c462e5-4821-48a8-b251-390d3ab624c1

67.3.4 High Threshold Configuration Register (HTCR)

Offset

Register	Offset
HTCR	8h

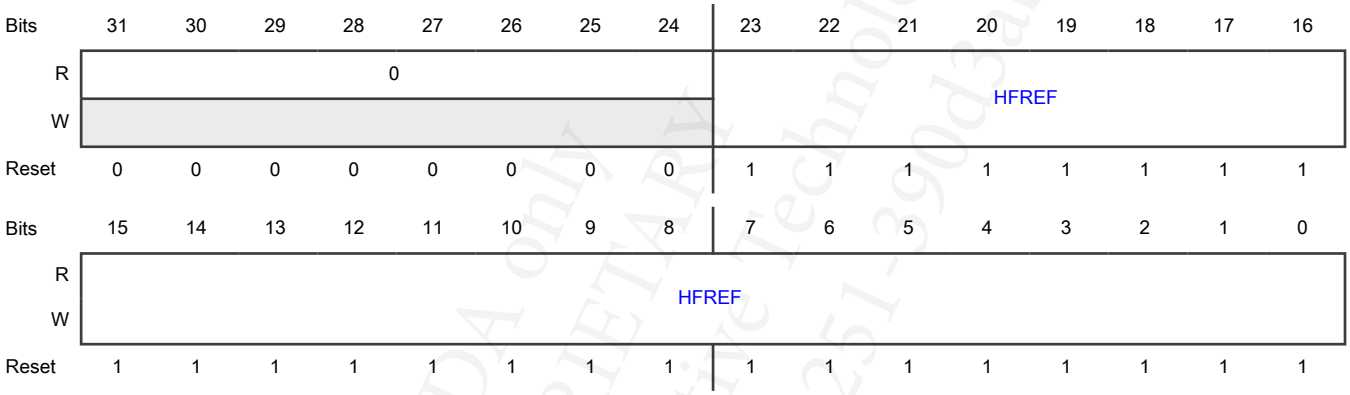
Function

Determines the high threshold limit of the monitored clock counter.

NOTE

Write HTCR only when GCR[FCE] = 0. A bus transfer error results if software writes HTCR when GCR[FCE] = 1.

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 HFREF	High frequency reference threshold HFREF determines the high reference value for the monitored clock frequency. See Programming HFREF and LFREF for HFREF calculation. **NOTE** Do not program HFREF to a value greater than 0x00FFFFFC.

67.3.5 Low Threshold Configuration Register (LTCR)

Offset

Register	Offset
LTCR	Ch

Function

Determines the low threshold limit of the monitored clock counter.

NOTE

Write LTCR only when GCR[FCE] = 0. A bus transfer error results if software writes LTCR when GCR[FCE] = 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								LFREF							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LFREF															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24 —	Reserved
23-0 LFREF	Low Frequency Reference Threshold LFREF determines the low reference value for the monitored clock frequency. See Programming HFREF and LFREF for LFREF calculation. **NOTE** Do not program LFREF to a value less than 0x00000003.

67.3.6 Status Register (SR)

Offset

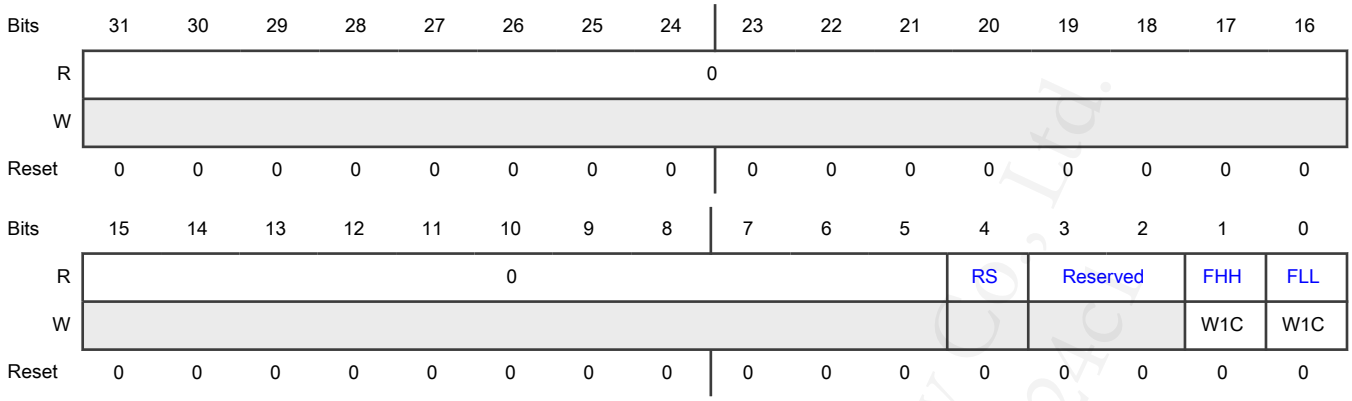
Register	Offset
SR	10h

Function

Provides the internal status of the module.

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-45ab-b251-390d1ab624c1

Diagram



Fields

Field	Function
31-5 —	Reserved
4 RS	Run Status Shows the running status of module internal operation. After enabling the frequency check operation (GCR[FCE] = 1), there is a fixed delay before this field shows running status. If the system wants to disable the frequency check operation, then the software should read this register and write GCR[FCE] = 0 only when this field is 1. 0b - Frequency check stopped 1b - Frequency check running
3-2 —	Reserved
1 FHH	Frequency higher than high frequency reference threshold event status Hardware writes 1 to FHH when the monitored clock frequency becomes higher than the high threshold value in HTCR[HFREF]. FHH clears when software writes 1. 0b - No FHH event 1b - FHH event occurred
0 FLL	Frequency lower than low frequency reference threshold event status Hardware writes 1 to FLL when the monitored clock frequency becomes lower than the low threshold value in LTCR[LFREF]. FLL clears when software writes 1. 0b - No FLL event 1b - FLL event occurred

Preliminary NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625e-4821-48a8-b253-390d8ab624c1

67.3.7 Interrupt Enable Register (IER)

Offset

Register	Offset
IER	14h

Function

Enables CMU_FC interrupts.

NOTE

Write IER only when GCR[FCE] = 0. A bus transfer error results if software writes IER when GCR[FCE] = 1.

See Clocking chapter for interrupt usage information based on CMU_FC instances.

NOTE

Each module instance supports a different number of registers.

Register supported	Register not supported
CMU_FC_0_IER	—
CMU_FC_5_IER	—
—	CMU_FC_6_IER
CMU_FC_7_IER	—
—	CMU_FC_8_IER
—	CMU_FC_9_IER
CMU_FC_10_IER	—
CMU_FC_11_IER	—
CMU_FC_12_IER	—
CMU_FC_13_IER	—
CMU_FC_14_IER	—
CMU_FC_15_IER	—
CMU_FC_16_IER	—
CMU_FC_17_IER	—
CMU_FC_18_IER	—

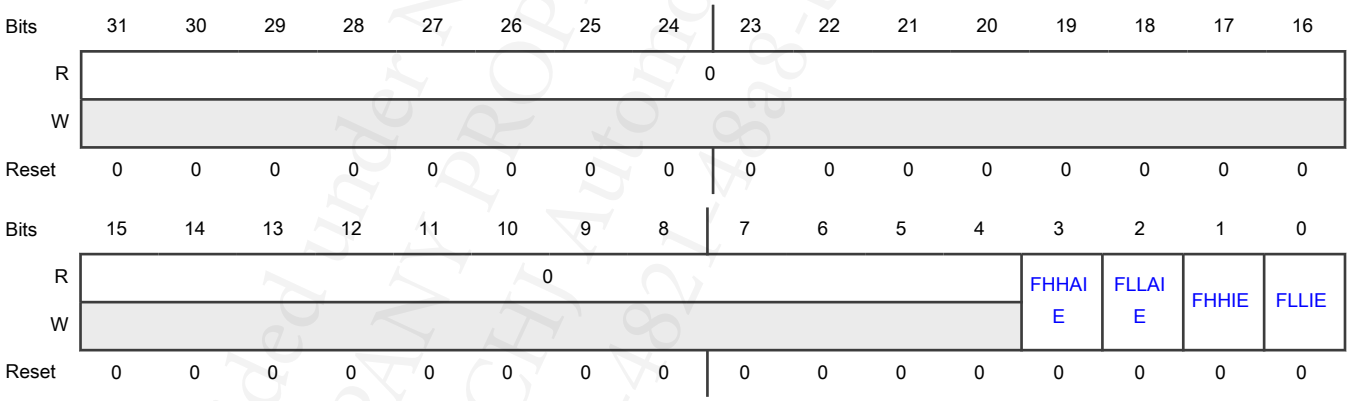
Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4818-b251-39ed3ab624c1

Table continued from the previous page...

Register supported	Register not supported
CMU_FC_20_IER	—
CMU_FC_21_IER	—
CMU_FC_22_IER	—
CMU_FC_27_IER	—
CMU_FC_28_IER	—
CMU_FC_39_IER	—
CMU_FC_46_IER	—
CMU_FC_47_IER	—
CMU_FC_48_IER	—
CMU_FC_49_IER	—
CMU_FC_50_IER	—
CMU_FC_51_IER	—

Diagram



Fields

Field	Function
31-4 —	Reserved
3	Frequency Higher than High Frequency Reference Threshold Asynchronous Interrupt Enable

<table>
<tr><th>Field</th><th>Function</th></tr>
<tr><td>FHHAIE</td><td>FHHAIE enables FHH asynchronous interrupt at the module boundary.
NOTE
This field is not supported in every instance. The following table includes only supported registers.
<table>
<tr><th>Field supported in</th><th>Field not supported in</th></tr>
<tr><td>CMU_FC_0_IER</td><td>—</td></tr>
<tr><td>CMU_FC_5_IER</td><td>—</td></tr>
<tr><td>—</td><td>CMU_FC_7_IER</td></tr>
<tr><td>—</td><td>CMU_FC_10_IER</td></tr>
<tr><td>—</td><td>CMU_FC_11_IER</td></tr>
<tr><td>—</td><td>CMU_FC_12_IER</td></tr>
<tr><td>—</td><td>CMU_FC_13_IER</td></tr>
<tr><td>—</td><td>CMU_FC_14_IER</td></tr>
<tr><td>—</td><td>CMU_FC_15_IER</td></tr>
<tr><td>—</td><td>CMU_FC_16_IER</td></tr>
<tr><td>—</td><td>CMU_FC_17_IER</td></tr>
<tr><td>—</td><td>CMU_FC_18_IER</td></tr>
<tr><td>—</td><td>CMU_FC_20_IER</td></tr>
<tr><td>—</td><td>CMU_FC_21_IER</td></tr>
<tr><td>—</td><td>CMU_FC_22_IER</td></tr>
<tr><td>—</td><td>CMU_FC_27_IER</td></tr>
<tr><td>—</td><td>CMU_FC_28_IER</td></tr>
<tr><td>—</td><td>CMU_FC_39_IER</td></tr>
<tr><td>—</td><td>CMU_FC_46_IER</td></tr>
<tr><td>—</td><td>CMU_FC_47_IER</td></tr>
</table></td></tr>
</table>

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4846-8255-590d3ab624c2

Table continued from the previous page...

Field	Function
	<table><tr><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>—</td><td>CMU_FC_48_IER</td></tr><tr><td>—</td><td>CMU_FC_49_IER</td></tr><tr><td>—</td><td>CMU_FC_50_IER</td></tr><tr><td>—</td><td>CMU_FC_51_IER</td></tr></table> 0b - Asynchronous FHH event interrupt disabled 1b - Asynchronous FHH event interrupt enabled
2 FLLAIE	Frequency Lower than Low Frequency Reference Threshold Asynchronous Interrupt Enable FLLAIE enables FLL asynchronous interrupt at the module boundary. **NOTE** This field is not supported in every instance. The following table includes only supported registers. <table><tr><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>CMU_FC_0_IER</td><td>—</td></tr><tr><td>CMU_FC_5_IER</td><td>—</td></tr><tr><td>—</td><td>CMU_FC_7_IER</td></tr><tr><td>—</td><td>CMU_FC_10_IER</td></tr><tr><td>—</td><td>CMU_FC_11_IER</td></tr><tr><td>—</td><td>CMU_FC_12_IER</td></tr><tr><td>—</td><td>CMU_FC_13_IER</td></tr><tr><td>—</td><td>CMU_FC_14_IER</td></tr><tr><td>—</td><td>CMU_FC_15_IER</td></tr><tr><td>—</td><td>CMU_FC_16_IER</td></tr><tr><td>—</td><td>CMU_FC_17_IER</td></tr><tr><td>—</td><td>CMU_FC_18_IER</td></tr></table>

Table continues on the next page...

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4886-b251-390c3ab624c1

Table continued from the previous page...

Field	Function

Field supported in	Field not supported in
—	CMU_FC_20_IER
—	CMU_FC_21_IER
—	CMU_FC_22_IER
—	CMU_FC_27_IER
—	CMU_FC_28_IER
—	CMU_FC_39_IER
—	CMU_FC_46_IER
—	CMU_FC_47_IER
—	CMU_FC_48_IER
—	CMU_FC_49_IER
—	CMU_FC_50_IER
—	CMU_FC_51_IER

0b - Asynchronous FLL event interrupt disabled

1b - Asynchronous FLL event interrupt enabled

1

FHHIE

Frequency Higher than High Frequency Reference Threshold Synchronous Interrupt Enable

FHHIE enables an FHH synchronous interrupt at the module boundary.

NOTE

This field is not supported in every instance. The following table includes only supported registers.

Field supported in	Field not supported in
—	CMU_FC_0_IER
—	CMU_FC_5_IER
CMU_FC_7_IER	—
CMU_FC_10_IER	—

Table continues on the next page...

Table continued from the previous page...

Field	Function

Field supported in	Field not supported in
CMU_FC_11_IER	—
CMU_FC_12_IER	—
CMU_FC_13_IER	—
CMU_FC_14_IER	—
CMU_FC_15_IER	—
CMU_FC_16_IER	—
CMU_FC_17_IER	—
CMU_FC_18_IER	—
CMU_FC_20_IER	—
CMU_FC_21_IER	—
CMU_FC_22_IER	—
CMU_FC_27_IER	—
CMU_FC_28_IER	—
CMU_FC_39_IER	—
CMU_FC_46_IER	—
CMU_FC_47_IER	—
CMU_FC_48_IER	—
CMU_FC_49_IER	—
CMU_FC_50_IER	—
CMU_FC_51_IER	—

0b - Synchronous FHH event interrupt disabled

1b - Synchronous FHH event interrupt enabled

0	Frequency Lower than Low Frequency Reference Threshold Synchronous Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
FLLIE	FLLIE enables an FLL synchronous interrupt at the module boundary. **NOTE** This field is not supported in every instance. The following table includes only supported registers.

Field supported in	Field not supported in
—	CMU_FC_0_IER
—	CMU_FC_5_IER
CMU_FC_7_IER	—
CMU_FC_10_IER	—
CMU_FC_11_IER	—
CMU_FC_12_IER	—
CMU_FC_13_IER	—
CMU_FC_14_IER	—
CMU_FC_15_IER	—
CMU_FC_16_IER	—
CMU_FC_17_IER	—
CMU_FC_18_IER	—
CMU_FC_20_IER	—
CMU_FC_21_IER	—
CMU_FC_22_IER	—
CMU_FC_27_IER	—
CMU_FC_28_IER	—
CMU_FC_39_IER	—
CMU_FC_46_IER	—
CMU_FC_47_IER	—

Table continues on the next page...

Table continued from the previous page...

Field	Function
	<table><tr><th>Field supported in</th><th>Field not supported in</th></tr><tr><td>CMU_FC_48_IER</td><td>—</td></tr><tr><td>CMU_FC_49_IER</td><td>—</td></tr><tr><td>CMU_FC_50_IER</td><td>—</td></tr><tr><td>CMU_FC_51_IER</td><td>—</td></tr></table> 0b - Synchronous FLL event interrupt disabled 1b - Synchronous FLL event interrupt enabled

67.4 Functional description

67.4.1 Frequency checking

Frequency checking starts when software writes GCR[FCE] = 1. The Run Status field SR[RS] shows 1 after frequency check operation starts.

CMU_FC performs continuous periodic clock checking of the monitored clock. Each checking window involves the following operations:

- Stage 1 – Reference clock counter runs for RCCR[REF_CNT] reference clock cycles. Monitored clock counter runs in parallel for the same time duration as reference clock counter.
- Stage 2 – After stage 1, the module compares the final monitored clock count against the programmed thresholds.

If the monitored clock count is greater than the high threshold value in HTCR[HFREF], an FHH event occurs.

If the monitored clock count is lower than the low threshold value in LTCR[LFREF], an FLL event occurs.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a48-b251-290d8ab624c1

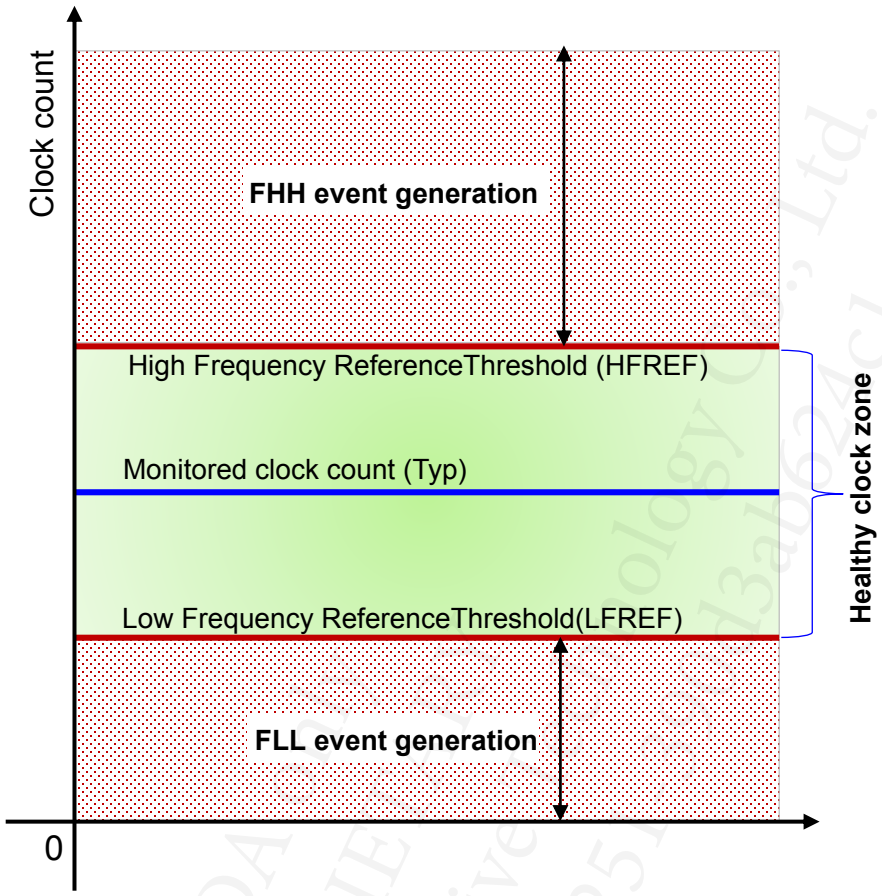


Figure 561. Frequency checking HFREF and LFREF thresholds

67.4.1.1 Monitored clock lost

If a monitored clock ceases operation before evaluation of that clock (Stage 2, see Frequency checking), CMU_FC waits for RCCR[REF_CNT] reference clock periods by extending Stage 2 before triggering an FLL event.

- If *monitored_clock* is not running when enabling CMU_FC:

  Worst case FLL response time = (2 × RCCR[REF_CNT] + 12) *reference_clock* cycles + 15 *bus_clock* cycles

- If *monitored_clock* stops when the CMU_FC is running:

  Worst case FLL response time = (2 × RCCR[REF_CNT] + 2) *reference_clock* cycles + 3 *bus_clock* cycles

67.5 Programming guidelines

67.5.1 Programming RCCR[REF_CNT]

RCCR[REF_CNT] defines the duration of the checking operation in number of *reference_clock* cycles. The minimum RCCR[REF_CNT] value can be calculated using the following formula:

$$\textbf{Minimum RCCR [REF\_CNT]} = \text{CEILING value of MAX}\left(3 \text{ X } \left(\frac{f_{reference\_clock}}{f_{bus\_clock}}\right), 8 + 5 \text{ X } \left(\frac{f_{reference\_clock}}{f_{monitored\_clock}}\right)\right)$$

...where:

- $f_{reference\_clock}$ is the frequency of the reference clock
- f_{bus_clock} is the frequency of the bus clock
- $f_{monitored\_clock}$ is the frequency of the monitored clock

Example to determine RCCR[REF_CNT]:

- f_{bus_clock} = 16 MHz
- $f_{reference\_clock}$ = 8 MHz
- $f_{monitored\_clock}$ = 48 MHz

Formula #1:

$$3 \text{ X } \left(\frac{f_{reference\_clock}}{f_{bus\_clock}} \right)$$

Inserting the values: 3 × (8 / 16) = 1.5

Formula #2:

$$8 + 5 \text{ X } \left(\frac{f_{reference\_clock}}{f_{monitored\_clock}} \right)$$

Inserting the values: 8 + 5 × (8 / 48) ≈ 8.83

MAX (1.5, 8.83): 8.83

CEILING value of MAX: 9

Therefore, minimum RCCR[REF_CNT]: 9

Programmed RCCR[REF_CNT] value must be greater than or equal to the calculated minimum value and should be decided after considering application requirements of frequency check completion response time and overall accuracy required from CMU_FC.

- Higher values of RCCR[REF_CNT] results in longer measurement window, leading to better accuracy in monitored clock check.
- Lower values of RCCR[REF_CNT] results in shorter measurement window, leading to faster FHH and FLL event response, but higher inaccuracy in reported result.

For measurement window calculation, see step 8 in Programming HFREF and LFREF

67.5.2 Programming HFREF and LFREF

You must consider the following when programming HFREF/LFREF:

- Tolerable monitored clock frequency variation
- Maximum reference clock frequency variation
- Inherent module inaccuracy

CMU_FC has an expected maximum deviation of ± 3 *monitored_clock* cycles (CMU_FC_{VAR+} = 3, CMU_FC_{VAR-} = –3).

The following procedure shows how to calculate optimum values for HTCR[HFREF] and LTCR[LFREF] (numbers shown are from example in table):

1. Determine the ideal *monitored_clock* frequency ($f_{monitored\_clock}$ (ideal), 48 MHz).
2. Determine the specified variation of the *monitored_clock* (1.1%).

3. Calculate *monitored_clock* frequency variation: .

$$f_{monitored\_clock}\ (ideal)\ \mathrm{X}\left(1 \pm variation\ (\text{step 2})\right)$$

   - $f_{monitored\_clock}$ (max) = 48 MHz × (1 + (1.1 ÷ 100)) = 48.53 MHz
   - $f_{monitored\_clock}$ (min)= 48 MHz × (1 – (1.1 ÷ 100)) = 47.47 MHz
4. Determine the ideal *reference_clock* frequency ($f_{reference\_clock}$ (ideal), 8 MHz).
5. Determine the specified variation of the *reference_clock* (3.3%).
6. Calculate ideal *reference_clock* frequency variation:

$$f_{reference\_clock}\ (ideal)\ \mathrm{X}\left(1 \pm variation\ (\text{step 5})\right)$$

   - $f_{reference\_clock}$ (max) = 8 MHz × (1 + (3.3 ÷ 100)) = 8.26 MHz
   - $f_{reference\_clock}$ (min) = 8 MHz × (1 – (3.3 ÷ 100)) = 7.74 MHz
7. Select the value of RCCR[REF_CNT] needed (80). See Programming RCCR[REF_CNT].
8. Calculate ideal measurement window:

$$\frac{RCCR\,[REF\_CNT]}{f_{reference\_clock}\ (ideal)}$$

   - 80 ÷ 8 MHz = 10000 ns
9. Calculate ideal *monitored_clock* count:

$$\frac{RCCR[REF\_CNT]}{f_{reference\_clock}\ (ideal)} \times f_{monitored\_clock}\ (ideal)$$

   - 80 ÷ 8 MHz × 48 MHz = 480.00 (round off this value, 480 in this case)
10. Calculate high threshold value (HTCR[HFREF]):

$$\text{Ceiling value of}\ \left(\frac{f_{monitored\_clock}\ (max)}{f_{reference\_clock}\ (min)} \times RCCR[REF\_CNT]\right) + CMU\_FC_{VAR+}$$

    - HTCR[HFREF] = Ceiling(48.53 MHz ÷ 7.74 MHz × 80 + 3) = 505
11. Calculate low threshold value (LTCR[LFREF]):

$$\text{Floor value of}\ \left(\frac{f_{monitored\_clock}\ (min)}{f_{reference\_clock}\ (max)} \times RCCR[REF\_CNT]\right) + CMU\_FC_{VAR-}$$

    - LTCR[LFREF] = Floor(47.47 MHz ÷ 8.26 MHz × 80 – 3) = 456

Table 556. HTCR[HFREF] and LTCR[LFREF] calculation example

Property	Value	Unit
Bus clock frequency (f_{bus_clock})	16	MHz

Table continues on the next page...

Table 556. HTCR[HFREF] and LTCR[LFREF] calculation example (continued)

Property	Value	Unit
Monitored clock frequency ($f_{monitored\_clock}$ (ideal))	48	MHz
Specified variation of monitored clock	1.1	%
Monitored clock frequency (maximum after specified variation, $f_{monitored\_clock}$ (max))	48.53	MHz
Monitored clock frequency (minimum after specified variation, $f_{monitored\_clock}$ (min))	47.47	MHz
Reference clock frequency ($f_{reference\_clock}$ (ideal))	8	MHz
Specified variation of reference clock	3.3	%
Reference clock frequency (maximum after specified variation, $f_{reference\_clock}$ (max))	8.26	MHz
Reference clock frequency (minimum after specified variation, $f_{reference\_clock}$ (min))	7.74	MHz
Minimum reference count (RCCR[REF_CNT])	9	—
Programmed reference count (RCCR[REF_CNT]) (must be ≥ minimum RCCR[REF_CNT])	80	—
Measurement window	10000.00	ns
Calculated monitored clock count	480.00	—
Calculated monitored clock Count (rounded)	480	—
Module positive variation (CMU_FC_{VAR+})	3	—
Module negative variation (CMU_FC_{VAR-})	-3	—
Programmed higher threshold (HTCR[HFREF])	505	—
Programmed lower threshold (LTCR[LFREF])	456	—

67.5.3 Module programming sequence

The recommended programming sequence for CMU_FCfrequency checking is:

1. In any order, complete the following:
   - Program RCCR[REF_CNT] to define the frequency measuring window duration in *reference_clock* cycles.
   - Program LTCR[LFREF] and HTCR[HFREF] to set the permissible frequency range of the *monitored_clock*.
   - Program IER to enable interrupt.
2. Write GCR[FCE] = 1 to start the frequency check operation.

NOTE

When GCR[FCE] = 1, you must not write to RCCR, HTCR, LTCR, or IER. Attempting to do so generates a bus transfer error.

3. To reconfigure RCCR, LTCR, HTCR , and IER or to stop frequency check operation, wait for SR[RS] = 1, then write GCR[FCE] = 0.

67.6 Glossary

FHH The state that the monitored clock frequency is higher than the high frequency reference. FHH is also the name of the register field that indicates this state.

FHH event The event that the module triggers when it detects FHH.

FLL The state that the monitored clock frequency is lower than the lower frequency reference. FLL is also the name of the register field that indicates this state.

FLL event The event that the module triggers when it detects FLL.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 68
Clock Monitoring Unit - Frequency Meter (CMU_FM)

68.1 Chip-specific CMU_FM information

68.1.1 Overview

The CMU module consists of both Frequency Check (CMU_FC) and Frequency Meter (CMU_FM) submodules. This device instantiates both of these submodules.

For details on clocks monitored by each CMU instance, see the "Clock input sources" table in the Clocking chapter.

CMU interrupts

CMU_FC and CMU_FM instances each have an Interrupt Enable Register (IER) for enabling module interrupts. However, enabled CMU_FC and CMU_FM interrupt requests do not map to system-level interrupts.

- CMU_FC interrupt requests map to system-level FCCU faults as events.
- CMU_FM interrupt requests do not map to the system level at all (not as interrupts or faults).

68.2 Introduction

CMU_FM measures the frequency of the metered clock (*metered_clock*) using another clock as reference. This reference clock (*reference_clock*) must be within documented limits for correct CMU_FM operation. CMU_FM operation is only guaranteed if a reference clock is within normal operating parameters.

68.2.1 Basic operation

The CMU_FM counts the clock cycles of a metered clock in the user programmable time duration *n* number of clock cycles of the reference clock. After completion of a metered clock operation, a register field sets, indicating the operation has completed. The count of the metered clock also updates in the Status Register (SR) after completion of a metered clock operation.

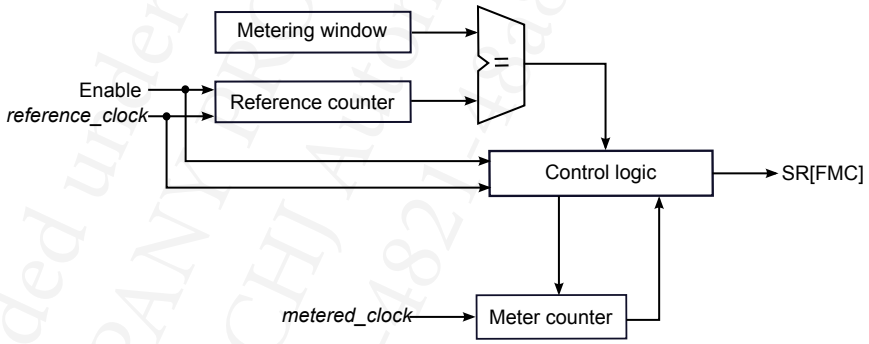


Figure 562. Block diagram

68.2.2 Features

Features of the CMU_FM are:

- Programmable duration of *reference_clock* cycles
- Timeout feature indicates loss of *metered_clock*

Table 557. Clock description

Clock	I/O	Description
reference_clock	I	Clock signal that the module uses as a reference to evaluate *metered_clock*
metered_clock	I	Clock signal on which frequency metering is performed
bus_clock	I	Clock signal on which register read/write operation is performed

NOTE

See the Clocking chapter for details of metered and reference clocks used on this chip.

68.3 CMU_FM register descriptions

68.3.1 CMU_FM memory map

This section describes the address order of all the CMU_FM registers. Each description includes a standard register diagram and associated field descriptions.

CMU_FM_1 base address: 4005_C020h

CMU_FM_2 base address: 4005_C040h

CMU_FM_3 base address: 4005_C060h

CMU_FM_4 base address: 4005_C080h

Offset	Register	Width (In bits)	Access	Reset value
0h	Global Configuration Register (GCR)	32	RW	0000_0000h
4h	Reference Count Configuration Register (RCCR)	32	RW	0000_0000h
8h	Status Register (SR)	32	W1C	0000_0000h

68.3.2 Global Configuration Register (GCR)

Offset

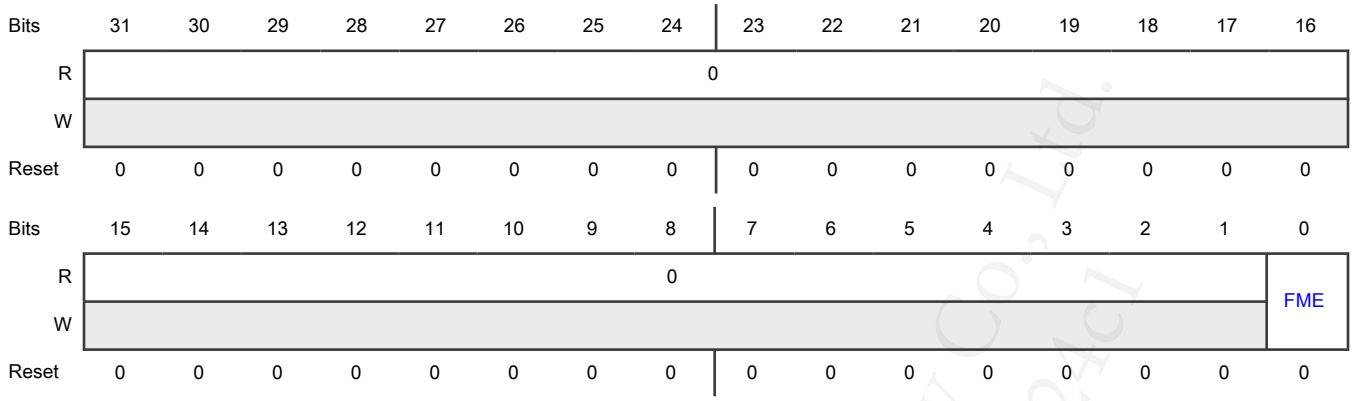
Register	Offset
GCR	0h

Function

Controls module level configurations such as enabling frequency metering.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.
c8c4625-4821-48a8-b251-3903ab624c

Diagram



Fields

Field	Function
31-1 —	Reserved
0 FME	Frequency Meter Enable Starts or stops frequency metering. FME is disabled by default. Software can enable FME at any time. To stop the ongoing operation, write 0 to FME only when SR[RS] = 1. FME automatically clears upon frequency metering completion or timeout event. 0b - Stops frequency metering 1b - Starts frequency metering

68.3.3 Reference Count Configuration Register (RCCR)

Offset

Register	Offset
RCCR	4h

Function

Programs reference count duration of the frequency meter window.

NOTE

Write to RCCR only when GCR[FME] = 0. A bus transfer error results if software writes RCCR when GCR[FME] = 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REF_CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 —	Reserved
15-0 REF_CNT	Reference Clock Count Total number of counts of *reference_clock* for which frequency metering runs. This field defines the duration of one frequency metering window. See Programming RCCR[REF_CNT] for RCCR calculation.

68.3.4 Status Register (SR)

Offset

Register	Offset
SR	8h

Function

Provides the internal status of the module and metered clock count.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MET_CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MET_CNT								0			RS	Reserved		FMTO	FMC
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c84626e5-482f-48a8-b251-390d3ab624c1

Fields

Field	Function
31-8 MET_CNT	Meter Clock Count This field stores the count value of the metered clock cycles when frequency metering operation is complete, in other words, SR[FMC] = 1. CMU_FM has a maximum deviation of ± 3 *metered_clock* cycles over the ideal *metered_clock* cycles expected by the user.
7-5 —	Reserved
4 RS	Run Status Shows the running status of module internal operation. After enabling the frequency metering operation (GCR[FME] = 1), there is a fixed delay before this field shows running status. If the system wants to disable the frequency metering operation, then the software should read this register and write GCR[FME] = 0 only when this field is 1. 0b - Frequency meter stopped 1b - Frequency meter running
3-2 —	Reserved
1 FMTO	Frequency Meter Time Out Sets if there is a loss of metered clock during the metering operation. After metering window, the module waits for a timeout duration of RCCR[REF_CNT] cycles before writing one to this field. GCR[FME] = 0 and SR[FMC] = 0 when FMTO = 1. Writing one clears this field.
0 FMC	Frequency Meter Operation Complete On completion of frequency metering operation, FMC = 1, SR[MET_CNT] field updates, and GCR[FME] clears. Writing one clears this field.

68.4 Functional description

The frequency meter operation initiates when software writes GCR[FME] = 1. When the operation starts, hardware writes SR[RS] = 1. Safety critical applications must poll SR[RS] to determine when metering operation starts.

Frequency metering window involves these operations:

1. Stage 1 — The reference clock counter runs for RCCR[REF_CNT] cycles of *reference_clock*. The metered clock counter runs in parallel for the same time duration as the *reference_clock* counter.
2. Stage 2 — At the end of stage 1, SR[FMC] = 1, SR[MET_CNT] updates with the *metered_clock* count, and CMU_FM writes GCR[FME] = 0.

68.5 Programming guidelines

68.5.1 Programming RCCR[REF_CNT]

RCCR[REF_CNT] defines the duration of the metering operation in number of *reference_clock* cycles. The minimum RCCR[REF_CNT] value can be calculated using the following formula:

$$\textbf{Minimum RCCR [REF\_CNT]} = \text{CEILING value of MAX}\left(3 \text{ X } \left(\frac{f_{reference\_clock}}{f_{bus\_clock}}\right), 6 + 4 \text{ X } \left(\frac{f_{reference\_clock}}{f_{metered\_clock}}\right)\right)$$

...where:

- $f_{reference\_clock}$ is the frequency of the reference clock
- f_{bus_clock} is the frequency of the bus clock
- $f_{metering\_clock}$ is the frequency of the metered clock

Example to determine RCCR[REF_CNT]:

- f_{bus_clock} = 16 MHz
- $f_{reference\_clock}$ = 8 MHz
- $f_{metered\_clock}$ = 48 MHz

Formula #1:

$$3 \text{ X } \left(\frac{f_{reference\_clock}}{f_{bus\_clock}}\right)$$

Inserting the values: 3 × (8 / 16) = 1.5

Formula #2:

$$6 + 4 \text{ X } \left(\frac{f_{reference\_clock}}{f_{metered\_clock}}\right)$$

Inserting the values: 6 + 4 × (8 / 48) ≈ 6.67

MAX (1.5, 6.67): 6.67

CEILING value of MAX: 7

Therefore, minimum RCCR[REF_CNT]: 7

Programmed RCCR[REF_CNT] value must be greater than or equal to the calculated minimum value and should be decided after considering application requirements of frequency metering completion response time and overall accuracy required from CMU_FM.

- Higher values of RCCR[REF_CNT] results in longer metering window, leading to better accuracy in metered clock measurement.
- Lower values of RCCR[REF_CNT] results in shorter metering window, leading to faster FMC response, but higher inaccuracy in reported result.

68.5.2 Module programming sequence

The recommended programming sequence of CMU_FM is:

1. Program RCCR[REF_CNT] to define the frequency metering window duration in *reference_clock* cycles.
2. Write GCR[FME] = 1 to start frequency metering operation. Write to RCCR is disabled after GCR[FME] = 1. Attempting a write to RCCR after GCR[FME] = 1 generates a bus transfer error.
3. To reconfigure RCCR or to stop frequency metering operation, wait for SR[RS] = 1, then write GCR[FME] = 0.

Upon completion of metering operations:

- Hardware writes SR[FMC] = 1

Provided under NDA only
COPYRIGHT PROPRIETARY
Paint Chip Automotive Technology Co., Ltd.
cac6205-4a21-43a8-b251-90cadb624c1

- SR[MET_CNT] updates with metered count value
- Hardware writes GCR[FME] = 0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 69
Selftest Top General Purpose Registers (SELFTEST_GPR_TOP)

69.1 Introduction

SELFTEST_GPR_TOP contains control and status registers that are used for the global configuration of STCU2 self-tests. The configuration includes:

- Enabling self-test in different reset domains
- Providing status of self-test enablement

See the "Software resettable domains" section in the Reset chapter for more information on this chip's available reset domains.

NOTE

The register space allotted to SELFTEST_GPR_TOP is of 32 bytes. Transfer error might not be issued for the reserved or hidden register space access.

69.2 SELFTEST_GPR_TOP register descriptions

69.2.1 SELFTEST_GPR_TOP memory map

SELFTEST_GPR_TOP base address: 4001_CFE0h

Offset	Register	Width (In bits)	Access	Reset value
0h	Reset Domain Self-test Enable (Reset_Domain_Selftest_Enable_Register)	32	RW	0000_0000h
4h	Reset Domain Self-test Enable Status (Reset_Domain_Selftest_Enable_Status_Register)	32	W1C	0000_0000h

69.2.2 Reset Domain Self-test Enable (Reset_Domain_Selftest_Enable_Register)

Offset

Register	Offset
Reset_Domain_Selftest_Enable_Register	0h

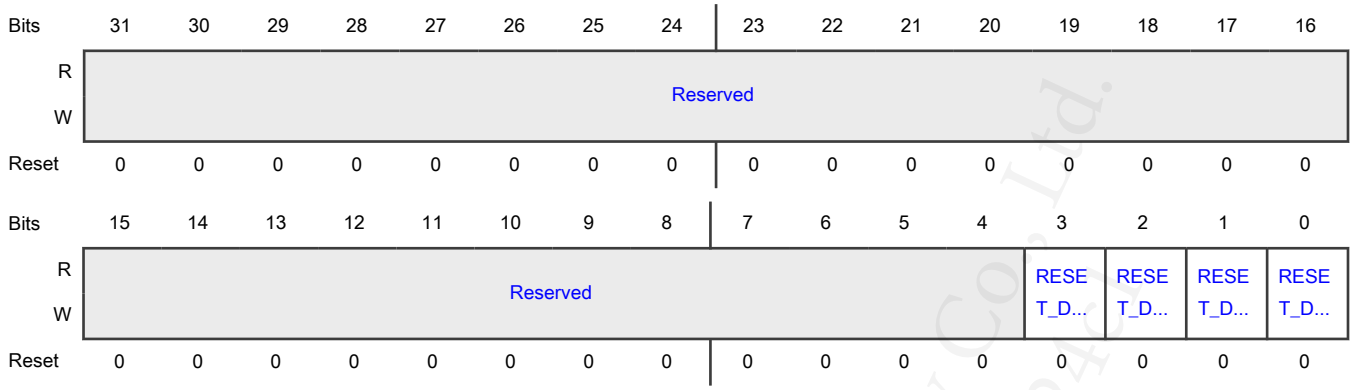
Function

Enables self-test in various reset domains on the chip.

The fields in this register automatically become 0 when STCU2 stops self-test execution.

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462c5-4821-48b-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 RESET_DOMAIN_n_SELFTEST_ENABLE	Reset Domain Self-Test Enable Enables and disables self-test on the BIST partitions of reset domain *n*. The following table shows the mapping of field and reset domain. Furthermore, for the exact mapping of fields to reset domains, see the "List of software reset domains/partitions on the chip" section in the Reset chapter.

Field name	Reset domain
RESET_DOMAIN_3_SELFTEST_ENABLE	Software-resettable domain RD3
RESET_DOMAIN_2_SELFTEST_ENABLE	Software-resettable domain RD2
RESET_DOMAIN_1_SELFTEST_ENABLE	Software-resettable domain RD1
RESET_DOMAIN_0_SELFTEST_ENABLE	Software-resettable domain RD0

0b - Disabled

1b - Enabled

69.2.3 Reset Domain Self-test Enable Status (Reset_Domain_Selftest_Enable_Status_Register)

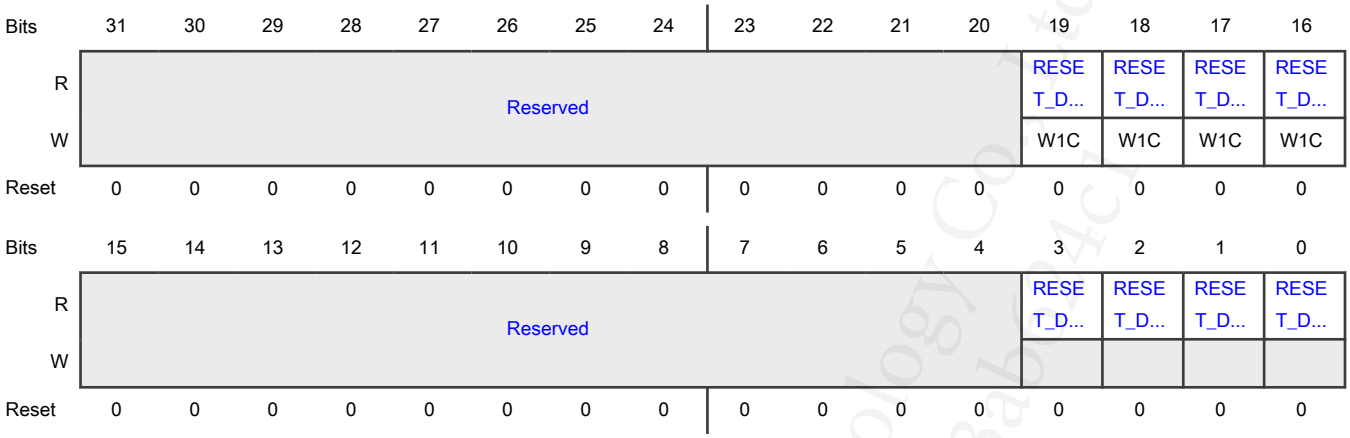
Offset

Register	Offset
Reset_Domain_Selftest_Enable_Status_Register	4h

Function

Shows the self-test enable status for various reset domains in the current or last self-test run.

Diagram



Fields

Field	Function
31-20 —	Reserved
19-16 RESET_DOMAIN_n_SELFTEST_ENABLE_LAST_RUN_STATUS	Reset Domain Self-Test Enable Last Run Status Shows whether the self-test was enabled or disabled for reset domain *n* during the last run. This field is a flag, and you can write 1 to return this field to 0. 0b - Disabled 1b - Enabled
15-4 —	Reserved
3-0 RESET_DOMAIN_n_SELFTEST_ENABLE_STATUS	Reset Domain Self-Test Enable Status Shows whether self-test of BIST partitions for reset domain *n* is enabled or disabled. 0b - Disabled 1b - Enabled

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4221-48a8-1251-390d3ab6f4c1

Chapter 70
Selftest General Purpose Registers (SELFTEST_GPR)

70.1 Introduction

SELFTEST_GPR configures the STCU2 LBIST shift cycles. Programming self-test GPRs is a part of the self-test programming sequence. See the "Self-test programming sequence" section in the STCU2 chapter.

There are multiple self-test GPR modules in the chip, each corresponding to an LBIST partition.

70.2 Memory map for self-test GPRs

The system memory map allots a 4 KB slot for self-test GPRs, divided into 128 slots of 32 bytes each. The base address of the slot is 4001_C000h. To calculate the base address of a self-test GPR block, use the following equation:

Base address of self-test GPR(*n*) = Base address of self-test GPRs in system memory map + *n* * 20h, where *n* = self-test GPR index.

If you access a self-test GPR region that corresponds to a reserved index or reserved register space, you might not receive a transfer error.

70.3 SELFTEST_GPR register descriptions

70.3.1 SELFTEST_GPR memory map

SELFTEST_GPR base address: 4001_C000h

SELFTEST_GPR_CHIPTOP base address: 4001_C560h

SELFTEST_GPR_MEM_NOC_0 base address: 4001_C520h

SELFTEST_GPR_MEM_NOC_1 base address: 4001_C540h

SELFTEST_GPR_MISC base address: 4001_C500h

SELFTEST_GPR_NOC_PART1 base address: 4001_C480h

SELFTEST_GPR_NOC_PART2 base address: 4001_C4A0h

SELFTEST_GPR_NOC_PART3 base address: 4001_C4C0h

SELFTEST_GPR_NOC_PART4 base address: 4001_C4E0h

Offset	Register	Width (In bits)	Access	Reset value
0h	Generic 0 (Generic_Reg0)	32	RW	0000_ADD5h
4h	Generic 1 (Generic_Reg1)	32	RW	0000_0000h
14h	LBIST Program (LBIST_Prog_Reg)	32	RW	0003_0050h

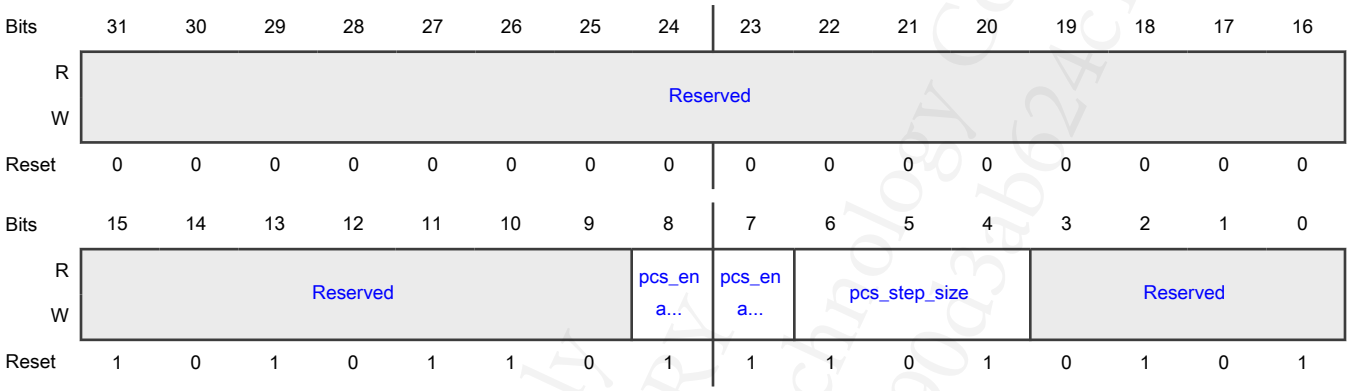
Preliminary Information
COMPANY CONFIDENTIAL
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-4821-48a8-b251-390d3ab64c1

70.3.2 Generic 0 (Generic_Reg0)

Offset

Register	Offset
Generic_Reg0	0h

Diagram



Fields

Field	Function
31-9 —	Reserved
8 pcs_enable_end	PCS enable end Enables Progressive Shift Clock Switching for LBIST at end. The feature gets enabled when this bit is set to 1. It is recommended to run LBIST patterns by enabling this feature.
7 pcs_enable_start	PCS enable start Enables Progressive Shift Clock Switching for LBIST at start. The feature gets enabled when this bit is set to 1. It is recommended to run LBIST patterns by enabling this feature.
6-4 pcs_step_size	PCS step size pcs_step_size[2:0] controls the step size for progressive increase of shift frequency: 000 - step size of 1 pattern 001 - step size of 2 patterns ... 111 - step size of 8 patterns **NOTE** The recommended value of this register is available in a separate application note. Contact your NXP sales representative.

Table continues on the next page...

Table continued from the previous page...

Field	Function
3-0 —	Reserved

70.3.3 Generic 1 (Generic_Reg1)

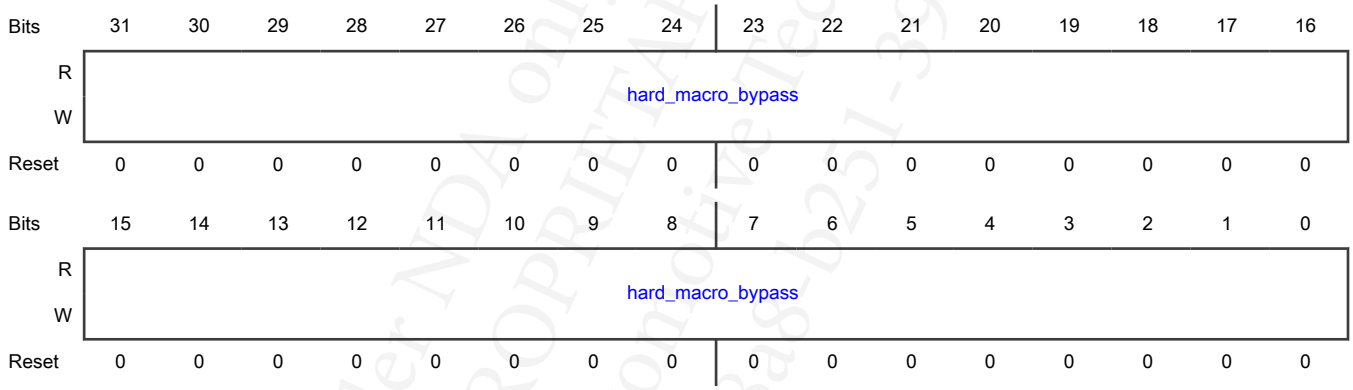
Offset

Register	Offset
Generic_Reg1	4h

Function

Provides hard macro bypass values.

Diagram



Fields

Field	Function
31-0 hard_macro_bypass	Hard Macro Bypass Controls the scan-chain bypass of hard modules. For LBIST4, you must write FFFF_FFEBh to this field. For others, program it to 0000_0000h.

70.3.4 LBIST Program (LBIST_Prog_Reg)

Offset

Register	Offset
LBIST_Prog_Reg	14h

Provided under NDA only
COMPANY PROPRIETARY
Beijing HJ Automotive Technology Co., Ltd.
c8c462e5-1821-4848-b251-590d3ab624c1

Function

Configures the LBIST shift count.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved								LBIST_shift_count							
Reset	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 LBIST_shift_count	LBIST Shift Count Specifies the number of shift cycles in a scan chain in one pattern count for the LBIST partition. **NOTE** The recommended value of this register is available in a separate application note. Contact your NXP sales representative.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 71
Self-Test Control Unit (STCU2)

71.1 Chip-specific STCU2 information

71.1.1 Supported BIST sequences

NXP supports only STCU2 Built-In Self-Test (BIST) sequences that it has validated for in-field testing usage. After the chip is validated, this section is replaced with information on the supported BIST sequences.

71.1.2 Self-test overview

The chip supports two types of BISTs. The self-test infrastructure includes scan-based testing through logic BIST (LBIST) and memory array testing through memory BIST (MBIST). STCU2 divides the logic and memories being tested into multiple groups.

Each partition of digital logic has an LBIST controller that performs the scan-based test. The LBIST controller is responsible for:

- Configuring the partition in the LBIST mode
- Running the test
- Accumulating the pass or fail signature for each partition

The MBIST controller runs algorithms to detect defects in the memory array that behave as:

- Stuck-at faults
- At-speed transition faults
- Coupling faults

The MBIST controller (MCT) connects to multiple local MBIST controllers that carry out the actual memory tests. The central master MBIST controller programs the BIST instances to run memory test algorithms and collect pass or fail status from the local MBIST controllers.

STCU2 provides central control of both the LBIST controllers and the master MBIST controller. This module configures the BIST controllers at the beginning of self-test and collects and stores the final status to be read by the core. The module also schedules the sequence of LBIST self-test between partitions and MBIST instances.

Each logical partition also includes a block called self-test GPR, a set of registers that provide miscellaneous configurations or parameters for LBIST and MBIST operation.

Figure 1 shows a self-test overview.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a6c-b251-390d3a0e04c1

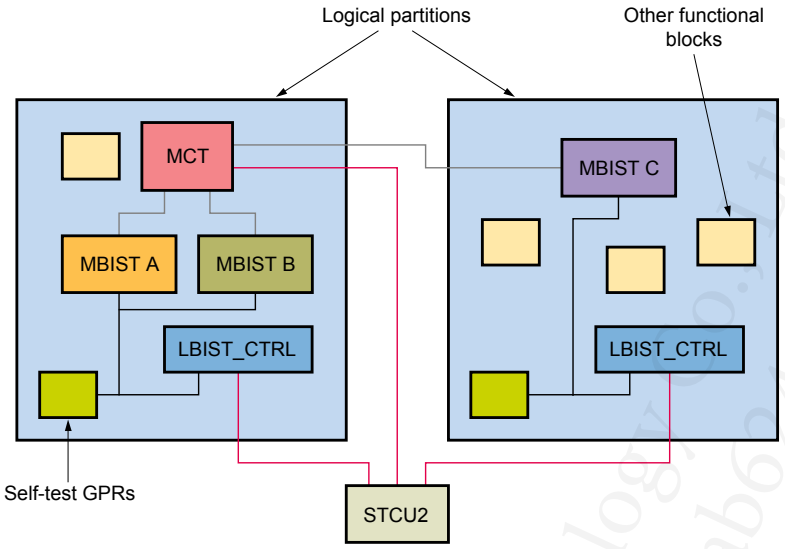


Figure 563. Self-test overview

A logical partition belongs to either the main reset domain or a software domain. The chip becomes unavailable for software application for the duration of the self-test, if software triggers a self-test on any logical partition of the main reset domain. However, a self-test running on logical partitions of a software reset domain does not affect the functionality of modules present in the main reset domain or other software reset domains. This increases the availability of the system. See the Reset chapter for a list of software reset domains available in this chip. For each LBIST and MBIST controller in the chip, STCU2 LBIST mapping and STCU2 MBIST mapping show the associated reset domain.

71.1.3 STCU2 LBIST mapping

BIST ID	BIST instance name	Description	Reset domain	Self-test GPR index
0	LBIST0	HSE_RUN	Main	0
1	LBIST1	SerDes_0	Main	1
2	LBIST2	DDR_0_PARTITION0	Main	2
3	LBIST3	DDR_0_PARTITION1	Main	3
4	LBIST4	DDR_0_PARTITION2	Main	4
5	LBIST5	CLUSTER0_CPU0	A53	5
6	LBIST6	CLUSTER0_CPU1	A53	6
7	LBIST7	CLUSTER0_GIC0	A53	7
8	LBIST8	CLUSTER1_CPU0	A53	8
9	LBIST9	CLUSTER1_CPU1	A53	9
10	LBIST10	CLUSTER1_GIC1	A53	10
11	LBIST11	CM7_0	Main	11
12	LBIST12	CM7_1	Main	12

Table continues on the next page...

Table continued from the previous page...

BIST ID	BIST instance name	Description	Reset domain	Self-test GPR index
13	LBIST13	CM7_2	Main	13
14	LBIST14	PERIPH0	Main	14
15	LBIST15	CHECKER0	Main	15
16	LBIST16	SRAM_CTRL_PARTITION0	Main	16
17	LBIST17	SRAM_CTRL_PARTITION1	Main	17
18	LBIST18	PERIPH1	Main	18
19	LBIST19	CHECKER1	Main	19
20	LBIST20	ENET_PARTITION	Main	20
21	LBIST21	SDHC_QSPI	Main	21
22	LBIST22	PERIPH2	Main	22
23	LBIST23	CHECKER2	Main	23
24	LBIST24	DDR_0_PARTITION3	Main	24
25	LBIST25	NCORE_PARTITION0	A53	25
26	LBIST26	NCORE_PARTITION1	A53	26
27	LBIST27	A53_REST	Main	27
28	LBIST28	NOC_INTERNAL	Main	28
29	LBIST29	DDR_0_PARTITION4	Main	29
30	LBIST30	PFE_PARTITION1	PFE	30
31	LBIST31	PFE_PARTITION2	PFE	31
32	LBIST32	PFE_PARTITION3	PFE	32
33	LBIST33	SerDes_1	Main	33
34	LBIST34	LLCE_PARTITION1	LLCE	34
35	LBIST35	LLCE_PARTITION2	LLCE	35
36	LBIST36	ACCEL_NOC_PARTITION1	Main	36
37	LBIST37	ACCEL_NOC_PARTITION2	Main	37
38	LBIST38	ACCEL_NOC_PARTITION3	Main	38
39	LBIST39	ACCEL_NOC_PARTITION4	Main	39
40	LBIST40	MISC_PARTITION	Main	40
41	LBIST41	MEM_NOC_PARTITION1	Main	41
42	LBIST42	MEM_NOC_PARTITION1	Main	42

Provided under NDA
COMPANY CONFIDENTIAL
Freescale Semiconductor, Inc.
c8c6f6c2-21a8-b251-3903ab624

71.1.4 STCU2 MBIST mapping

Table 558. MBIST mapping

BIST ID/memory partition (NMCUT)	BIST instance name	Reset domain
0	HSE_400	Main
1	HSE_200	Main
2	HSE_BOOT	Main
3	HSE_ARM	Main
4	SUB_DDR_SS1	Main
5	SUB_DDR_SS	Main
6	SUB_DDR_SS2	Main
7	SUB_PCIE_CORE	Main
8	SUB_PCIE_RADM	Main
9	SUB_PCIE_SLV	Main
10	SUB_PCIE_MDAUX	Main
11	SUB_PCIE_PHY0	Main
12	a53_c0l1	A53
13	a53_c0l2	A53
14	a53_gic0	A53
15	a53_c1l1	A53
16	a53_c1l2	A53
17	a53_gic1	A53
18	SUB_CM7	Main
19	SUB_CM7_1	Main
20	SUB_CM7_2	Main
21	CAN_CHECKER	Main
22	DMA	Main
23	CAN_PRIME	Main
24	ADC	Main
25	FLEXRAY_LRAM	Main
26	FLEXRAY_DRAM	Main
27	ENET	Main
28	ENET_EST	Main
29	ETF_CA53	Main
30	ETF_MISC	Main

Table continues on the next page...

Table 558. MBIST mapping (continued)

BIST ID/memory partition (NMCUT)	BIST instance name	Reset domain
31	USDHC_FIFO	Main
32	HSIO_OTC_QSPI	Main
33	NCORE_PAR	A53
34	NCORE_SEQ	A53
35	CAIU	A53
36	ETF5NOC	Main
37	ETF4NOC	Main
38	LLCE_PART1_DIV2CLK	LLCE
39	LLCE_PART2_FLXPECLK	LLCE
40	LLCE_PART2_DIV2CLK	LLCE
41	PFE_P2SYSCLK	PFE
42	PFE_P3SYSCLK	PFE
43	PFE_P3PECLK	PFE
44	PFE_P1SYSCLK	PFE
45	PFE_P1PECLK	PFE
46	PCIE_CORE	Main
47	PCIE_RADM	Main
48	SUB_PCIE_SLV	Main
49	PCIE_MDAUX	Main
50	SUB_PCIE_PHY0	Main
51	USB_TXRX	Main
52	STDBY	Main
53	DBG_TRACE	Main
54	SRAM_00_0	Main
55	SRAM_00_1	Main
56	SRAM_01_0	Main
57	SRAM_01_1	Main
58	SRAM_02_0	Main
59	SRAM_02_1	Main
60	SRAM_03_0	Main
61	SRAM_03_1	Main
62	SRAM_04_0	Main

Table continues on the next page...

Table 558. MBIST mapping (continued)

BIST ID/memory partition (NMCUT)	BIST instance name	Reset domain
63	SRAM_04_1	Main
64	SRAM_05_0	Main
65	SRAM_05_1	Main
66	SRAM_06_0	Main
67	SRAM_06_1	Main
68	SRAM_07_0	Main
69	SRAM_07_1	Main
70	SRAM_08_0	Main
71	SRAM_09_0	Main
72	SRAM_10_0	Main
73	SRAM_11_0	Main
74	SRAM_12_0	Main
75	SRAM_13_0	Main
76	SRAM_14_0	Main
77	SRAM_15_0	Main

71.1.5 Running MBIST sequences on CM7 memories

You must enable the corresponding core reset and CM7 core clock to run MBIST sequences on CM7 memories. If the corresponding CM7 core is disabled, then enable its reset and clock as follows:

1. Use the corresponding CM7 GPR to put the core in CPU_WAIT if you do not need to execute anything with the CM7 core and you need to run MBIST on its memories.
2. Ungate the clock of the corresponding CM7 core using the respective core-related registers in MC_ME.
3. Deassert the reset of the corresponding CM7 core by writing 0 to the respective field in PRST0_0.

71.1.6 Indicating fault state during the main reset domain self-test

The Software must program the MC_RGM's ERCTRL[ERASSERT] field to 1b prior to initiating the main reset domain self-test. This results in the RESET_B pin being asserted, the FCCU EOUT pins indicating the fault state, and all other pins being put into their default state as indicated by the "Pad State During Main Reset Domain Selftest" column of the "Misc With Pin Controls" tab of the IOMUX sheet.

The fault state remains asserted until the EOUT Override Clear field in SRC's FCCU_EOUT_OVERRIDE_CLEAR_REG[EOUT_OVERRIDE_CLEAR] field is programmed in the post-self-test reset sequence.

Configure the EOUT override disable during self-test field in SRC's FCCU_EOUT_OVERRIDE_CLEAR_REG[EOUT_OVERRIDE_CLEAR] field before configuring MC_RGM's ERCTRL[ERASSERT] field, if it is not required to indicate the fault state during the main reset domain self-test.

71.1.7 Error events management in main reset domain self-test

You can map error events to unrecoverable faults for the main reset domain self-test using STCU2's ERR_FM register.

71.1.8 Online self-test reset or interrupt generation

After an online self-test (LBIST or MBIST) completes or STCU2 detects an ABORT condition during self-test execution, reset is issued to all BIST partitions present inside the reset domains enabled for self-test in that run. This action resets all modules present inside those BIST partitions. Also, if the self-test includes the main reset domain, MC_RGM executes a functional reset sequence.

You must enable interrupts inside STCU2 using the RUNSW register if the self-test includes only software reset domains. STCU2 generates the interrupts and notifies the target core of those interrupts at the end of the self-test.

Enable STCU2 interrupts for those self-tests that run only on software reset domains. If the self-test includes the main reset domain, disable STCU2 interrupts.

71.1.9 AUTOLOCK_VALUE for register write access through STCU2_SKC

Register write access to STCU2 registers through the STCU2_SKC security-key mechanism times out after the number of STCU2 clock cycles as specified by the AUTOLOCK_VALUE. On this chip, AUTOLOCK_VALUE is 1_FFFFh.

71.1.10 PLL loss of lock during self test

Detection of PLL loss of lock during LBIST or MBIST causes a destructive reset in the system.

CMU0 programming is required before the self-test programming begins.

All the registers present in LBIST domains do not retain their content after self-test reset.

71.1.11 STCU2 BIST start (BSTART) register description

Attempted accesses with the undefined reserved values in the BSTART register can result in undefined behavior. The following table defines the fields of BSTART.

Diagram

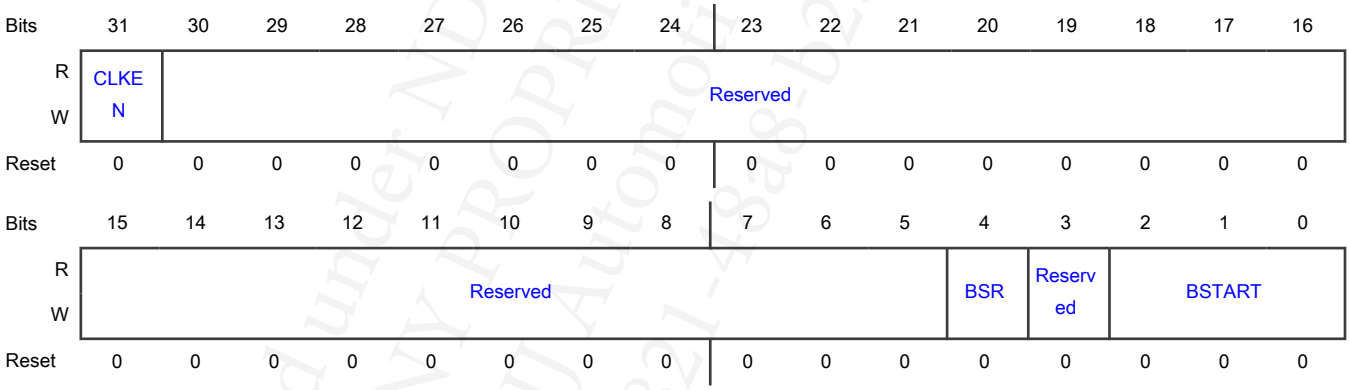


Table 559. Fields

Field	Function
31 CLKEN	Clock enable If set, it enables the clocks for the BISTs otherwise clock will remain disabled.
30-5 —	Reserved
4	BIST Status Reset

Table continues on the next page...

Table 559. Fields (continued)

Field	Function
BSR	setting this bit causes result and status information captured from MBIST run to be reset.
3 —	Reserved
2-0 BSTART	000b - NOP (reset value) 001b - Run the selected BISTs without programming them first. 010b - Reserved 011b - Reserved 100b - Program BISTs only, but do not start them. This enables different BISTs to be programmed with different algorithms before all the selected BISTs are started by setting BSTART value to 001. 101b - Program BISTs and start them. 110b - Reserved 111b - Reserved

71.1.12 STCU2 algorithm select (ALGOSEL) register description

This register selects any one of the predefined algorithms. The value used in this register is mapped to a predefined algorithm register in BIST. After MCT has programmed information on this register and starts some BIST operation, this value is transferred to the predefined algorithm register in every BIST that is selected.

If multiple predefined algorithms are selected, the order of execution is from LSB to MSB.

Any field that does not have a predefined algorithm associated with it is a reserved bit. No data is written to reserved bits and when you read these bits, they always return 0. These predefined algorithms include:

- Backgrounds
- Address modes and other register programming that may be necessary
- The number of BIST runs needed to perform the exact coverage requested.

MCT decodes this information through a series of look-up tables. The following table describes the high-level BIST algorithms that can be invoked through the MCT. The columns in that table have the following meaning:

- Index: The position in this MCT register enables this algorithm.
- Name: The symbolic name for this algorithm.
- Description: The intent and use model for this algorithm.
- BIST runs: The sequence of BIST invocations comprising this algorithm.
- Sequences: The ordered list of march elements to run. Each march element consists of a direction indicator (for increasing addresses and for decreasing addresses) and a set of march phases. A march phase consists of reading (R) or writing (W), the non-inverted (0) or inverted (1) background pattern.
- Used by default: Whether this algorithm is selected in the post-reset state of this register.

Table 560. Table 2. ALGOSEL description

Index	Name	Description	BIST runs	Sequences	Used by default
3	March C+ single	March C+ with column fast and solid background	MarchC+ column fast 1x (#4)	W0, R0W1, R1W0R0W1, R1W0, R0W1R1W0, R0	Yes
13	BasicChk	Basic self-contained checkerboard, using column fast	BasicChk (#18)	R0W1R1	_

71.1.13 Self-test programming sequence

This section contains high-level programming sequences for executing self-test in a software reset domain and the main reset domain. Detailed programming sequences for supported configurations are available in a separate application note (contact your NXP sales representative).

Software reset domain:

1. Turn off the software reset domain following the steps in the "Software reset partition turn-off flowchart" section in the Reset chapter.
2. Configure the software reset domain clocks to initial frequencies.
3. Write 1 to SELFTEST_GPR_TOP's Reset_Domain_Selftest_Enable_Register[RESET_DOMAIN_*n*_SELFTEST_ENABLE] field, where *n* corresponds to the software reset domain(s) in which self-test is to be executed.
4. Wait for SELFTEST_GPR_TOP's corresponding Reset_Domain_Selftest_Enable_Status_Register[RESET_DOMAIN_*n*_SELFTEST_ENABLE_LAST_RUN_STATUS] field to become 1.
5. Program clock sources and MC_CGMs according to the clock configuration.
6. Program self-test GPR of LBIST partitions in which self-test is to be run according to the required configuration.
7. Program STCU2 according to the required LBIST or MBIST configuration.
8. Enable LBIST or MBIST interrupt in STCU2 and trigger self-test execution by programming STCU2's RUNSW register.

At the end of the self-test run, an interrupt is generated. You can check the STCU2 status registers and take actions accordingly.

Main reset domain:

1. Turn off all software reset domains in the chip following the steps in the "Software reset partition turn-off flowchart" section in the Reset chapter.
2. Disable the modules present in the main reset domain (such as GMAC, SerDes, DDR, and TMU).
3. If any software reset domain is enabled for self-test in this run, configure the clocks of that domain to initial frequencies.
4. Write 1 to SELFTEST_GPR_TOP's Reset_Domain_Selftest_Enable_Register[RESET_DOMAIN_*n*_SELFTEST_ENABLE] field, where *n* corresponds to main reset domain. If the self-test run in any software reset domain(s) in the same configuration, write 1 to the corresponding field(s).
5. Wait for SELFTEST_GPR_TOP's Reset_Domain_Selftest_Enable_Status_Register[RESET_DOMAIN_*n*_SELFTEST_ENABLE_LAST_RUN_STATUS] field to become 1 for any main reset domains and software reset domains.
6. Program clock sources and MC_CGMs according to the clock configuration.
7. Program self-test GPR of LBIST partitions in which self-test is to be run according to the required configuration.
8. Program STCU2 according to the required LBIST or MBIST configuration.

9. Write 1 to SRC's FCCU_EOUT_OVERRIDE_CLEAR_REG[EOUT_OVERRIDE_DISABLE_DURING_SELFTEST] field if you do not want the fault state to be indicated on the EOUT pins.
10. Write 1 to MC_RGM's ERCTRL[ERASSERT] field to move the GPIO pins to a safe state and to deassert the RESET_B pin.
11. Trigger self-test execution by programming STCU2's RUNSW register.

At the end of the self-test run, a functional reset is generated. After this reset, you can check the STCU2 status registers and take actions accordingly.

SELFTEST_GPR_TOP's Reset_Domain_Selftest_Enable_Status_Register[RESET_DOMAIN_*n*_SELFTEST_ENABLE_LAST_RUN_STATUS] fields tell you whether self-test was enabled in a particular reset domain in the last self-test run.

Perform the following instructions for the configurations when the selftest is executed on the software resettable domain followed by the selftest run on the main reset domain, or software resettable domain:

- For the first MBIST run on software resettable domain, write BSTART of STCU2 at 8000_0005h, and then start selftest.
- For MBIST run on either the main reset domain or software resettable domain just after the MBIST run on the software resettable domain, write BSTART of STCU2 at 8000_0015h to clear the results and the status information from the previous MBIST run on SRD.

71.1.14 Handling unrecoverable faults

Only recoverable and non-critical faults map to FCCU. The unrecoverable faults from STCU2 do not map to FCCU. The unrecoverable and critical faults map to MC_RGM instead of FCCU. Therefore, ignore this chapter's references to mapping of these faults to FCCU.

71.2 Introduction

STCU2 is a comprehensive programmable hardware module that:

- Manages the execution of BISTs
- Indicates whether each BIST passed
- Manages the chip's LBIST and the MBIST blocks

71.3 Main features

STCU2 includes:

- System interface for reading from and writing to the STCU2 registers during runtime (online) via the CPU
- Programmable scheduler for BIST execution
- Control over LBIST concurrent or sequential execution
- Control over MBIST concurrent or sequential execution
- Programmable LBIST delayed concurrent start
- Programmable internal clock prescaler to reduce internal and BIST clocks
- Dedicated online MBIST interrupt source generation
- Dedicated online LBIST interrupt source generation
- Programmable BIST-sequence-execution watchdog timer that specifies the maximum time allowed for the execution of a BIST sequence
- Fixed register-write-access watchdog timer that specifies the maximum amount of time that STCU2 allows you to write to its registers after you unlock them
- Fields that indicate the execution status of each BIST, see Types of BIST sequences and Types of BIST partitions

- Fields that indicate the status of each type of online STCU2 internal error condition
- Field that indicates the status of an online hardware abort
- Programmable fault mapping of each BIST for controlling the type of fault that STCU2 reports (recoverable or unrecoverable) when the BIST fails to execute
- Programmable fault mapping of each STCU2 internal error condition for controlling the type of fault that STCU2 reports (recoverable or unrecoverable) when the condition occurs
- Signals to report recoverable and unrecoverable faults to the FCCU module
- Redundant recoverable and unrecoverable fault-generation logic to improve reliability
- FCCU recoverable and unrecoverable fault-injection mechanism
- Global register write-protection mechanism that requires two security key codes
- Global automatic power saving after a BIST sequence is completed when watchdog timer time-out is detected
- Watchdog automatic clock wake-up mechanism when software unlocks the STCU2 registers for write access
- Watchdog automatic power saving when the fixed register-write-access watchdog timer times out

71.4 Block diagram

This diagram shows the parts of STCU2:

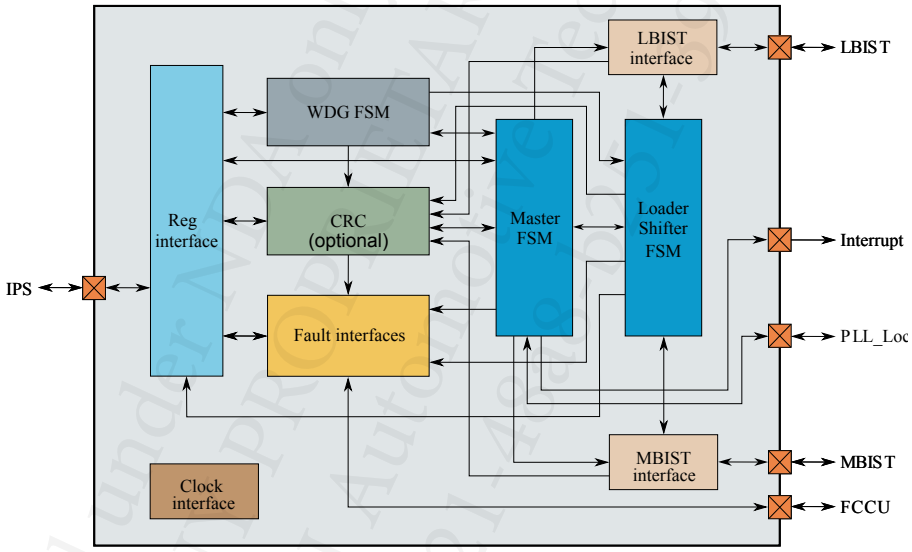


Figure 564. STCU2 block diagram

This table describes the parts of STCU2:

Table 561. STCU2 parts

Part	Function
Reg interface	Provides access to: • Registers • The security key logic • The IPS interface

Table continues on the next page...

Table 561. STCU2 parts (continued)

Part	Function
Fault interface	Performs the following tasks: • Collects the fault conditions caused by STCU2 internal error conditions and each BIST executed in a sequence • Sets the global recoverable or unrecoverable status flag, depending on the fault mapping for a given BIST • Manages the recoverable and unrecoverable fault lines to and from the FCCU module • Manages the set/clear injection mechanism provided by the FCCU module To improve the intrinsic reliability of this critical logic, the generation logic is duplicated.
Clock interface	Manages the internal and the BIST clock prescaler, the internal clock-gating power saving, and the wake-up clock feature
WDG FSM	Provides the following: • A programmable BIST-sequence-execution watchdog timer that specifies the maximum time allowed for the execution of a BIST sequence • A fixed register-write-access watchdog timer that specifies the maximum amount of time that STCU2 allows you to write to its registers after you unlock them
Master FSM	Coordinates and schedules all of the operations performed during a BIST sequence
Loader Shifter FSM	Programs the BIST registers and reads back the data to be checked at the end of each test operation
LBIST interface	Provides the interface between the chip's LBIST engines and the STCU2 controllers
MBIST interface	Provides the interface between the MBIST controller and the STCU2 controllers

71.5 Peripheral bus interface

The peripheral bus interface is a slave bus used for configuration purposes via CPU. The module supports the following bus read operations:

- Word (32 bits) data read operations to any registers
- Any other operation is not supported.

Word, low and high half-words, byte operations are supported in write mode only:

- Write access is allowed by software only when CFG[WRP] is cleared. The default value of CFG[WRP] is cleared.
- Exceptions are the following read/write fields: CFG[WRP], ERR_STAT[UFSF] and ERR_STAT[RFSF].
- In all the other conditions, the write operations are not supported and all the STCU2 registers can be only read.
- STCU2 Configuration (CFG) should only be accessed in 32 bit (partial access not allowed).

The STCU2 module generates a transfer error in the following cases:

- Any read access to the registers after writing Key1 and before writing Key2 (in online) and before register write-access watchdog timer expires.

- Any write/read access to the register addresses not mapped on the peripheral but included in the address space of the peripheral
- Any write/read operation different from byte/halfword/word (free byte enables or other operations) on each register
- Any write operation on Double Security Key register applying a wrong sequence of keys (the two write operations cannot be interleaved with other access to STCU2 registers)
- Any write operation performed on a register when the Double Security keys have not been applied
- Any write operation performed on registers when CFG[WRP] is set and the access is performed through software (internal peripheral interface). The exceptions are SKC register (for valid keys value and sequence) and INT_FLG register(interrupt flag) in which STCU2 does not assert transfer error.
- Any write operation performed on Read Only registers

The registers of the STCU2 module are accessible (read/write) in each access mode: user, supervisor, or test.

In case there are write operations on bits marked as reserved, the transfer error is not generated.

NOTE

See the chip-specific STCU2 information for the wait time necessary to write to the online registers.

71.6 BISTs and BIST partitions

71.6.1 Definition: BIST

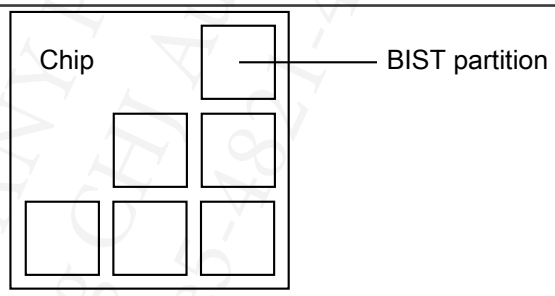
A BIST is a test that the chip can execute to verify the functional integrity of a part of itself. The chip uses STCU2 and other on-chip hardware to execute a BIST.

71.6.2 Definition: BIST partition

A BIST partition is a part of a chip for which a BIST has been defined. The hardware that is included in a given BIST partition is chip-specific. For a list of the hardware included in each of the BIST partitions on this chip, see the chip-specific STCU2 information.

71.6.3 Example: BIST partitions on a chip

This is an example of a chip with six BIST partitions:



71.6.4 Types of BIST partitions

The STCU2 module on this chip supports the following type or types of BIST partitions:

Table 562. Types of BIST partitions

BIST partition type	Description
MBIST partition	An SRAM or ROM block
LBIST partition	One or more digital modules

71.7 BIST sequences

71.7.1 Definition: BIST sequence

A BIST sequence is a programmable series of one or more phases, each of which executes one or more individual BISTs.

71.7.2 STCU2 executes MBISTs before LBISTs

If a BIST sequence includes both MBISTs and LBISTs; STCU2 executes the MBISTs first.

71.7.3 Example: Single-phase BIST sequence

This is an example of a single-phase BIST sequence in which STCU2 executes only one BIST:

Table 563. Example: Single-phase BIST sequence

Phase	BIST executed
0	MBIST 16

71.7.4 Example: Multi-phase BIST sequence

This is an example of a multiphase BIST sequence in which STCU2 executes more than one BIST in parallel in some phases:

Table 564. Example: Multi-phase BIST sequence

Phase	BISTs executed
0	MBISTs 7, 38
1	MBIST 23
2	MBISTs 6, 16, 29, 23
3	LBIST 12
4	LBISTs 13, 8, 11
5	LBISTs 19, 6

71.7.5 Types of BIST sequences

The STCU2 module on this chip supports these types of BIST sequences:

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4848-8251-39013ab624c1

Table 565. Example: Types of BIST sequences

BIST sequence type	Description
Online BIST sequence	A BIST sequence that STCU2 executes during runtime at the request of software. Application software configures and initiates execution of the BIST sequence by loading values into STCU2 registers.

71.7.6 Supported BIST sequences

NXP supports only a specific set of BIST sequences for this chip. For the list of supported BIST sequences, see the chip-specific STCU2 information.

71.8 STCU2 register descriptions

The STCU2 registers are listed in this section.

NOTE

Always write the reset value for the Reserved fields.

During online self-test, after the STCU2 registers are configured, they must not be overridden via internal peripheral system until the self-test is complete.

71.8.1 STCU2 memory map

STCU2 base address: 4002_8000h

Offset	Register	Width (In bits)	Access	Reset value
4h	STCU2 Run Software (RUNSW)	32	RW	0000_0000h
8h	STCU2 SK Code (SKC)	32	WO	0000_0000h
Ch	STCU2 Configuration (CFG)	32	RW	0000_0001h
14h	STCU2 Watchdog Granularity (WDG)	32	RW	0000_FFFFh
18h	STCU2 Interrupt Flag (INT_FLG)	32	W1C	0000_0000h
24h	STCU2 Error (ERR_STAT)	32	RW	0000_0000h
28h	STCU2 Error FM (ERR_FM)	32	RW	0000_0000h
4Ch	STCU2 Online LBIST Status (LBSSW0)	32	RO	0000_0000h
50h	STCU2 Online LBIST Status (LBSSW1)	32	RO	0000_0000h
5Ch	STCU2 Online LBIST End Flag (LBESW0)	32	RO	0000_0000h
60h	STCU2 Online LBIST End Flag (LBESW1)	32	RO	0000_0000h
7Ch	STCU2 Online LBIST Unrecoverable FM (LBUFM0)	32	RW	0000_0000h
80h	STCU2 Online LBIST Unrecoverable FM (LBUFM1)	32	RW	0000_0000h
10Ch	STCU2 Online MBIST Status (MBSSW0)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
110h	STCU2 Online MBIST Status (MBSSW1)	32	RO	0000_0000h
114h	STCU2 Online MBIST Status (MBSSW2)	32	RO	0000_0000h
118h	STCU2 Online MBIST Status (MBSSW3)	32	RO	0000_0000h
14Ch	STCU2 Online MBIST End Flag (MBESW0)	32	RO	0000_0000h
150h	STCU2 Online MBIST End Flag (MBESW1)	32	RO	0000_0000h
154h	STCU2 Online MBIST End Flag (MBESW2)	32	RO	0000_0000h
158h	STCU2 Online MBIST End Flag (MBESW3)	32	RO	0000_0000h
18Ch	STCU2 MBIST Unrecoverable FM (MBUFM0)	32	RW	0000_0000h
190h	STCU2 MBIST Unrecoverable FM (MBUFM1)	32	RW	0000_0000h
194h	STCU2 MBIST Unrecoverable FM (MBUFM2)	32	RW	0000_0000h
198h	STCU2 MBIST Unrecoverable FM (MBUFM3)	32	RW	0000_0000h
200h	STCU2 LBIST Control (LB_CTRL0)	32	RW	0000_0000h
204h	STCU2 LBIST PC Stop (LB_PCS0)	32	RW	0000_0000h
220h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW0)	32	RW	FFFF_FFFFh
224h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW0)	32	RW	FFFF_FFFFh
228h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW0)	32	RO	0000_0000h
22Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW0)	32	RO	0000_0000h
240h	STCU2 LBIST Control (LB_CTRL1)	32	RW	0000_0000h
244h	STCU2 LBIST PC Stop (LB_PCS1)	32	RW	0000_0000h
260h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW1)	32	RW	FFFF_FFFFh
264h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW1)	32	RW	FFFF_FFFFh
268h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW1)	32	RO	0000_0000h
26Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW1)	32	RO	0000_0000h
280h	STCU2 LBIST Control (LB_CTRL2)	32	RW	0000_0000h
284h	STCU2 LBIST PC Stop (LB_PCS2)	32	RW	0000_0000h
2A0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW2)	32	RW	FFFF_FFFFh
2A4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW2)	32	RW	FFFF_FFFFh
2A8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW2)	32	RO	0000_0000h
2ACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW2)	32	RO	0000_0000h
2C0h	STCU2 LBIST Control (LB_CTRL3)	32	RW	0000_0000h
2C4h	STCU2 LBIST PC Stop (LB_PCS3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2E0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW3)	32	RW	FFFF_FFFFh
2E4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW3)	32	RW	FFFF_FFFFh
2E8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW3)	32	RO	0000_0000h
2ECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW3)	32	RO	0000_0000h
300h	STCU2 LBIST Control (LB_CTRL4)	32	RW	0000_0000h
304h	STCU2 LBIST PC Stop (LB_PCS4)	32	RW	0000_0000h
320h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW4)	32	RW	FFFF_FFFFh
324h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW4)	32	RW	FFFF_FFFFh
328h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW4)	32	RO	0000_0000h
32Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW4)	32	RO	0000_0000h
340h	STCU2 LBIST Control (LB_CTRL5)	32	RW	0000_0000h
344h	STCU2 LBIST PC Stop (LB_PCS5)	32	RW	0000_0000h
360h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW5)	32	RW	FFFF_FFFFh
364h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW5)	32	RW	FFFF_FFFFh
368h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW5)	32	RO	0000_0000h
36Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW5)	32	RO	0000_0000h
380h	STCU2 LBIST Control (LB_CTRL6)	32	RW	0000_0000h
384h	STCU2 LBIST PC Stop (LB_PCS6)	32	RW	0000_0000h
3A0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW6)	32	RW	FFFF_FFFFh
3A4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW6)	32	RW	FFFF_FFFFh
3A8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW6)	32	RO	0000_0000h
3ACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW6)	32	RO	0000_0000h
3C0h	STCU2 LBIST Control (LB_CTRL7)	32	RW	0000_0000h
3C4h	STCU2 LBIST PC Stop (LB_PCS7)	32	RW	0000_0000h
3E0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW7)	32	RW	FFFF_FFFFh
3E4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW7)	32	RW	FFFF_FFFFh
3E8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW7)	32	RO	0000_0000h
3ECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW7)	32	RO	0000_0000h
400h	STCU2 LBIST Control (LB_CTRL8)	32	RW	0000_0000h
404h	STCU2 LBIST PC Stop (LB_PCS8)	32	RW	0000_0000h
420h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW8)	32	RW	FFFF_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
424h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW8)	32	RW	FFFF_FFFFh
428h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW8)	32	RO	0000_0000h
42Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW8)	32	RO	0000_0000h
440h	STCU2 LBIST Control (LB_CTRL9)	32	RW	0000_0000h
444h	STCU2 LBIST PC Stop (LB_PCS9)	32	RW	0000_0000h
460h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW9)	32	RW	FFFF_FFFFh
464h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW9)	32	RW	FFFF_FFFFh
468h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW9)	32	RO	0000_0000h
46Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW9)	32	RO	0000_0000h
480h	STCU2 LBIST Control (LB_CTRL10)	32	RW	0000_0000h
484h	STCU2 LBIST PC Stop (LB_PCS10)	32	RW	0000_0000h
4A0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW10)	32	RW	FFFF_FFFFh
4A4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW10)	32	RW	FFFF_FFFFh
4A8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW10)	32	RO	0000_0000h
4ACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW10)	32	RO	0000_0000h
4C0h	STCU2 LBIST Control (LB_CTRL11)	32	RW	0000_0000h
4C4h	STCU2 LBIST PC Stop (LB_PCS11)	32	RW	0000_0000h
4E0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW11)	32	RW	FFFF_FFFFh
4E4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW11)	32	RW	FFFF_FFFFh
4E8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW11)	32	RO	0000_0000h
4ECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW11)	32	RO	0000_0000h
500h	STCU2 LBIST Control (LB_CTRL12)	32	RW	0000_0000h
504h	STCU2 LBIST PC Stop (LB_PCS12)	32	RW	0000_0000h
520h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW12)	32	RW	FFFF_FFFFh
524h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW12)	32	RW	FFFF_FFFFh
528h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW12)	32	RO	0000_0000h
52Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW12)	32	RO	0000_0000h
540h	STCU2 LBIST Control (LB_CTRL13)	32	RW	0000_0000h
544h	STCU2 LBIST PC Stop (LB_PCS13)	32	RW	0000_0000h
560h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW13)	32	RW	FFFF_FFFFh
564h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW13)	32	RW	FFFF_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
568h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW13)	32	RO	0000_0000h
56Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW13)	32	RO	0000_0000h
580h	STCU2 LBIST Control (LB_CTRL14)	32	RW	0000_0000h
584h	STCU2 LBIST PC Stop (LB_PCS14)	32	RW	0000_0000h
5A0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW14)	32	RW	FFFF_FFFFh
5A4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW14)	32	RW	FFFF_FFFFh
5A8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW14)	32	RO	0000_0000h
5ACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW14)	32	RO	0000_0000h
5C0h	STCU2 LBIST Control (LB_CTRL15)	32	RW	0000_0000h
5C4h	STCU2 LBIST PC Stop (LB_PCS15)	32	RW	0000_0000h
5E0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW15)	32	RW	FFFF_FFFFh
5E4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW15)	32	RW	FFFF_FFFFh
5E8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW15)	32	RO	0000_0000h
5ECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW15)	32	RO	0000_0000h
600h	STCU2 LBIST Control (LB_CTRL16)	32	RW	0000_0000h
604h	STCU2 LBIST PC Stop (LB_PCS16)	32	RW	0000_0000h
620h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW16)	32	RW	FFFF_FFFFh
624h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW16)	32	RW	FFFF_FFFFh
628h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW16)	32	RO	0000_0000h
62Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW16)	32	RO	0000_0000h
640h	STCU2 LBIST Control (LB_CTRL17)	32	RW	0000_0000h
644h	STCU2 LBIST PC Stop (LB_PCS17)	32	RW	0000_0000h
660h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW17)	32	RW	FFFF_FFFFh
664h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW17)	32	RW	FFFF_FFFFh
668h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW17)	32	RO	0000_0000h
66Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW17)	32	RO	0000_0000h
680h	STCU2 LBIST Control (LB_CTRL18)	32	RW	0000_0000h
684h	STCU2 LBIST PC Stop (LB_PCS18)	32	RW	0000_0000h
6A0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW18)	32	RW	FFFF_FFFFh
6A4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW18)	32	RW	FFFF_FFFFh
6A8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW18)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
6ACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW18)	32	RO	0000_0000h
6C0h	STCU2 LBIST Control (LB_CTRL19)	32	RW	0000_0000h
6C4h	STCU2 LBIST PC Stop (LB_PCS19)	32	RW	0000_0000h
6E0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW19)	32	RW	FFFF_FFFFh
6E4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW19)	32	RW	FFFF_FFFFh
6E8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW19)	32	RO	0000_0000h
6ECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW19)	32	RO	0000_0000h
700h	STCU2 LBIST Control (LB_CTRL20)	32	RW	0000_0000h
704h	STCU2 LBIST PC Stop (LB_PCS20)	32	RW	0000_0000h
720h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW20)	32	RW	FFFF_FFFFh
724h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW20)	32	RW	FFFF_FFFFh
728h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW20)	32	RO	0000_0000h
72Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW20)	32	RO	0000_0000h
740h	STCU2 LBIST Control (LB_CTRL21)	32	RW	0000_0000h
744h	STCU2 LBIST PC Stop (LB_PCS21)	32	RW	0000_0000h
760h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW21)	32	RW	FFFF_FFFFh
764h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW21)	32	RW	FFFF_FFFFh
768h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW21)	32	RO	0000_0000h
76Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW21)	32	RO	0000_0000h
780h	STCU2 LBIST Control (LB_CTRL22)	32	RW	0000_0000h
784h	STCU2 LBIST PC Stop (LB_PCS22)	32	RW	0000_0000h
7A0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW22)	32	RW	FFFF_FFFFh
7A4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW22)	32	RW	FFFF_FFFFh
7A8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW22)	32	RO	0000_0000h
7ACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW22)	32	RO	0000_0000h
7C0h	STCU2 LBIST Control (LB_CTRL23)	32	RW	0000_0000h
7C4h	STCU2 LBIST PC Stop (LB_PCS23)	32	RW	0000_0000h
7E0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW23)	32	RW	FFFF_FFFFh
7E4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW23)	32	RW	FFFF_FFFFh
7E8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW23)	32	RO	0000_0000h
7ECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW23)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
800h	STCU2 LBIST Control (LB_CTRL24)	32	RW	0000_0000h
804h	STCU2 LBIST PC Stop (LB_PCS24)	32	RW	0000_0000h
820h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW24)	32	RW	FFFF_FFFFh
824h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW24)	32	RW	FFFF_FFFFh
828h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW24)	32	RO	0000_0000h
82Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW24)	32	RO	0000_0000h
840h	STCU2 LBIST Control (LB_CTRL25)	32	RW	0000_0000h
844h	STCU2 LBIST PC Stop (LB_PCS25)	32	RW	0000_0000h
860h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW25)	32	RW	FFFF_FFFFh
864h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW25)	32	RW	FFFF_FFFFh
868h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW25)	32	RO	0000_0000h
86Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW25)	32	RO	0000_0000h
880h	STCU2 LBIST Control (LB_CTRL26)	32	RW	0000_0000h
884h	STCU2 LBIST PC Stop (LB_PCS26)	32	RW	0000_0000h
8A0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW26)	32	RW	FFFF_FFFFh
8A4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW26)	32	RW	FFFF_FFFFh
8A8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW26)	32	RO	0000_0000h
8ACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW26)	32	RO	0000_0000h
8C0h	STCU2 LBIST Control (LB_CTRL27)	32	RW	0000_0000h
8C4h	STCU2 LBIST PC Stop (LB_PCS27)	32	RW	0000_0000h
8E0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW27)	32	RW	FFFF_FFFFh
8E4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW27)	32	RW	FFFF_FFFFh
8E8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW27)	32	RO	0000_0000h
8ECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW27)	32	RO	0000_0000h
900h	STCU2 LBIST Control (LB_CTRL28)	32	RW	0000_0000h
904h	STCU2 LBIST PC Stop (LB_PCS28)	32	RW	0000_0000h
920h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW28)	32	RW	FFFF_FFFFh
924h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW28)	32	RW	FFFF_FFFFh
928h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW28)	32	RO	0000_0000h
92Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW28)	32	RO	0000_0000h
940h	STCU2 LBIST Control (LB_CTRL29)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
944h	STCU2 LBIST PC Stop (LB_PCS29)	32	RW	0000_0000h
960h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW29)	32	RW	FFFF_FFFFh
964h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW29)	32	RW	FFFF_FFFFh
968h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW29)	32	RO	0000_0000h
96Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW29)	32	RO	0000_0000h
980h	STCU2 LBIST Control (LB_CTRL30)	32	RW	0000_0000h
984h	STCU2 LBIST PC Stop (LB_PCS30)	32	RW	0000_0000h
9A0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW30)	32	RW	FFFF_FFFFh
9A4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW30)	32	RW	FFFF_FFFFh
9A8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW30)	32	RO	0000_0000h
9ACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW30)	32	RO	0000_0000h
9C0h	STCU2 LBIST Control (LB_CTRL31)	32	RW	0000_0000h
9C4h	STCU2 LBIST PC Stop (LB_PCS31)	32	RW	0000_0000h
9E0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW31)	32	RW	FFFF_FFFFh
9E4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW31)	32	RW	FFFF_FFFFh
9E8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW31)	32	RO	0000_0000h
9ECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW31)	32	RO	0000_0000h
A00h	STCU2 LBIST Control (LB_CTRL32)	32	RW	0000_0000h
A04h	STCU2 LBIST PC Stop (LB_PCS32)	32	RW	0000_0000h
A20h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW32)	32	RW	FFFF_FFFFh
A24h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW32)	32	RW	FFFF_FFFFh
A28h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW32)	32	RO	0000_0000h
A2Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW32)	32	RO	0000_0000h
A40h	STCU2 LBIST Control (LB_CTRL33)	32	RW	0000_0000h
A44h	STCU2 LBIST PC Stop (LB_PCS33)	32	RW	0000_0000h
A60h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW33)	32	RW	FFFF_FFFFh
A64h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW33)	32	RW	FFFF_FFFFh
A68h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW33)	32	RO	0000_0000h
A6Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW33)	32	RO	0000_0000h
A80h	STCU2 LBIST Control (LB_CTRL34)	32	RW	0000_0000h
A84h	STCU2 LBIST PC Stop (LB_PCS34)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
AA0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW34)	32	RW	FFFF_FFFFh
AA4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW34)	32	RW	FFFF_FFFFh
AA8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW34)	32	RO	0000_0000h
AACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW34)	32	RO	0000_0000h
AC0h	STCU2 LBIST Control (LB_CTRL35)	32	RW	0000_0000h
AC4h	STCU2 LBIST PC Stop (LB_PCS35)	32	RW	0000_0000h
AE0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW35)	32	RW	FFFF_FFFFh
AE4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW35)	32	RW	FFFF_FFFFh
AE8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW35)	32	RO	0000_0000h
AECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW35)	32	RO	0000_0000h
B00h	STCU2 LBIST Control (LB_CTRL36)	32	RW	0000_0000h
B04h	STCU2 LBIST PC Stop (LB_PCS36)	32	RW	0000_0000h
B20h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW36)	32	RW	FFFF_FFFFh
B24h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW36)	32	RW	FFFF_FFFFh
B28h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW36)	32	RO	0000_0000h
B2Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW36)	32	RO	0000_0000h
B40h	STCU2 LBIST Control (LB_CTRL37)	32	RW	0000_0000h
B44h	STCU2 LBIST PC Stop (LB_PCS37)	32	RW	0000_0000h
B60h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW37)	32	RW	FFFF_FFFFh
B64h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW37)	32	RW	FFFF_FFFFh
B68h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW37)	32	RO	0000_0000h
B6Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW37)	32	RO	0000_0000h
B80h	STCU2 LBIST Control (LB_CTRL38)	32	RW	0000_0000h
B84h	STCU2 LBIST PC Stop (LB_PCS38)	32	RW	0000_0000h
BA0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW38)	32	RW	FFFF_FFFFh
BA4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW38)	32	RW	FFFF_FFFFh
BA8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW38)	32	RO	0000_0000h
BACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW38)	32	RO	0000_0000h
BC0h	STCU2 LBIST Control (LB_CTRL39)	32	RW	0000_0000h
BC4h	STCU2 LBIST PC Stop (LB_PCS39)	32	RW	0000_0000h
BE0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW39)	32	RW	FFFF_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
BE4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW39)	32	RW	FFFF_FFFFh
BE8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW39)	32	RO	0000_0000h
BECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW39)	32	RO	0000_0000h
C00h	STCU2 LBIST Control (LB_CTRL40)	32	RW	0000_0000h
C04h	STCU2 LBIST PC Stop (LB_PCS40)	32	RW	0000_0000h
C20h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW40)	32	RW	FFFF_FFFFh
C24h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW40)	32	RW	FFFF_FFFFh
C28h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW40)	32	RO	0000_0000h
C2Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW40)	32	RO	0000_0000h
C40h	STCU2 LBIST Control (LB_CTRL41)	32	RW	0000_0000h
C44h	STCU2 LBIST PC Stop (LB_PCS41)	32	RW	0000_0000h
C60h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW41)	32	RW	FFFF_FFFFh
C64h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW41)	32	RW	FFFF_FFFFh
C68h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW41)	32	RO	0000_0000h
C6Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW41)	32	RO	0000_0000h
C80h	STCU2 LBIST Control (LB_CTRL42)	32	RW	0000_0000h
C84h	STCU2 LBIST PC Stop (LB_PCS42)	32	RW	0000_0000h
CA0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW42)	32	RW	FFFF_FFFFh
CA4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW42)	32	RW	FFFF_FFFFh
CA8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW42)	32	RO	0000_0000h
CACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW42)	32	RO	0000_0000h
CC0h	STCU2 LBIST Control (LB_CTRL43)	32	RW	0000_0000h
CC4h	STCU2 LBIST PC Stop (LB_PCS43)	32	RW	0000_0000h
CE0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW43)	32	RW	FFFF_FFFFh
CE4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW43)	32	RW	FFFF_FFFFh
CE8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW43)	32	RO	0000_0000h
CECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW43)	32	RO	0000_0000h
D00h	STCU2 LBIST Control (LB_CTRL44)	32	RW	0000_0000h
D04h	STCU2 LBIST PC Stop (LB_PCS44)	32	RW	0000_0000h
D20h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW44)	32	RW	FFFF_FFFFh
D24h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW44)	32	RW	FFFF_FFFFh

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
D28h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW44)	32	RO	0000_0000h
D2Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW44)	32	RO	0000_0000h
D40h	STCU2 LBIST Control (LB_CTRL45)	32	RW	0000_0000h
D44h	STCU2 LBIST PC Stop (LB_PCS45)	32	RW	0000_0000h
D60h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW45)	32	RW	FFFF_FFFFh
D64h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW45)	32	RW	FFFF_FFFFh
D68h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW45)	32	RO	0000_0000h
D6Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW45)	32	RO	0000_0000h
D80h	STCU2 LBIST Control (LB_CTRL46)	32	RW	0000_0000h
D84h	STCU2 LBIST PC Stop (LB_PCS46)	32	RW	0000_0000h
DA0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW46)	32	RW	FFFF_FFFFh
DA4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW46)	32	RW	FFFF_FFFFh
DA8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW46)	32	RO	0000_0000h
DACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW46)	32	RO	0000_0000h
DC0h	STCU2 LBIST Control (LB_CTRL47)	32	RW	0000_0000h
DC4h	STCU2 LBIST PC Stop (LB_PCS47)	32	RW	0000_0000h
DE0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW47)	32	RW	FFFF_FFFFh
DE4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW47)	32	RW	FFFF_FFFFh
DE8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW47)	32	RO	0000_0000h
DECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW47)	32	RO	0000_0000h
E00h	STCU2 LBIST Control (LB_CTRL48)	32	RW	0000_0000h
E04h	STCU2 LBIST PC Stop (LB_PCS48)	32	RW	0000_0000h
E20h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW48)	32	RW	FFFF_FFFFh
E24h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW48)	32	RW	FFFF_FFFFh
E28h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW48)	32	RO	0000_0000h
E2Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW48)	32	RO	0000_0000h
E40h	STCU2 LBIST Control (LB_CTRL49)	32	RW	0000_0000h
E44h	STCU2 LBIST PC Stop (LB_PCS49)	32	RW	0000_0000h
E60h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW49)	32	RW	FFFF_FFFFh
E64h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW49)	32	RW	FFFF_FFFFh
E68h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW49)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
E6Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW49)	32	RO	0000_0000h
E80h	STCU2 LBIST Control (LB_CTRL50)	32	RW	0000_0000h
E84h	STCU2 LBIST PC Stop (LB_PCS50)	32	RW	0000_0000h
EA0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW50)	32	RW	FFFF_FFFFh
EA4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW50)	32	RW	FFFF_FFFFh
EA8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW50)	32	RO	0000_0000h
EACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW50)	32	RO	0000_0000h
EC0h	STCU2 LBIST Control (LB_CTRL51)	32	RW	0000_0000h
EC4h	STCU2 LBIST PC Stop (LB_PCS51)	32	RW	0000_0000h
EE0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW51)	32	RW	FFFF_FFFFh
EE4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW51)	32	RW	FFFF_FFFFh
EE8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW51)	32	RO	0000_0000h
EECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW51)	32	RO	0000_0000h
F00h	STCU2 LBIST Control (LB_CTRL52)	32	RW	0000_0000h
F04h	STCU2 LBIST PC Stop (LB_PCS52)	32	RW	0000_0000h
F20h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW52)	32	RW	FFFF_FFFFh
F24h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW52)	32	RW	FFFF_FFFFh
F28h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW52)	32	RO	0000_0000h
F2Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW52)	32	RO	0000_0000h
F40h	STCU2 LBIST Control (LB_CTRL53)	32	RW	0000_0000h
F44h	STCU2 LBIST PC Stop (LB_PCS53)	32	RW	0000_0000h
F60h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW53)	32	RW	FFFF_FFFFh
F64h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW53)	32	RW	FFFF_FFFFh
F68h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW53)	32	RO	0000_0000h
F6Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW53)	32	RO	0000_0000h
F80h	STCU2 LBIST Control (LB_CTRL54)	32	RW	0000_0000h
F84h	STCU2 LBIST PC Stop (LB_PCS54)	32	RW	0000_0000h
FA0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW54)	32	RW	FFFF_FFFFh
FA4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW54)	32	RW	FFFF_FFFFh
FA8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW54)	32	RO	0000_0000h
FACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW54)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
FC0h	STCU2 LBIST Control (LB_CTRL55)	32	RW	0000_0000h
FC4h	STCU2 LBIST PC Stop (LB_PCS55)	32	RW	0000_0000h
FE0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW55)	32	RW	FFFF_FFFFh
FE4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW55)	32	RW	FFFF_FFFFh
FE8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW55)	32	RO	0000_0000h
FECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW55)	32	RO	0000_0000h
1000h	STCU2 LBIST Control (LB_CTRL56)	32	RW	0000_0000h
1004h	STCU2 LBIST PC Stop (LB_PCS56)	32	RW	0000_0000h
1020h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW56)	32	RW	FFFF_FFFFh
1024h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW56)	32	RW	FFFF_FFFFh
1028h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW56)	32	RO	0000_0000h
102Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW56)	32	RO	0000_0000h
1040h	STCU2 LBIST Control (LB_CTRL57)	32	RW	0000_0000h
1044h	STCU2 LBIST PC Stop (LB_PCS57)	32	RW	0000_0000h
1060h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW57)	32	RW	FFFF_FFFFh
1064h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW57)	32	RW	FFFF_FFFFh
1068h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW57)	32	RO	0000_0000h
106Ch	STCU2 Online LBIST MISR Read High (LB_MISRRHSW57)	32	RO	0000_0000h
1080h	STCU2 LBIST Control (LB_CTRL58)	32	RW	0000_0000h
1084h	STCU2 LBIST PC Stop (LB_PCS58)	32	RW	0000_0000h
10A0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW58)	32	RW	FFFF_FFFFh
10A4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW58)	32	RW	FFFF_FFFFh
10A8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW58)	32	RO	0000_0000h
10ACh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW58)	32	RO	0000_0000h
10C0h	STCU2 LBIST Control (LB_CTRL59)	32	RW	0000_0000h
10C4h	STCU2 LBIST PC Stop (LB_PCS59)	32	RW	0000_0000h
10E0h	STCU2 Online LBIST MISR Expected Low (LB_MISRELSW59)	32	RW	FFFF_FFFFh
10E4h	STCU2 Online LBIST MISR Expected High (LB_MISREHSW59)	32	RW	FFFF_FFFFh
10E8h	STCU2 Online LBIST MISR Read Low (LB_MISRRLSW59)	32	RO	0000_0000h
10ECh	STCU2 Online LBIST MISR Read High (LB_MISRRHSW59)	32	RO	0000_0000h
2200h	STCU2 Algorithm Select (ALGOSEL)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
220Ch	STCU2 MBIST Stagger (STGGR)	32	RW	0000_0000h
2210h	STCU2 BIST Start (BSTART)	32	RW	0000_0000h
2214h - 23C8h	STCU2 MBIST Control (MB_CTRL0 - MB_CTRL109)	32	RW	0000_0000h

71.8.2 STCU2 Run Software (RUNSW)

Offset

Register	Offset
RUNSW	4h

Function

The RUNSW register defines the RUN bit to start the online self-testing procedure.

The R/W fields in this register are readable at any time. However, you can write to these fields only when CFG[WRP] = 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				MBIE	LBIE	0	0	0						0	RUNSW
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-12 —	Reserved
11 MBIE	MBIST Interrupt Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Interrupt is not generated at the end of the software MBIST execution phase for the selected memories. 1b - At the end of the software MBIST execution phase for sequential or concurrently selected memories, STCU2 forces an interrupt request to notify that MBISTs of these memories are over.
10 LBIE	LBIST Interrupt Enable 0b - Interrupt is not generated at the end of the software LBIST execution phase for the selected LBISTs. 1b - At the end of the software LBIST execution phase for sequential or concurrently selected LBIST runs, STCU2 forces an interrupt request to notify that LBISTs of the selected partitions are over.
9 —	Reserved
8 —	Reserved
7-2 —	Reserved
1 —	Reserved
0 RUNSW	The RUNSW bit is automatically cleared by STCU2 when the online self-testing procedure is complete. 0b - Idle 1b - Online self-testing procedure is running

71.8.3 STCU2 SK Code (SKC)

Offset

Register	Offset
SKC	8h

Function

The SKC register implements the security key code mechanism needed to access in write mode to the other STCU2 registers.

To unlock STCU2 access after the STCU2 asynchronous reset and at the end of the STCU2 run, the software (IPS bus) need to apply the following sequence:

- write the key1 into the SKC register
- write the key2 into the SKC register

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e54-821-48a8-b251-390d3ab924c1

Depending on the online test step, the two keys are different. Byte write operation is not allowed because the full key has to be recognized as one unit.

In case of invalid access or sequence (Key1/2 have to be applied consecutively), a transfer error on the IPS is asserted. The STCU2 write access is locked and to unlock the access, the sequence has to be applied again.

In case the STCU2 register access lasts more cycles than the one defined in the hard-coded WDG timeout, the STCU2 write access is locked and the WDG and register interface clocks are switched off. Also, in this case, to enable the write access to the STCU2 again and the WDG and register interface clocks, it is required to apply the sequence again. This hard-coded WDG counter starts decrementing its value right after POR is deasserted.

In case it is required to extend the STCU2 register access cycles before the hard-coded WDG timeout expires, Key1 and Key2 sequences need to be applied. The effect of this write operation is to re-initialize the WDG timeout counter. The STCU2 write access is locked and to unlock the access, the sequence has to be applied again.

The SKC register is not readable. The value 00000000h is always returned in case of read operation.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	SKC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	SKC															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SKC	STCU2 SK Code STCU2 security key code for online test = 753F924Eh: Key1 to unlock the write access the STCU2 = 8AC06DB1h: Key2 to unlock the write access the STCU2

71.8.4 STCU2 Configuration (CFG)

Offset

Register	Offset
CFG	Ch

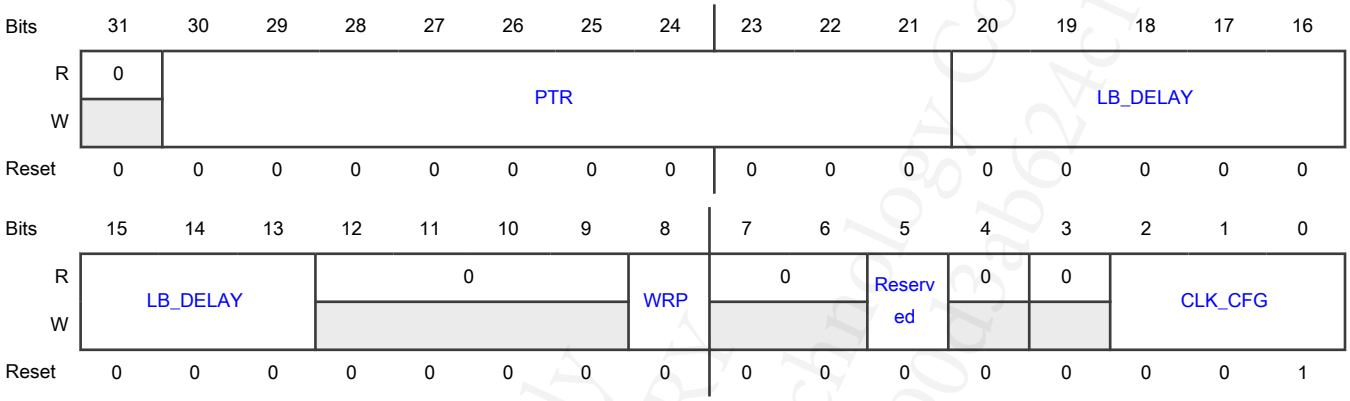
Function

The CFG register includes the global configuration of the STCU2 and can be updated in the online test steps.

The access to this register is described in the following figure. It further depends on the state of the WRP bit as follows:

- When WRP = 0: The register can be written during the online self-test case without restrictions.
- When WRP = 1:
  - The only bit that can be written without any restriction is WRP.
  - In case there are software operations that write all the register's bits, a transfer error is raised only if the value of the selected byte written differs from the current status of the register. This functionality has been implemented to prevent potential compilation behavior that might invalidate this single bit clean capability.

Diagram



Fields

Field	Function
31 —	Reserved
30-21 PTR	First LBIST or MBIST pointer PTR defines the logical pointer to the first LBIST or MBIST to be scheduled when the self-testing procedure is enabled. PTR is the entry pointer to a linked list of BIST descriptors. If PTR = 0 then the LBIST0 is initially scheduled. See BIST scheduling for details. **NOTE** You can write to this field only when WRP = 0. 0h to (3Ch - 1): pointer to LBIST 00000080h to (00000080h + 06Eh - 1): pointer to MBIST 3FFh: pointer to NIL. No BIST execution. others: invalid pointer => an error is set into the STCU2 Error (ERR_STAT).
20-13 LB_DELAY	Delay LBIST run LB_DELAY defines the delay between the LBIST starts when more than a single LBIST is selected to be executed concurrently with the purpose of smoothing the power consumption transient. The allowed delay time are these: 00h: No delay 01h: 1 x 16 STCU2 CORE_CLK cycles

Table continues on the next page...

Table continued from the previous page...

Field	Function
	02h: 2 x 16 STCU2 CORE_CLK cycles 03h: 3 x 16 STCU2 CORE_CLK cycles ... FDh: 253 x 16 STCU2 CORE_CLK cycles FEh: 254 x 16 STCU2 CORE_CLK cycles FFh: 255 x 16 STCU2 CORE_CLK cycles **NOTE** You can write to this field only when WRP = 0.
12-9 —	Reserved
8 WRP	Write Protection 0: Specific STCU2 registers can be written through IPS bus interface 1: STCU2 registers cannot be written through IPS, preventing any user application write operation
7-6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2-0 CLK_CFG	Logic, Memory BIST, and STCU2 CORE_CLK configuration CLK_CFG defines the ratio between the sys_clk and the internal clock used to program both the LBIST and the MBIST and the STCU2 CORE_CLK. The punch-out mechanism is used to generate the derived clocks. The following configurations are allowed: **NOTE** You can write to this field only when WRP = 0. 000b - sys_clk/1 001b - sys_clk/2 010b - sys_clk/3 011b - sys_clk/4

Table continues on the next page...

Table continued from the previous page...

Field	Function
	100b - sys_clk/5 101b - sys_clk/6 110b - sys_clk/7 111b - sys_clk/8

71.8.5 STCU2 Watchdog Granularity (WDG)

Offset

Register	Offset
WDG	14h

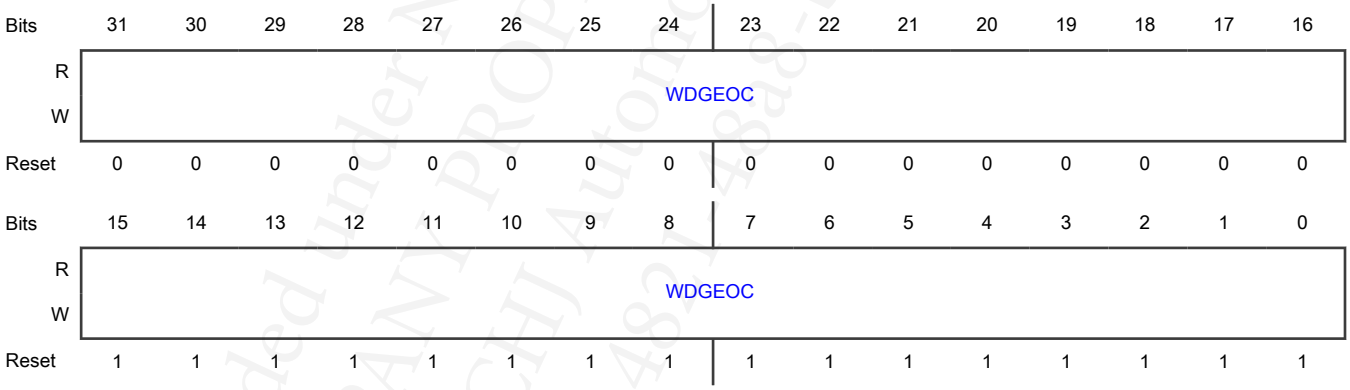
Function

The WDG register defines the time budget of LBIST and MBIST execution providing a protection mechanism in case of dead-lock or endless conditions during the self-test procedure.

In case online self-test sequence is run, it defines the timeout of the execution run.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram



Fields

Field	Function
31-0 WDGEOC	Watchdog End of Count Timer This value has to be set to define the time budget related to the online self-test execution and check that everything is correctly working within this slot of time. The delay time slots that are allowed are as follows: 0000 0000h: 1 x 16 STCU2 CORE_CLK cycles

Table continues on the next page...

Field	Function
	0000 0001h: 2 x 16 STCU2 CORE_CLK cycles 0000 0002h: 3 x 16 STCU2 CORE_CLK cycles ... FFFF FFFDh: 4294967294 x 16 STCU2 CORE_CLK cycles FFFF FFFEh: 4294967295 x 16 STCU2 CORE_CLK cycles FFFF FFFFh: 4294967296 x 16 STCU2 CORE_CLK cycles

71.8.6 STCU2 Interrupt Flag (INT_FLG)

Offset

Register	Offset
INT_FLG	18h

Function

The INT_FLG register includes the MBISTand LBIST interrupt pending bits. These bits are effective during the online self-test phase and are managed only when the corresponding control bit in the RUNSW register (MBIE and LBIE) is enabled.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														MBIFLG	LBIFLG
W															W1C	W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-2 —	Reserved
1 MBIFLG	MBIST Interrupt Flag

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No interrupt is pending 1b - An interrupt, highlighting that the online execution of the concurrent scheduled MBIST is completed, is pending. Write 1 to clear the bit.
0 LBIFLG	LBIST Interrupt Flag 0b - No interrupt is pending. 1b - An interrupt, highlighting that the online execution of the concurrent scheduled LBIST is completed, is pending. Write 1 to clear the bit.

71.8.7 STCU2 Error (ERR_STAT)

Offset

Register	Offset
ERR_STAT	24h

Function

The ERR_STAT register includes the status flags related to the STCU2 internal error conditions occurred during the configuration or the online self-testing execution.

The UFSF and RFSF can be set/cleared using the FCCU dedicated channels.

The access to this register is described in the following figure and as follows:

- If you select the byte write capability to write only UFSF and RFSF, then there is no restriction in writing these bits.
- If your software performs the write operations on other bits besides UFSF/RFSF, then a transfer error is generated only if the value you are writing to the other bits differs from their value currently stored in the register. This functionality has been implemented to prevent potential compilation behavior that might invalidate the UFSF/RFSF single bit set/clean capability.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						ABORTHW	0	0			Reserved	WDTOSW	0	ENGESW	INVPSW
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						UFSF	RFSF	0			0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8462e5a-6f1-4e3-8251-390d3ab624c1

Fields

Field	Function
31-26 —	Reserved
25 ABORTHW	Online hardware abort flag You can always read this field. The content of this field is initialized to its reset value when RUNSW[RUNSW] is set to 1. 0b - No hardware abort was requested during the online self-test sequence. 1b - A hardware abort was detected during the online self-test sequence.
24 —	Reserved
23-21 —	Reserved
20 —	Reserved
19 WDTOSW	Online watchdog timeout You can always read this field. The content of this field is initialized to its reset value when RUNSW[RUNSW] is set to 1. 0b - LBIST and MBIST time slots completed within the assigned watchdog time. 1b - LBIST and MBIST time slots not completed within the assigned watchdog time or there are internal mismatches among End of Execution signals.
18 —	Reserved
17 ENGESW	Online engine error You can always read this field. The content of this field is initialized to its reset value when RUNSW[RUNSW] is set to 1. 0b - Valid engine execution 1b - Invalid engine execution. The error conditions that set this bit are FSM, protocol error, and so on.
16 INVPSW	Online invalid pointer You can always read this field. The content of this field is initialized to its reset value when RUNSW[RUNSW] is set to 1. 0b - Valid linked pointer list

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Invalid linked pointer list. The following conditions set this bit: Initial LBIST orMBIST pointer is out of range; LBIST is selected when MBIST is concurrently running or vice versa; Error in the LBIST/MBIST linking (execution generates an infinite loop).
15-10 —	Reserved
9 UFSF	Unrecoverable Faults Status Flag This flag reports the global status of the Unrecoverable Faults(UF). This field can be set or cleared using the FCCU dedicated channel, and can also be set or cleared by software. 0b - No errors that trigger the UF condition. 1b - There are errors that trigger the UF condition.
8 RFSF	Recoverable Faults Status Flag This flag reports the global status of the Recoverable Fault (RF). This field can be set or cleared using the FCCU dedicated channel, and can also be set or cleared by software. 0b - No errors that trigger the Recoverable Faults condition 1b - There are errors that trigger the Recoverable Faults condition
7-5 —	Reserved
4-0 —	Reserved Write may occur on these fields but writes have no effect.

71.8.8 STCU2 Error FM (ERR_FM)

Offset

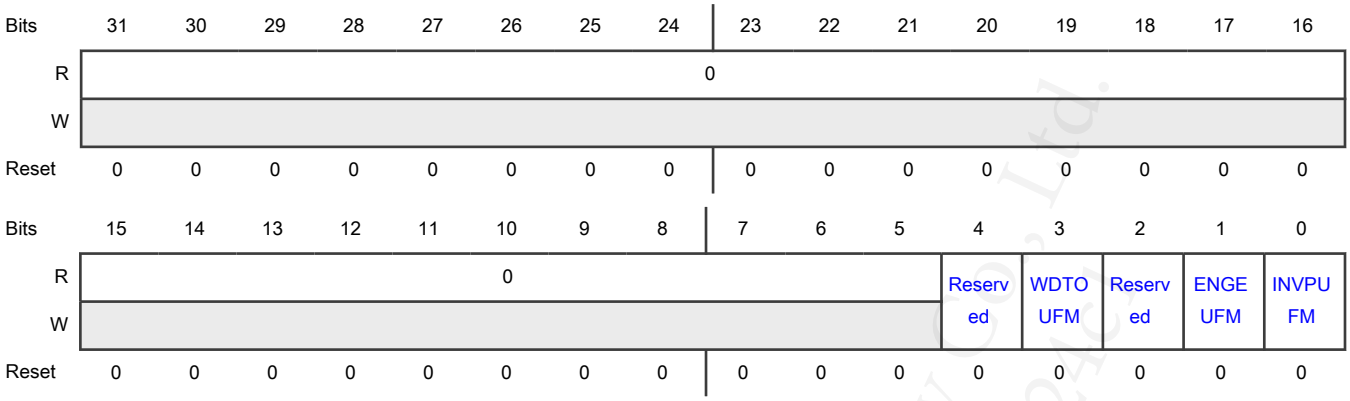
Register	Offset
ERR_FM	28h

Function

The ERR_FM register defines the fault mapping of the STCU2 faults described in the register ERR_STAT in terms of UF or RF. All sources of internal faults can be routed to UF and RF.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 —	Reserved
3 WDTOUFM	Watchdog Timeout Unrecoverable Fault Mapping 0b - Recoverable Fault Mapping 1b - Unrecoverable Fault Mapping
2 —	Reserved
1 ENGEUFM	Engine Error Unrecoverable Fault Mapping 0b - Recoverable Fault Mapping 1b - Unrecoverable Fault Mapping
0 INVPUFM	Invalid Pointer Unrecoverable Fault Mapping 0b - Recoverable Fault Mapping 1b - Unrecoverable Mapping

71.8.9 STCU2 Online LBIST Status (LBSSW0)

Offset

Register	Offset
LBSSW0	4Ch

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHTC Automotive Technology Co., Ltd.
c8c46255-4821-48a8-b251-390d3ab62452

Function

This register includes the results corresponding to the execution of the selected online LBIST.

The size of the register depends on the number of LBIST.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LBSSW31	LBSSW30	LBSSW29	LBSSW28	LBSSW27	LBSSW26	LBSSW25	LBSSW24	LBSSW23	LBSSW22	LBSSW21	LBSSW20	LBSSW19	LBSSW18	LBSSW17	LBSSW16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LBSSW15	LBSSW14	LBSSW13	LBSSW12	LBSSW11	LBSSW10	LBSSW9	LBSSW8	LBSSW7	LBSSW6	LBSSW5	LBSSW4	LBSSW3	LBSSW2	LBSSW1	LBSSW0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 LBSSWn	online status of the selected LBIST 0b - Failed LBIST execution 1b - Successful LBIST execution

71.8.10 STCU2 Online LBIST Status (LBSSW1)

Offset

Register	Offset
LBSSW1	50h

Function

This register includes the results corresponding to the execution of the selected online LBIST.

The size of the register depends on the number of LBIST .

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46203-4821-48a8-b251-39063e6248

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				LBSSW59	LBSSW58	LBSSW57	LBSSW56	LBSSW55	LBSSW54	LBSSW53	LBSSW52	LBSSW51	LBSSW50	LBSSW49	LBSSW48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LBSSW47	LBSSW46	LBSSW45	LBSSW44	LBSSW43	LBSSW42	LBSSW41	LBSSW40	LBSSW39	LBSSW38	LBSSW37	LBSSW36	LBSSW35	LBSSW34	LBSSW33	LBSSW32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-0 LBSSWn	LBSSWn online status of the selected LBIST 0b - Failed LBIST execution 1b - Successful LBIST execution

71.8.11 STCU2 Online LBIST End Flag (LBESW0)

Offset

Register	Offset
LBESW0	5Ch

Function

This register includes the End Flag related to the execution of the selected online LBIST.

The size of the register depends on the number of LBIST.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Provided under NDA only
CONFIDENTIAL PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-8214-48a8-b251-390d3ab624

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LBESW31	LBESW30	LBESW29	LBESW28	LBESW27	LBESW26	LBESW25	LBESW24	LBESW23	LBESW22	LBESW21	LBESW20	LBESW19	LBESW18	LBESW17	LBESW16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LBESW15	LBESW14	LBESW13	LBESW12	LBESW11	LBESW10	LBESW9	LBESW8	LBESW7	LBESW6	LBESW5	LBESW4	LBESW3	LBESW2	LBESW1	LBESW0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 LBESWn	LBESWx: online LBIST end status 0b - LBIST execution not yet completed 1b - LBIST execution finished

71.8.12 STCU2 Online LBIST End Flag (LBESW1)

Offset

Register	Offset
LBESW1	60h

Function

This register includes the results corresponding to the execution of the selected online LBIST.

The size of the register depends on the number of LBIST .

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				LBESW59	LBESW58	LBESW57	LBESW56	LBESW55	LBESW54	LBESW53	LBESW52	LBESW51	LBESW50	LBESW49	LBESW48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LBESW47	LBESW46	LBESW45	LBESW44	LBESW43	LBESW42	LBESW41	LBESW40	LBESW39	LBESW38	LBESW37	LBESW36	LBESW35	LBESW34	LBESW33	LBESW32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-28 —	Reserved
27-0 LBESWn	LBESW LBESWx: online LBIST end status 0b - LBIST execution not yet completed 1b - LBIST execution finished

71.8.13 STCU2 Online LBIST Unrecoverable FM (LBUFM0)

Offset

Register	Offset
LBUFM0	7Ch

Function

This register defines the fault mapping of each LBIST in terms of UF or RF.

The size of the register 1 depends on the number of LBIST.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Protected under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625e4821148a8b251-390d3ab624

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LBUFM31	LBUFM30	LBUFM29	LBUFM28	LBUFM27	LBUFM26	LBUFM25	LBUFM24	LBUFM23	LBUFM22	LBUFM21	LBUFM20	LBUFM19	LBUFM18	LBUFM17	LBUFM16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LBUFM15	LBUFM14	LBUFM13	LBUFM12	LBUFM11	LBUFM10	LBUFM9	LBUFM8	LBUFM7	LBUFM6	LBUFM5	LBUFM4	LBUFM3	LBUFM2	LBUFM1	LBUFM0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 LBUFMn	LBIST Unrecoverable Fault Mapping 0b - Recoverable Fault mapping 1b - Unrecoverable Fault mapping

71.8.14 STCU2 Online LBIST Unrecoverable FM (LBUFM1)

Offset

Register	Offset
LBUFM1	80h

Function

This register includes the results corresponding to the execution of the selected online LBIST.

The size of the register 1 depends on the number of LBIST .

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				LBUFM59	LBUFM58	LBUFM57	LBUFM56	LBUFM55	LBUFM54	LBUFM53	LBUFM52	LBUFM51	LBUFM50	LBUFM49	LBUFM48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LBUFM47	LBUFM46	LBUFM45	LBUFM44	LBUFM43	LBUFM42	LBUFM41	LBUFM40	LBUFM39	LBUFM38	LBUFM37	LBUFM36	LBUFM35	LBUFM34	LBUFM33	LBUFM32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing HJ Automotive Technology Co., Ltd.
c8c62e1821-48a8-b251-390d3ab624

Fields

Field	Function
31-28 —	Reserved
27-0 LBUFMn	LBIST Unrecoverable Fault Mapping 0b - Recoverable Fault mapping 1b - Unrecoverable Fault mapping

71.8.15 STCU2 Online MBIST Status (MBSSW0)

Offset

Register	Offset
MBSSW0	10Ch

Function

This register includes the results corresponding to the execution of the selected online MBIST in the range 31:0.

The size of the register depends on the number of BISTed RAMs/ROMs.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MBSSW31	MBSSW30	MBSSW29	MBSSW28	MBSSW27	MBSSW26	MBSSW25	MBSSW24	MBSSW23	MBSSW22	MBSSW21	MBSSW20	MBSSW19	MBSSW18	MBSSW17	MBSSW16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MBSSW15	MBSSW14	MBSSW13	MBSSW12	MBSSW11	MBSSW10	MBSSW9	MBSSW8	MBSSW7	MBSSW6	MBSSW5	MBSSW4	MBSSW3	MBSSW2	MBSSW1	MBSSW0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MBSSW31	MBSSW31 Online status of MBISTn (where n = 62:31).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
30 MBSSW30	MBSSW30 Online status of MBISTn (where n = 61:30). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
29 MBSSW29	MBSSW29 Online status of MBISTn (where n = 60:29). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
28 MBSSW28	MBSSW28 Online status of MBISTn (where n = 59:28). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
27 MBSSW27	MBSSW27 Online status of MBISTn (where n = 58:27). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
26 MBSSW26	MBSSW26 Online status of MBISTn (where n = 57:26). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
25 MBSSW25	MBSSW25 Online status of MBISTn (where n = 56:25). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
24 MBSSW24	MBSSW24 Online status of MBISTn (where n = 55:24). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
23	MBSSW23

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBSSW23	Online status of MBISTn (where n = 54:23). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
22 MBSSW22	MBSSW22 Online status of MBISTn (where n = 53:22). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
21 MBSSW21	MBSSW21 Online status of MBISTn (where n = 52:21). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
20 MBSSW20	MBSSW20 Online status of MBISTn (where n = 51:20). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
19 MBSSW19	MBSSW19 Online status of MBISTn (where n = 50:19). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
18 MBSSW18	MBSSW18 Online status of MBISTn (where n = 49:18). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
17 MBSSW17	MBSSW17 Online status of MBISTn (where n = 48:17). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
16 MBSSW16	MBSSW16 Online status of MBISTn (where n = 47:16). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 MBSSW15	MBSSW15 Online status of MBISTn (where n = 46:15). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
14 MBSSW14	MBSSW14 Online status of MBISTn (where n = 45:14). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
13 MBSSW13	MBSSW13 Online status of MBISTn (where n = 44:13). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
12 MBSSW12	MBSSW12 Online status of MBISTn (where n = 43:12). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
11 MBSSW11	MBSSW11 Online status of MBISTn (where n = 42:11). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
10 MBSSW10	MBSSW10 Online status of MBISTn (where n = 41:10). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
9 MBSSW9	MBSSW9 Online status of MBISTn (where n = 40:9). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
8 MBSSW8	MBSSW8 Online status of MBISTn (where n = 39:8).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
7 MBSSW7	MBSSW7 Online status of MBISTn (where n = 38:7). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
6 MBSSW6	MBSSW6 Online status of MBISTn (where n = 37:6). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
5 MBSSW5	MBSSW5 Online status of MBISTn (where n = 36:5). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
4 MBSSW4	MBSSW4 Online status of MBISTn (where n = 35:4). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
3 MBSSW3	MBSSW3 Online status of MBISTn (where n = 34:3). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
2 MBSSW2	MBSSW2 Online status of MBISTn (where n = 33:2). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
1 MBSSW1	MBSSW1 Online status of MBISTn (where n = 32:1). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
0	MBSSW0

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBSSW0	Online status of MBISTn (where n = 31:0). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution

71.8.16 STCU2 Online MBIST Status (MBSSW1)

Offset

Register	Offset
MBSSW1	110h

Function

This register includes the results corresponding to the execution of the selected online MBIST in the range 63:32.

The size of the register depends on the number of BISTed RAMs/ROMs.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MBSSW63	MBSSW62	MBSSW61	MBSSW60	MBSSW59	MBSSW58	MBSSW57	MBSSW56	MBSSW55	MBSSW54	MBSSW53	MBSSW52	MBSSW51	MBSSW50	MBSSW49	MBSSW48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MBSSW47	MBSSW46	MBSSW45	MBSSW44	MBSSW43	MBSSW42	MBSSW41	MBSSW40	MBSSW39	MBSSW38	MBSSW37	MBSSW36	MBSSW35	MBSSW34	MBSSW33	MBSSW32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MBSSW63	MBSSW63 Online status of MBISTn (where n = 94:63). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution

Table continues on the next page...

Table continued from the previous page...

Field	Function
30 MBSSW62	MBSSW62 Online status of MBISTn (where n = 93:62). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
29 MBSSW61	MBSSW61 Online status of MBISTn (where n = 92:61). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
28 MBSSW60	MBSSW60 Online status of MBISTn (where n = 91:60). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
27 MBSSW59	MBSSW59 Online status of MBISTn (where n = 90:59). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
26 MBSSW58	MBSSW58 Online status of MBISTn (where n = 89:58). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
25 MBSSW57	MBSSW57 Online status of MBISTn (where n = 88:57). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
24 MBSSW56	MBSSW56 Online status of MBISTn (where n = 87:56). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
23 MBSSW55	MBSSW55 Online status of MBISTn (where n = 86:55).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
22 MBSSW54	MBSSW54 Online status of MBISTn (where n = 85:54). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
21 MBSSW53	MBSSW53 Online status of MBISTn (where n = 84:53). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
20 MBSSW52	MBSSW52 Online status of MBISTn (where n = 83:52). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
19 MBSSW51	MBSSW51 Online status of MBISTn (where n = 82:51). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
18 MBSSW50	MBSSW50 Online status of MBISTn (where n = 81:50). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
17 MBSSW49	MBSSW49 Online status of MBISTn (where n = 80:49). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
16 MBSSW48	MBSSW48 Online status of MBISTn (where n = 79:48). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
15	MBSSW47

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBSSW47	Online status of MBISTn (where n = 78:47). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
14 MBSSW46	MBSSW46 Online status of MBISTn (where n = 77:46). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
13 MBSSW45	MBSSW45 Online status of MBISTn (where n = 76:45). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
12 MBSSW44	MBSSW44 Online status of MBISTn (where n = 75:44). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
11 MBSSW43	MBSSW43 Online status of MBISTn (where n = 74:43). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
10 MBSSW42	MBSSW42 Online status of MBISTn (where n = 73:42). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
9 MBSSW41	MBSSW41 Online status of MBISTn (where n = 72:41). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
8 MBSSW40	MBSSW40 Online status of MBISTn (where n = 71:40). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 MBSSW39	MBSSW39 Online status of MBISTn (where n = 70:39). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
6 MBSSW38	MBSSW38 Online status of MBISTn (where n = 69:38). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
5 MBSSW37	MBSSW37 Online status of MBISTn (where n = 68:37). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
4 MBSSW36	MBSSW36 Online status of MBISTn (where n = 67:36). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
3 MBSSW35	MBSSW35 Online status of MBISTn (where n = 66:35). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
2 MBSSW34	MBSSW34 Online status of MBISTn (where n = 65:34). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
1 MBSSW33	MBSSW33 Online status of MBISTn (where n = 64:33). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
0 MBSSW32	MBSSW32 Online status of MBISTn (where n = 63:32).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Failed MBIST execution 1b - No fault detected during the MBIST execution

71.8.17 STCU2 Online MBIST Status (MBSSW2)

Offset

Register	Offset
MBSSW2	114h

Function

This register includes the results corresponding to the execution of the selected online MBIST in the range 95:64.

The size of the register depends on the number of BISTed RAMs/ROMs.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MBSSW95	MBSSW94	MBSSW93	MBSSW92	MBSSW91	MBSSW90	MBSSW89	MBSSW88	MBSSW87	MBSSW86	MBSSW85	MBSSW84	MBSSW83	MBSSW82	MBSSW81	MBSSW80
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MBSSW79	MBSSW78	MBSSW77	MBSSW76	MBSSW75	MBSSW74	MBSSW73	MBSSW72	MBSSW71	MBSSW70	MBSSW69	MBSSW68	MBSSW67	MBSSW66	MBSSW65	MBSSW64
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MBSSW95	MBSSW95 Online status of MBISTn (where n = 126:95). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
30	MBSSW94

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBSSW94	Online status of MBISTn (where n = 125:94). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
29 MBSSW93	MBSSW93 Online status of MBISTn (where n = 124:93). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
28 MBSSW92	MBSSW92 Online status of MBISTn (where n = 123:92). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
27 MBSSW91	MBSSW91 Online status of MBISTn (where n = 122:91). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
26 MBSSW90	MBSSW90 Online status of MBISTn (where n = 121:90). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
25 MBSSW89	MBSSW89 Online status of MBISTn (where n = 120:89). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
24 MBSSW88	MBSSW88 Online status of MBISTn (where n = 119:88). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
23 MBSSW87	MBSSW87 Online status of MBISTn (where n = 118:87). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 MBSSW86	MBSSW86 Online status of MBISTn (where n = 117:86). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
21 MBSSW85	MBSSW85 Online status of MBISTn (where n = 116:85). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
20 MBSSW84	MBSSW84 Online status of MBISTn (where n = 115:84). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
19 MBSSW83	MBSSW83 Online status of MBISTn (where n = 114:83). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
18 MBSSW82	MBSSW82 Online status of MBISTn (where n = 113:82). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
17 MBSSW81	MBSSW81 Online status of MBISTn (where n = 112:81). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
16 MBSSW80	MBSSW80 Online status of MBISTn (where n = 111:80). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
15 MBSSW79	MBSSW79 Online status of MBISTn (where n = 110:79).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
14 MBSSW78	MBSSW78 Online status of MBISTn (where n = 109:78). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
13 MBSSW77	MBSSW77 Online status of MBISTn (where n = 108:77). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
12 MBSSW76	MBSSW76 Online status of MBISTn (where n = 107:76). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
11 MBSSW75	MBSSW75 Online status of MBISTn (where n = 106:75). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
10 MBSSW74	MBSSW74 Online status of MBISTn (where n = 105:74). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
9 MBSSW73	MBSSW73 Online status of MBISTn (where n = 104:73). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
8 MBSSW72	MBSSW72 Online status of MBISTn (where n = 103:72). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
7	MBSSW71

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBSSW71	Online status of MBISTn (where n = 102:71). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
6 MBSSW70	MBSSW70 Online status of MBISTn (where n = 101:70). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
5 MBSSW69	MBSSW69 Online status of MBISTn (where n = 100:69). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
4 MBSSW68	MBSSW68 Online status of MBISTn (where n = 99:68). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
3 MBSSW67	MBSSW67 Online status of MBISTn (where n = 98:67). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
2 MBSSW66	MBSSW66 Online status of MBISTn (where n = 97:66). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
1 MBSSW65	MBSSW65 Online status of MBISTn (where n = 96:65). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution
0 MBSSW64	MBSSW64 Online status of MBISTn (where n = 95:64). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
e8462625-4821-48a8-b251-390d3ab624c1

71.8.18 STCU2 Online MBIST Status (MBSSW3)

Offset

Register	Offset
MBSSW3	118h

Function

This register includes the results corresponding to the execution of the selected online MBIST in the range 96:109.

The size of the register depends on the number of BISTed RAMs/ROMs.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		MBSSW109	MBSSW108	MBSSW107	MBSSW106	MBSSW105	MBSSW104	MBSSW103	MBSSW102	MBSSW101	MBSSW100	MBSSW99	MBSSW98	MBSSW97	MBSSW96
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-14 —	Reserved
13-0 MBSSWn	MBSSW Online status of MBISTn (where n = 109:96). 0b - Failed MBIST execution 1b - No fault detected during the MBIST execution

71.8.19 STCU2 Online MBIST End Flag (MBESW0)

Offset

Register	Offset
MBESW0	14Ch

Function

This register includes the End Flag related to the execution of the selected online MBIST in the range 31:0.

The size of the register depends on the number of BISTed RAMs/ROMs.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MBESW31	MBESW30	MBESW29	MBESW28	MBESW27	MBESW26	MBESW25	MBESW24	MBESW23	MBESW22	MBESW21	MBESW20	MBESW19	MBESW18	MBESW17	MBESW16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MBESW15	MBESW14	MBESW13	MBESW12	MBESW11	MBESW10	MBESW9	MBESW8	MBESW7	MBESW6	MBESW5	MBESW4	MBESW3	MBESW2	MBESW1	MBESW0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MBESW31	Online end status of MBISTn (where n = 62:31) 0b - MBIST execution still ongoing 1b - MBIST execution finished
30 MBESW30	Online end status of MBISTn (where n = 61:30) 0b - MBIST execution still ongoing 1b - MBIST execution finished
29 MBESW29	Online end status of MBISTn (where n = 60:29) 0b - MBIST execution still ongoing 1b - MBIST execution finished
28 MBESW28	Online end status of MBISTn (where n = 59:28) 0b - MBIST execution still ongoing 1b - MBIST execution finished
27 MBESW27	Online end status of MBISTn (where n = 58:27) 0b - MBIST execution still ongoing 1b - MBIST execution finished
26	Online end status of MBISTn (where n = 57:26)

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBESW26	0b - MBIST execution still ongoing 1b - MBIST execution finished
25 MBESW25	Online end status of MBISTn (where n = 56:25) 0b - MBIST execution still ongoing 1b - MBIST execution finished
24 MBESW24	Online end status of MBISTn (where n = 55:24) 0b - MBIST execution still ongoing 1b - MBIST execution finished
23 MBESW23	Online end status of MBISTn (where n = 54:23) 0b - MBIST execution still ongoing 1b - MBIST execution finished
22 MBESW22	Online end status of MBISTn (where n = 53:22) 0b - MBIST execution still ongoing 1b - MBIST execution finished
21 MBESW21	Online end status of MBISTn (where n = 52:21) 0b - MBIST execution still ongoing 1b - MBIST execution finished
20 MBESW20	Online end status of MBISTn (where n = 51:20) 0b - MBIST execution still ongoing 1b - MBIST execution finished
19 MBESW19	Online end status of MBISTn (where n = 50:19) 0b - MBIST execution still ongoing 1b - MBIST execution finished
18 MBESW18	Online end status of MBISTn (where n = 49:18) 0b - MBIST execution still ongoing 1b - MBIST execution finished
17 MBESW17	Online end status of MBISTn (where n = 48:17) 0b - MBIST execution still ongoing 1b - MBIST execution finished
16	Online end status of MBISTn (where n = 47:16)

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBESW16	0b - MBIST execution still ongoing 1b - MBIST execution finished
15 MBESW15	Online end status of MBISTn (where n = 46:15) 0b - MBIST execution still ongoing 1b - MBIST execution finished
14 MBESW14	Online end status of MBISTn (where n = 45:14) 0b - MBIST execution still ongoing 1b - MBIST execution finished
13 MBESW13	Online end status of MBISTn (where n = 44:13) 0b - MBIST execution still ongoing 1b - MBIST execution finished
12 MBESW12	Online end status of MBISTn (where n = 43:12) 0b - MBIST execution still ongoing 1b - MBIST execution finished
11 MBESW11	Online end status of MBISTn (where n = 42:11) 0b - MBIST execution still ongoing 1b - MBIST execution finished
10 MBESW10	Online end status of MBISTn (where n = 41:10) 0b - MBIST execution still ongoing 1b - MBIST execution finished
9 MBESW9	Online end status of MBISTn (where n = 40:9) 0b - MBIST execution still ongoing 1b - MBIST execution finished
8 MBESW8	Online end status of MBISTn (where n = 39:8) 0b - MBIST execution still ongoing 1b - MBIST execution finished
7 MBESW7	Online end status of MBISTn (where n = 38:7) 0b - MBIST execution still ongoing 1b - MBIST execution finished
6	Online end status of MBISTn (where n = 37:6)

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBESW6	0b - MBIST execution still ongoing 1b - MBIST execution finished
5 MBESW5	Online end status of MBISTn (where n = 36:5) 0b - MBIST execution still ongoing 1b - MBIST execution finished
4 MBESW4	Online end status of MBISTn (where n = 35:4) 0b - MBIST execution still ongoing 1b - MBIST execution finished
3 MBESW3	Online end status of MBISTn (where n = 34:3) 0b - MBIST execution still ongoing 1b - MBIST execution finished
2 MBESW2	Online end status of MBISTn (where n = 33:2) 0b - MBIST execution still ongoing 1b - MBIST execution finished
1 MBESW1	Online end status of MBISTn (where n = 32:1) 0b - MBIST execution still ongoing 1b - MBIST execution finished
0 MBESW0	Online end status of MBISTn (where n = 31:0) 0b - MBIST execution still ongoing 1b - MBIST execution finished

71.8.20 STCU2 Online MBIST End Flag (MBESW1)

Offset

Register	Offset
MBESW1	150h

Function

This register includes the End Flag related to the execution of the selected online MBIST in the range 63:32.

The size of the register depends on the number of BISTed RAMs/ROMs.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca6625-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MBESW63	MBESW62	MBESW61	MBESW60	MBESW59	MBESW58	MBESW57	MBESW56	MBESW55	MBESW54	MBESW53	MBESW52	MBESW51	MBESW50	MBESW49	MBESW48
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MBESW47	MBESW46	MBESW45	MBESW44	MBESW43	MBESW42	MBESW41	MBESW40	MBESW39	MBESW38	MBESW37	MBESW36	MBESW35	MBESW34	MBESW33	MBESW32
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MBESW63	Online end status of MBISTn (where n = 94:63) 0b - MBIST execution still ongoing 1b - MBIST execution finished
30 MBESW62	Online end status of MBISTn (where n = 93:62) 0b - MBIST execution still ongoing 1b - MBIST execution finished
29 MBESW61	Online end status of MBISTn (where n = 92:61) 0b - MBIST execution still ongoing 1b - MBIST execution finished
28 MBESW60	Online end status of MBISTn (where n = 91:60) 0b - MBIST execution still ongoing 1b - MBIST execution finished
27 MBESW59	Online end status of MBISTn (where n = 90:59) 0b - MBIST execution still ongoing 1b - MBIST execution finished
26 MBESW58	Online end status of MBISTn (where n = 89:58) 0b - MBIST execution still ongoing 1b - MBIST execution finished
25 MBESW57	Online end status of MBISTn (where n = 88:57) 0b - MBIST execution still ongoing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - MBIST execution finished
24 MBESW56	Online end status of MBISTn (where n = 87:56) 0b - MBIST execution still ongoing 1b - MBIST execution finished
23 MBESW55	Online end status of MBISTn (where n = 86:55) 0b - MBIST execution still ongoing 1b - MBIST execution finished
22 MBESW54	Online end status of MBISTn (where n = 85:54) 0b - MBIST execution still ongoing 1b - MBIST execution finished
21 MBESW53	Online end status of MBISTn (where n = 84:53) 0b - MBIST execution still ongoing 1b - MBIST execution finished
20 MBESW52	Online end status of MBISTn (where n = 83:52) 0b - MBIST execution still ongoing 1b - MBIST execution finished
19 MBESW51	Online end status of MBISTn (where n = 82:51) 0b - MBIST execution still ongoing 1b - MBIST execution finished
18 MBESW50	Online end status of MBISTn (where n = 81:50) 0b - MBIST execution still ongoing 1b - MBIST execution finished
17 MBESW49	Online end status of MBISTn (where n = 80:49) 0b - MBIST execution still ongoing 1b - MBIST execution finished
16 MBESW48	Online end status of MBISTn (where n = 79:48) 0b - MBIST execution still ongoing 1b - MBIST execution finished
15 MBESW47	Online end status of MBISTn (where n = 78:47)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MBIST execution still ongoing 1b - MBIST execution finished
14 MBESW46	Online end status of MBISTn (where n = 77:46) 0b - MBIST execution still ongoing 1b - MBIST execution finished
13 MBESW45	Online end status of MBISTn (where n = 76:45) 0b - MBIST execution still ongoing 1b - MBIST execution finished
12 MBESW44	Online end status of MBISTn (where n = 75:44) 0b - MBIST execution still ongoing 1b - MBIST execution finished
11 MBESW43	Online end status of MBISTn (where n = 74:43) 0b - MBIST execution still ongoing 1b - MBIST execution finished
10 MBESW42	Online end status of MBISTn (where n = 73:42) 0b - MBIST execution still ongoing 1b - MBIST execution finished
9 MBESW41	Online end status of MBISTn (where n = 72:41) 0b - MBIST execution still ongoing 1b - MBIST execution finished
8 MBESW40	Online end status of MBISTn (where n = 71:40) 0b - MBIST execution still ongoing 1b - MBIST execution finished
7 MBESW39	Online end status of MBISTn (where n = 70:39) 0b - MBIST execution still ongoing 1b - MBIST execution finished
6 MBESW38	Online end status of MBISTn (where n = 69:38) 0b - MBIST execution still ongoing 1b - MBIST execution finished
5	Online end status of MBISTn (where n = 68:37)

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBESW37	0b - MBIST execution still ongoing 1b - MBIST execution finished
4 MBESW36	Online end status of MBISTn (where n = 67:36) 0b - MBIST execution still ongoing 1b - MBIST execution finished
3 MBESW35	Online end status of MBISTn (where n = 66:35) 0b - MBIST execution still ongoing 1b - MBIST execution finished
2 MBESW34	Online end status of MBISTn (where n = 65:34) 0b - MBIST execution still ongoing 1b - MBIST execution finished
1 MBESW33	Online end status of MBISTn (where n = 64:33) 0b - MBIST execution still ongoing 1b - MBIST execution finished
0 MBESW32	Online end status of MBISTn (where n = 63:32) 0b - MBIST execution still ongoing 1b - MBIST execution finished

71.8.21 STCU2 Online MBIST End Flag (MBESW2)

Offset

Register	Offset
MBESW2	154h

Function

This register includes the End Flag related to the execution of the selected online MBIST in the range 95:64.

The size of the register depends on the number of BISTed RAMs/ROMs.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4620c-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MBESW95	MBESW94	MBESW93	MBESW92	MBESW91	MBESW90	MBESW89	MBESW88	MBESW87	MBESW86	MBESW85	MBESW84	MBESW83	MBESW82	MBESW81	MBESW80
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MBESW79	MBESW78	MBESW77	MBESW76	MBESW75	MBESW74	MBESW73	MBESW72	MBESW71	MBESW70	MBESW69	MBESW68	MBESW67	MBESW66	MBESW65	MBESW64
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MBESW95	Online end status of MBISTn (where n = 126:95) 0b - MBIST execution still ongoing 1b - MBIST execution finished
30 MBESW94	Online end status of MBISTn (where n = 125:94) 0b - MBIST execution still ongoing 1b - MBIST execution finished
29 MBESW93	Online end status of MBISTn (where n = 124:93) 0b - MBIST execution still ongoing 1b - MBIST execution finished
28 MBESW92	Online end status of MBISTn (where n = 123:92) 0b - MBIST execution still ongoing 1b - MBIST execution finished
27 MBESW91	Online end status of MBISTn (where n = 122:91) 0b - MBIST execution still ongoing 1b - MBIST execution finished
26 MBESW90	Online end status of MBISTn (where n = 121:90) 0b - MBIST execution still ongoing 1b - MBIST execution finished
25 MBESW89	Online end status of MBISTn (where n = 120:89) 0b - MBIST execution still ongoing

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - MBIST execution finished
24 MBESW88	Online end status of MBISTn (where n = 119:88) 0b - MBIST execution still ongoing 1b - MBIST execution finished
23 MBESW87	Online end status of MBISTn (where n = 118:87) 0b - MBIST execution still ongoing 1b - MBIST execution finished
22 MBESW86	Online end status of MBISTn (where n = 117:86) 0b - MBIST execution still ongoing 1b - MBIST execution finished
21 MBESW85	Online end status of MBISTn (where n = 116:85) 0b - MBIST execution still ongoing 1b - MBIST execution finished
20 MBESW84	Online end status of MBISTn (where n = 115:84) 0b - MBIST execution still ongoing 1b - MBIST execution finished
19 MBESW83	Online end status of MBISTn (where n = 114:83) 0b - MBIST execution still ongoing 1b - MBIST execution finished
18 MBESW82	Online end status of MBISTn (where n = 113:82) 0b - MBIST execution still ongoing 1b - MBIST execution finished
17 MBESW81	Online end status of MBISTn (where n = 112:81) 0b - MBIST execution still ongoing 1b - MBIST execution finished
16 MBESW80	Online end status of MBISTn (where n = 111:80) 0b - MBIST execution still ongoing 1b - MBIST execution finished
15 MBESW79	Online end status of MBISTn (where n = 110:79)

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - MBIST execution still ongoing 1b - MBIST execution finished
14 MBESW78	Online end status of MBISTn (where n = 109:78) 0b - MBIST execution still ongoing 1b - MBIST execution finished
13 MBESW77	Online end status of MBISTn (where n = 108:77) 0b - MBIST execution still ongoing 1b - MBIST execution finished
12 MBESW76	Online end status of MBISTn (where n = 107:76) 0b - MBIST execution still ongoing 1b - MBIST execution finished
11 MBESW75	Online end status of MBISTn (where n = 106:75) 0b - MBIST execution still ongoing 1b - MBIST execution finished
10 MBESW74	Online end status of MBISTn (where n = 105:74) 0b - MBIST execution still ongoing 1b - MBIST execution finished
9 MBESW73	Online end status of MBISTn (where n = 104:73) 0b - MBIST execution still ongoing 1b - MBIST execution finished
8 MBESW72	Online end status of MBISTn (where n = 103:72) 0b - MBIST execution still ongoing 1b - MBIST execution finished
7 MBESW71	Online end status of MBISTn (where n = 102:71) 0b - MBIST execution still ongoing 1b - MBIST execution finished
6 MBESW70	Online end status of MBISTn (where n = 101:70) 0b - MBIST execution still ongoing 1b - MBIST execution finished
5	Online end status of MBISTn (where n = 100:69)

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBESW69	0b - MBIST execution still ongoing 1b - MBIST execution finished
4 MBESW68	Online end status of MBISTn (where n = 99:68) 0b - MBIST execution still ongoing 1b - MBIST execution finished
3 MBESW67	Online end status of MBISTn (where n = 98:67) 0b - MBIST execution still ongoing 1b - MBIST execution finished
2 MBESW66	Online end status of MBISTn (where n = 97:66) 0b - MBIST execution still ongoing 1b - MBIST execution finished
1 MBESW65	Online end status of MBISTn (where n = 96:65) 0b - MBIST execution still ongoing 1b - MBIST execution finished
0 MBESW64	Online end status of MBISTn (where n = 95:64) 0b - MBIST execution still ongoing 1b - MBIST execution finished

71.8.22 STCU2 Online MBIST End Flag (MBESW3)

Offset

Register	Offset
MBESW3	158h

Function

This register includes the End Flag related to the execution of the selected online MBIST in the range 96:109.

The size of the register depends on the number of BISTed RAMs/ROMs.

The content of this register is initialized to its reset value when RUNSW[RUNSW] is set to 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4620c-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		MBESW109	MBESW108	MBESW107	MBESW106	MBESW105	MBESW104	MBESW103	MBESW102	MBESW101	MBESW100	MBESW99	MBESW98	MBESW97	MBESW96
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-14 —	Reserved
13-0 MBESWn	Online end status of MBISTn (where n = 109:96). 0b - MBIST execution still ongoing 1b - MBIST execution finished

71.8.23 STCU2 MBIST Unrecoverable FM (MBUFM0)

Offset

Register	Offset
MBUFM0	18Ch

Function

This register defines the fault mapping, in terms of Unrecoverable or Recoverable fault, of the MBIST in the range 31:0.

The size of the register depends on the number of BISTed RAMs/ROMs.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

NOTE

The register reset value is chip-specific. See the chip-specific STCU2 information for reset values.

Provided under NDA only
COMPANY PROPRIETARY
Beijing C&I Automotive Technology Co., Ltd.
c8c462e5-4921-48a8-b251-390a3ab624c6

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MBUFM31	MBUFM30	MBUFM29	MBUFM28	MBUFM27	MBUFM26	MBUFM25	MBUFM24	MBUFM23	MBUFM22	MBUFM21	MBUFM20	MBUFM19	MBUFM18	MBUFM17	MBUFM16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MBUFM15	MBUFM14	MBUFM13	MBUFM12	MBUFM11	MBUFM10	MBUFM9	MBUFM8	MBUFM7	MBUFM6	MBUFM5	MBUFM4	MBUFM3	MBUFM2	MBUFM1	MBUFM0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MBUFM31	MBUFMn Online end status of MBISTn (where n = 62:31). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
30 MBUFM30	MBUFMn Online end status of MBISTn (where n = 61:30). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
29 MBUFM29	MBUFMn Online end status of MBISTn (where n = 60:29). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
28 MBUFM28	MBUFMn Online end status of MBISTn (where n = 59:28). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
27 MBUFM27	MBUFMn Online end status of MBISTn (where n = 58:27). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
26 MBUFM26	MBUFMn Online end status of MBISTn (where n = 57:26).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
25 MBUFM25	MBUFMn Online end status of MBISTn (where n = 56:25). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
24 MBUFM24	MBUFMn Online end status of MBISTn (where n = 55:24). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
23 MBUFM23	MBUFMn Online end status of MBISTn (where n = 54:23). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
22 MBUFM22	MBUFMn Online end status of MBISTn (where n = 53:22). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
21 MBUFM21	MBUFMn Online end status of MBISTn (where n = 52:21). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
20 MBUFM20	MBUFMn Online end status of MBISTn (where n = 51:20). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
19 MBUFM19	MBUFMn Online end status of MBISTn (where n = 50:19). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
18	MBUFMn

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBUFM18	Online end status of MBISTn (where n = 49:18). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
17 MBUFM17	MBUFMn Online end status of MBISTn (where n = 48:17). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
16 MBUFM16	MBUFMn Online end status of MBISTn (where n = 47:16). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
15 MBUFM15	MBUFMn Online end status of MBISTn (where n = 46:15). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
14 MBUFM14	MBUFMn Online end status of MBISTn (where n = 45:14). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
13 MBUFM13	MBUFMn Online end status of MBISTn (where n = 44:13). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
12 MBUFM12	MBUFMn Online end status of MBISTn (where n = 43:12). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
11 MBUFM11	MBUFMn Online end status of MBISTn (where n = 42:11). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 MBUFM10	MBUFMn Online end status of MBISTn (where n = 41:10). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
9 MBUFM9	MBUFMn Online end status of MBISTn (where n = 40:9). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
8 MBUFM8	MBUFMn Online end status of MBISTn (where n = 39:8). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
7 MBUFM7	MBUFMn Online end status of MBISTn (where n = 38:7). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
6 MBUFM6	MBUFMn Online end status of MBISTn (where n = 37:6). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
5 MBUFM5	MBUFMn Online end status of MBISTn (where n = 36:5). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
4 MBUFM4	MBUFMn Online end status of MBISTn (where n = 35:4). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
3 MBUFM3	MBUFMn Online end status of MBISTn (where n = 34:3).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
2 MBUFM2	MBUFMn Online end status of MBISTn (where n = 33:2). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
1 MBUFM1	MBUFMn Online end status of MBISTn (where n = 32:1). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
0 MBUFM0	MBUFMn Online end status of MBISTn (where n = 31:0). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping

71.8.24 STCU2 MBIST Unrecoverable FM (MBUFM1)

Offset

Register	Offset
MBUFM1	190h

Function

This register defines the fault mapping, in terms of Unrecoverable or Recoverable fault, of the MBIST in the range 63:32.

The size of the register depends on the number of BISTed RAMs/ROMs.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

NOTE

The register reset value is chip-specific. See the chip-specific STCU2 information for reset values.

Provided under NDA
COMPANY PROPRIETARY
Beijing C-TA Automotive Technology Co., Ltd.
c8c4625-48a1-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	MBUFM63	MBUFM62	MBUFM61	MBUFM60	MBUFM59	MBUFM58	MBUFM57	MBUFM56	MBUFM55	MBUFM54	MBUFM53	MBUFM52	MBUFM51	MBUFM50	MBUFM49	MBUFM48
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MBUFM47	MBUFM46	MBUFM45	MBUFM44	MBUFM43	MBUFM42	MBUFM41	MBUFM40	MBUFM39	MBUFM38	MBUFM37	MBUFM36	MBUFM35	MBUFM34	MBUFM33	MBUFM32
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MBUFM63	MBUFMn Online end status of MBISTn (where n = 94:63). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
30 MBUFM62	MBUFMn Online end status of MBISTn (where n = 93:62). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
29 MBUFM61	MBUFMn Online end status of MBISTn (where n = 92:61). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
28 MBUFM60	MBUFMn Online end status of MBISTn (where n = 91:60). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
27 MBUFM59	MBUFMn Online end status of MBISTn (where n = 90:59). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
26 MBUFM58	MBUFMn Online end status of MBISTn (where n = 89:58).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
25 MBUFM57	MBUFMn Online end status of MBISTn (where n = 88:57). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
24 MBUFM56	MBUFMn Online end status of MBISTn (where n = 87:56). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
23 MBUFM55	MBUFMn Online end status of MBISTn (where n = 86:55). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
22 MBUFM54	MBUFMn Online end status of MBISTn (where n = 85:54). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
21 MBUFM53	MBUFMn Online end status of MBISTn (where n = 84:53). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
20 MBUFM52	MBUFMn Online end status of MBISTn (where n = 83:52). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
19 MBUFM51	MBUFMn Online end status of MBISTn (where n = 82:51). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
18	MBUFMn

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBUFM50	Online end status of MBISTn (where n = 81:50). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
17 MBUFM49	MBUFMn Online end status of MBISTn (where n = 80:49). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
16 MBUFM48	MBUFMn Online end status of MBISTn (where n = 79:48). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
15 MBUFM47	MBUFMn Online end status of MBISTn (where n = 78:47). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
14 MBUFM46	MBUFMn Online end status of MBISTn (where n = 77:46). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
13 MBUFM45	MBUFMn Online end status of MBISTn (where n = 76:45). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
12 MBUFM44	MBUFMn Online end status of MBISTn (where n = 75:44). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
11 MBUFM43	MBUFMn Online end status of MBISTn (where n = 74:43). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 MBUFM42	MBUFMn Online end status of MBISTn (where n = 73:42). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
9 MBUFM41	MBUFMn Online end status of MBISTn (where n = 72:41). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
8 MBUFM40	MBUFMn Online end status of MBISTn (where n = 71:40). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
7 MBUFM39	MBUFMn Online end status of MBISTn (where n = 70:39). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
6 MBUFM38	MBUFMn Online end status of MBISTn (where n = 69:38). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
5 MBUFM37	MBUFMn Online end status of MBISTn (where n = 68:37). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
4 MBUFM36	MBUFMn Online end status of MBISTn (where n = 67:36). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
3 MBUFM35	MBUFMn Online end status of MBISTn (where n = 66:35).

Table continues on the next page...

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8460c3c-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
	0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
2 MBUFM34	MBUFMn Online end status of MBISTn (where n = 65:34). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
1 MBUFM33	MBUFMn Online end status of MBISTn (where n = 64:33). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
0 MBUFM32	MBUFMn Online end status of MBISTn (where n = 63:32). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping

71.8.25 STCU2 MBIST Unrecoverable FM (MBUFM2)

Offset

Register	Offset
MBUFM2	194h

Function

This register defines the fault mapping, in terms of Unrecoverable or Recoverable fault, of the MBIST in the range 95:64.

The size of the register depends on the number of BISTed RAMs/ROMs.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

NOTE

The register reset value is chip-specific. See the chip-specific STCU2 information for reset values.

Provided under NDA
COMPANY PROPRIETARY
Beijing C-Automotive Technology Co., Ltd.
c8c4625-491-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MBUFM95	MBUFM94	MBUFM93	MBUFM92	MBUFM91	MBUFM90	MBUFM89	MBUFM88	MBUFM87	MBUFM86	MBUFM85	MBUFM84	MBUFM83	MBUFM82	MBUFM81	MBUFM80
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MBUFM79	MBUFM78	MBUFM77	MBUFM76	MBUFM75	MBUFM74	MBUFM73	MBUFM72	MBUFM71	MBUFM70	MBUFM69	MBUFM68	MBUFM67	MBUFM66	MBUFM65	MBUFM64
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 MBUFM95	MBUFMn Online end status of MBISTn (where n = 126:95). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
30 MBUFM94	MBUFMn Online end status of MBISTn (where n = 125:94). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
29 MBUFM93	MBUFMn Online end status of MBISTn (where n = 124:93). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
28 MBUFM92	MBUFMn Online end status of MBISTn (where n = 123:92). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
27 MBUFM91	MBUFMn Online end status of MBISTn (where n = 122:91). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
26 MBUFM90	MBUFMn Online end status of MBISTn (where n = 121:90).

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
25 MBUFM89	MBUFMn Online end status of MBISTn (where n = 120:89). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
24 MBUFM88	MBUFMn Online end status of MBISTn (where n = 119:88). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
23 MBUFM87	MBUFMn Online end status of MBISTn (where n = 118:87). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
22 MBUFM86	MBUFMn Online end status of MBISTn (where n = 117:86). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
21 MBUFM85	MBUFMn Online end status of MBISTn (where n = 116:85). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
20 MBUFM84	MBUFMn Online end status of MBISTn (where n = 115:84). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
19 MBUFM83	MBUFMn Online end status of MBISTn (where n = 114:83). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
18	MBUFMn

Table continues on the next page...

Table continued from the previous page...

Field	Function
MBUFM82	Online end status of MBISTn (where n = 113:82). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
17 MBUFM81	MBUFMn Online end status of MBISTn (where n = 112:81). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
16 MBUFM80	MBUFMn Online end status of MBISTn (where n = 111:80). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
15 MBUFM79	MBUFMn Online end status of MBISTn (where n = 110:79). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
14 MBUFM78	MBUFMn Online end status of MBISTn (where n = 109:78). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
13 MBUFM77	MBUFMn Online end status of MBISTn (where n = 108:77). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
12 MBUFM76	MBUFMn Online end status of MBISTn (where n = 107:76). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
11 MBUFM75	MBUFMn Online end status of MBISTn (where n = 106:75). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 MBUFM74	MBUFMn Online end status of MBISTn (where n = 105:74). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
9 MBUFM73	MBUFMn Online end status of MBISTn (where n = 104:73). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
8 MBUFM72	MBUFMn Online end status of MBISTn (where n = 103:72). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
7 MBUFM71	MBUFMn Online end status of MBISTn (where n = 102:71). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
6 MBUFM70	MBUFMn Online end status of MBISTn (where n = 101:70). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
5 MBUFM69	MBUFMn Online end status of MBISTn (where n = 100:69). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
4 MBUFM68	MBUFMn Online end status of MBISTn (where n = 99:68). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
3 MBUFM67	MBUFMn Online end status of MBISTn (where n = 98:67).

Table continues on the next page...

Provided under NDA to Beijing CHJ Automotive Technology Co., Ltd.
COMPANY PROPRIETARY
8460c3c-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
	0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
2 MBUFM66	MBUFMn Online end status of MBISTn (where n = 97:66). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
1 MBUFM65	MBUFMn Online end status of MBISTn (where n = 96:65). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping
0 MBUFM64	MBUFMn Online end status of MBISTn (where n = 95:64). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping

71.8.26 STCU2 MBIST Unrecoverable FM (MBUFM3)

Offset

Register	Offset
MBUFM3	198h

Function

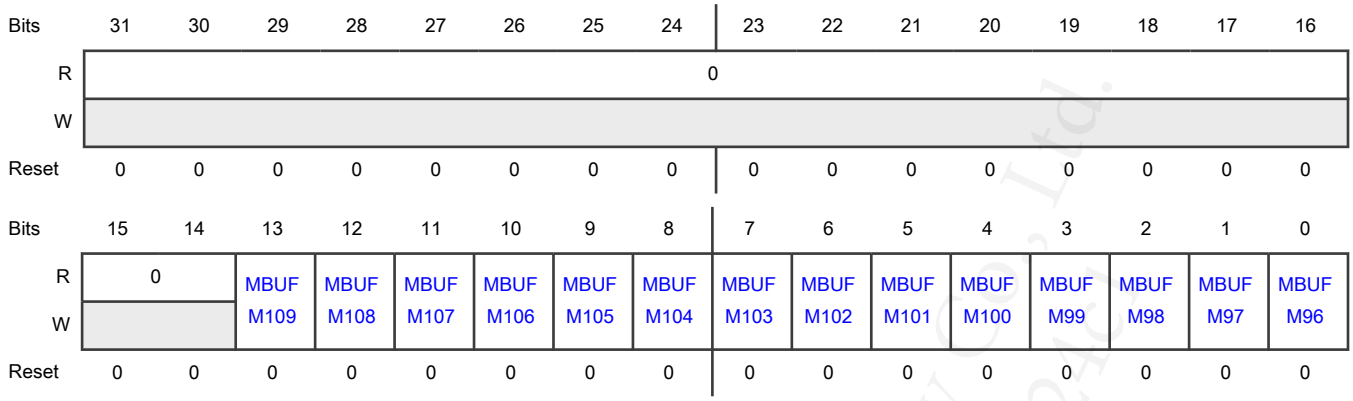
This register defines the fault mapping, in terms of UF or RF, of the MBIST in the range 96:109

The size of the register depends on the number of BISTed RAMs/ROMs.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing C-TA Automotive Technology Co., Ltd.
c8c462e5-4891-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0 MBUFMn	MBESW Online end status of MBISTn (where n = 109:96). 0b - Recoverable fault mapping 1b - Unrecoverable fault mapping

71.8.27 STCU2 LBIST Control (LB_CTRL0 - LB_CTRL59)

Offset

For a = 0 to 59:

Register	Offset
LB_CTRLa	200h + (a × 40h)

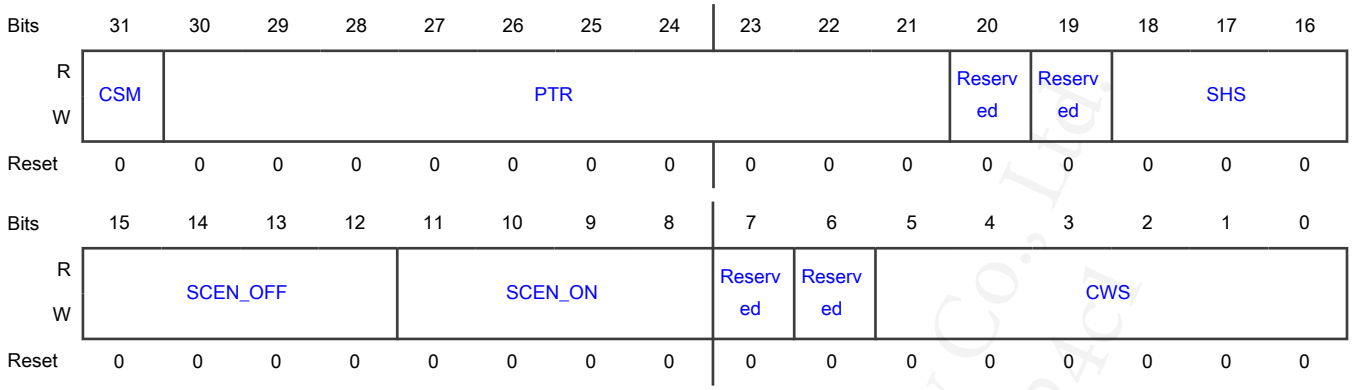
Function

This register defines the control setting of each LBIST controller.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-b251-390d8ab62f4c

Diagram



Fields

Field	Function
31 CSM	Concurrent/sequential mode The next LBIST is scheduled concurrently to the current one if the CSM bit is set to 1; otherwise, it is scheduled sequentially to the completion of the current LBIST execution. 0b - Sequential mode 1b - Concurrent mode
30-21 PTR	Next LBIST or MBIST pointer PTR defines the logical pointer to the next LBIST or MBIST to be scheduled. The next LBIST or MBIST is scheduled concurrently to the current one if the CSM bit is set to 1, otherwise it is scheduled sequentially to the completion of the current LBIST execution. In case of NIL pointer, the CSM bit has to be set Sequential (0) to define this is the end of the list. The self-testing procedure stops after the last BIST in the configuration chain is complete. See BIST scheduling for details. 0h to (3Ch - 1): pointer to NLBIST-1 00000080h to (00000080h + 06Eh - 1): pointer to MBIST 3FFh: pointer to NIL. No BIST execution. others: invalid pointer => an error is set into the STCU2 Error (ERR_STAT).
20 —	Reserved **NOTE** This reserved field is writable but do not write any value to it other than its reset value
19 —	Reserved **NOTE** This reserved field is writable but do not write any value to it other than its reset value
18-16 SHS	Shift speed SHS defines the shift speed

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000b - Shift at full rate (BIST clock). 001b - Shift at 1/2 rate (BIST clock). 010b - Shift at 1/3 rate (BIST clock). 011b - Shift at 1/4 rate (BIST clock). 100b - Shift at 1/5 rate (BIST clock). 101b - Shift at 1/6 rate (BIST clock). 110b - Shift at 1/7 rate (BIST clock). 111b - Shift at 1/8 rate (BIST clock).
15-12 SCEN_OFF	Scan enable OFF SCEN_OFF information is used to configure the lbist controller hardware to generate off_cycles, delay cycles during the scan enable off transition. **NOTE** SCEN_OFF must be programmed to a value >=1. 0000b - 0 delay cycles 0001b - 1 delay cycle 0010b - 2 delay cycles 0011b - 3 delay cycles 0100b - 4 delay cycles 0101b - 5 delay cycles 0110b - 6 delay cycles 0111b - 7 delay cycles 1000b - 8 delay cycles 1001b - 9 delay cycles 1010b - 10 delay cycles 1011b - 11 delay cycles 1100b - 12 delay cycles 1101b - 13 delay cycles 1110b - 14 delay cycles 1111b - 15 delay cycles
11-8 SCEN_ON	Scan enable ON

Table continues on the next page...

Table continued from the previous page...

Field	Function
	SCEN_ON information is used to configure the lbist controller hardware to generate on_cycles, delay cycles during the scan enable on transition, **NOTE** SCEN_ON delay register value must be programmed to a value >=1 0000b - 0 delay cycles 0001b - 1 delay cycle 0010b - 2 delay cycles 0011b - 3 delay cycles 0100b - 4 delay cycles 0101b - 5 delay cycles 0110b - 6 delay cycles 0111b - 7 delay cycles 1000b - 8 delay cycles 1001b - 9 delay cycles 1010b - 10 delay cycles 1011b - 11 delay cycles 1100b - 12 delay cycles 1101b - 13 delay cycles 1110b - 14 delay cycles 1111b - 15 delay cycles
7 —	Reserved
6 —	Reserved
5-0 CWS	Capture window size CWS defines the capture window size. 000000b - Illegal 000001b - Controller waits 1 shift cycle for capture to finish. 000010b - Controller waits 2 shift cycles for capture to finish. 000011b - Controller waits 3 shift cycles for capture to finish. 000100b - Controller waits 4 shift cycles for capture to finish. 000101b - Controller waits 5 shift cycles for capture to finish.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000110b - Controller waits 6 shift cycles for capture to finish. 000111b - Controller waits 7 shift cycles for capture to finish.

71.8.28 STCU2 LBIST PC Stop (LB_PCS0 - LB_PCS59)

Offset

For a = 0 to 59:

Register	Offset
LB_PCSa	204h + (a × 40h)

Function

This register defines the pattern counter stop of each LBIST controller.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								PCS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PCS															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-26 —	Reserved
25-0 PCS	PCS Pattern counter stop PCS defines the pattern counter stop value.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
0c462e5-4821a8-b251-390d3ab624c1

71.8.29 STCU2 Online LBIST MISR Expected Low (LB_MISRELSW0 - LB_MISRELSW59)

Offset

For a = 0 to 59:

Register	Offset
LB_MISRELSWa	220h + (a × 40h)

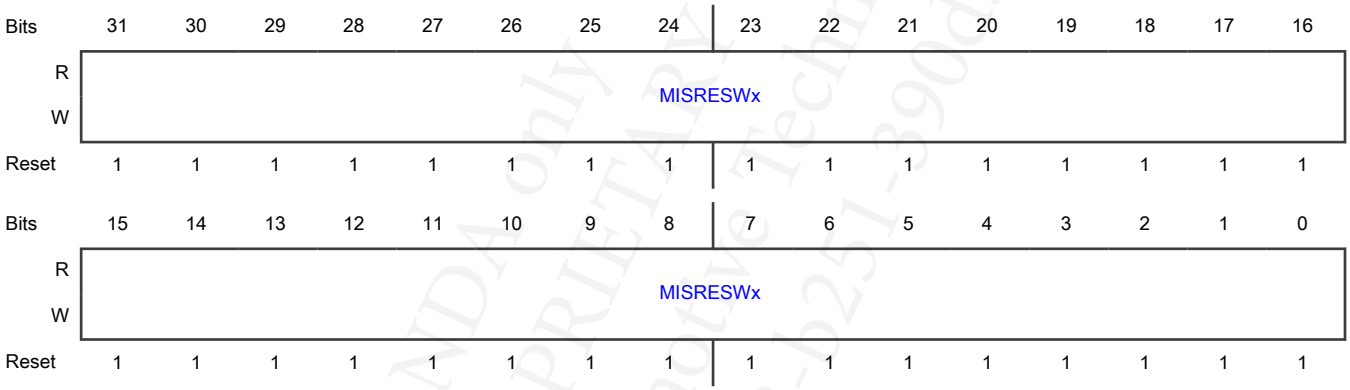
Function

This register defines bits 32 of the expected MISR of the online LBIST controller.

The size of the register depends on the number of MISR bits of the related LBIST. .

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram



Fields

Field	Function
31-0 MISRESWx	Online MISR expected low bits This field defines 32 bits of the expected MISR.

71.8.30 STCU2 Online LBIST MISR Expected High (LB_MISREHSW0 - LB_MISREHSW59)

Offset

For a = 0 to 59:

Register	Offset
LB_MISREHSWa	224h + (a × 40h)

Function

The size of the register depends on the number of MISR bits of the related LBIST. .

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MISRESWx															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MISRESWx															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Fields

Field	Function
31-0 MISRESWx	Online MISR Expected High Bits This field defines the 32 bits of the expected MISR.

71.8.31 STCU2 Online LBIST MISR Read Low (LB_MISRRLSW0 - LB_MISRRLSW59)

Offset

For a = 0 to 59:

Register	Offset
LB_MISRRLSWa	228h + (a × 40h)

Function

This register reports 32 bits of the MISR obtained at the end of the online LBIST controller execution.

The size of the register depends on the number of MISR bits of the related LBIST. .

The content of this register is initialized to its reset value reset value when RUNSW[RUNSW] is set to 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4621-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MISRRSWx															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MISRRSWx															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 MISRRSWx	MISRRSWx Online MISR Read Low Bin This field is equivalent to 32 bits of the MISR obtained at the end of the online LBIST execution.

71.8.32 STCU2 Online LBIST MISR Read High (LB_MISRRHSW0 - LB_MISRRHSW59)

Offset

For a = 0 to 59:

Register	Offset
LB_MISRRHSWa	22Ch + (a × 40h)

Function

The size of the register depends on the number of MISR bits of the related LBIST. .

The content of this register is initialized to its reset value reset value when RUNSW[RUNSW] is set to 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	MISRRSWx															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MISRRSWx															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca0462e5-4821-48a8-b251-30d3ab624c1

Fields

Field	Function
31-0 MISRRSWx	MISRRSWx Online MISR Read High Bits This field is equivalent to 32 bits of the MISR obtained at the end of the online LBIST execution.

71.8.33 STCU2 Algorithm Select (ALGOSEL)

Offset

Register	Offset
ALGOSEL	2200h

Function

This is a 32-bit register intended to be programmed by the user to select algorithms to be run on BIST. See the chip-specific STCU2 information for details of this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL...	ALGO SEL9	ALGO SEL8	ALGO SEL7	ALGO SEL6	ALGO SEL5	ALGO SEL4	ALGO SEL3	ALGO SEL2	ALGO SEL1	ALGO SEL0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ALGOSELn	Algorithm Select

71.8.34 STCU2 MBIST Stagger (STGGR)

Offset

Register	Offset
STGGR	220Ch

Function

This register allows one to program number of clock cycles between execution of one BIST and the next one.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STAG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STAG															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STAG	STAG Number of STCU2 CORE_CLK cycles between execution of one BIST and the next one.

71.8.35 STCU2 BIST Start (BSTART)

Offset

Register	Offset
BSTART	2210h

Function

This is a 32-bit register intended to be programmed by the user to run BISTs with different configuration. See the chip-specific STCU2 information for details of this register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BSTART31	BSTART30	BSTART29	BSTART28	BSTART27	BSTART26	BSTART25	BSTART24	BSTART23	BSTART22	BSTART21	BSTART20	BSTART19	BSTART18	BSTART17	BSTART16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BSTART15	BSTART14	BSTART13	BSTART12	BSTART11	BSTART10	BSTART9	BSTART8	BSTART7	BSTART6	BSTART5	BSTART4	BSTART3	BSTART2	BSTART1	BSTART0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
NXP PROPRIETARY
Visible to GHJ Automotive Technology Co., Ltd.
C6262482148a8-b251390d3ab624c1

Fields

Field	Function
31-0 BSTARTn	BIST Start

71.8.36 STCU2 MBIST Control (MB_CTRL0 - MB_CTRL109)

Offset

For a = 0 to 109:

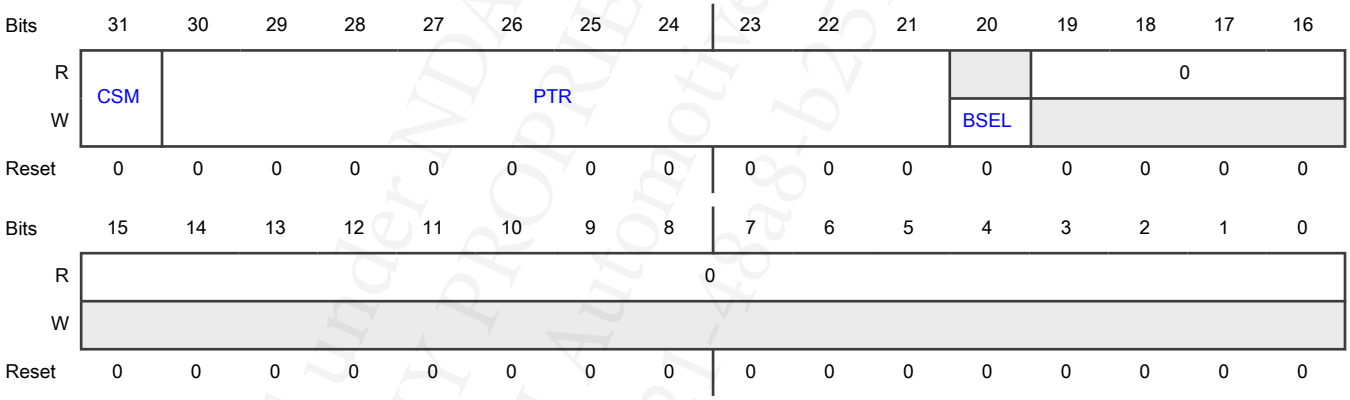
Register	Offset
MB_CTRLa	2214h + (a × 4h)

Function

The MB_CTRL register defines the control setting of MBIST controller.

The R/W fields in this register are readable at any time. You can write to these fields when online self-test phase is active and CFG[WRP] = 0.

Diagram



Fields

Field	Function
31 CSM	CSM Concurrent/sequential mode 0b - Sequential mode 1b - Concurrent mode
30-21	PTR

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-4a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
PTR	Next LBIST or MBIST pointer PTR defines the logical pointer to the next LBIST or MBIST to be scheduled. The next LBIST orMBIST is scheduled concurrently to the current one if the CSM bit is set to 1; otherwise it is scheduled sequentially to the completion of the current MBIST execution. In case of NIL pointer the CSM bit must be set Sequential (0) to define this is the end of the list. The self-testing procedure stops after the last BIST in the configuration chain is complete. See BIST scheduling for details. **NOTE** If the pointer is invalid and MBIST is scheduled to run concurrently than this invalid scenario is handled by watchdog timeout feature and corresponding watchdog timeout status bit will be updated in ERR_STAT[WDTOSW]. In this particular case ERR_STAT[INVPSW] will not be set. 0h to (3Ch - 1): pointer to LBIST 00000080h to (00000080h + 06Eh - 1): pointer to MBIST 3FFh: pointer to NIL. No next BIST execution. others: invalid pointer => an error is set into the STCU2 Error (ERR_STAT).
20 BSEL	BSEL BIST Select 0b - Selected BIST is not selected for execution. 1b - Selected BIST is selected for execution.
19-0 —	Reserved

71.9 Functional description

71.9.1 FSM description

The module has three state machines that work together: Master State Machine, Loader/Shifter State Machine, and the Watchdog State Machine. Basically the Master State Machine is the core unit of the STCU2 module. It coordinates all the self-test operations and the other state machines. The Loader/Shifter State Machine is used to program the MBIST and the LBIST parameters and to retrieve the related results depending on the parameters stored into the STCU2 registers and under the control of the Master State Machine. The Watchdog State Machine evaluates all the schedule time for MBIST and LBIST and the time-out in case of wrong STCU2 programming.

71.9.2 BIST scheduling

STCU2 is designed to program the parallel/serial execution of the MBIST or LBIST depending on the power, timing, and coverage constraints.

The mechanism used to provide this flexibility is a linked list of BIST descriptors where the starting pointer is defined in CFG[PTR]. The first LBIST is mapped on 0, the second on 1, and so on. A BIST descriptor identifies a LBIST or MBIST via an index that is associated with its control register (STCU2 LBIST Control (LB_CTRL0 - LB_CTRL59) or STCU2 MBIST Control (MB_CTRL0 - MB_CTRL109)). The first MBIST is mapped on (00000080h + 0), the second on (00000080h + 1), and so on. The additional pointers of the linked list are in LB_CTRL*n*[PTR] for LBIST *n* (where *n* is the selected LBIST) and MB_CTRL*m*[PTR] for MBIST *m* (where *m* is the selected MBIST) and must be populated depending on the selected run sequence. The additional fields

LB_CTRL[CSM] and MB_CTRL[CSM] provide the flexibility to run concurrently or sequentially the chosen set of the LBIST or MBIST or to close the linked list by setting the NIL pointer.

71.9.3 ABORT management

The STCU2 module provides online self test execution abort. When the abort is detected, the online self test operation stops running and the status of the currently running MBIST or LBIST is saved into the related registers to provide intermediate results that might be useful in case of debug or in case of concurrent run.

71.9.3.1 Hardware ABORT management

The STCU2 enters the hardware abort condition as a consequence of a functional reset which causes self test execution to abort.

When hardware abort is detected the bit ABORTHW of the ERR_STAT register is set. This flag allows the software to diagnose what has happened during the online self test execution run.

71.9.4 FCCU interface

The FCCU interface is the hardware flag mechanism towards the system, to indicate the occurrence of an Unrecoverable fault and/or a Recoverable fault failure during the Self Test sequence.

To diagnose physical defects on the two fault signals, a fault injection mechanism is also provided. In this case, the FCCU interface allows the user application to check the integrity of the UF and RF connection lines between the STCU2 and the FCCU. Refer to the description of FCCU fault injection mechanism to understand how the UF/RF set/clear mechanism works.

71.9.5 Watchdogs

The STCU2 implements different watchdogs to ensure that operations are finished in time.

71.9.5.1 BIST watchdog timer

The LBIST and MBIST execution time has to be configured as described in STCU2 Watchdog Granularity (WDG), to account for the overall execution time of the self test sequence. In case the selected LBISTs or MBISTs are not yet completed during assigned time, the current LBISTs or MBISTs execution is interrupted and a failure is flagged into ERR_STAT[WDTOSW] and MBESWx or LBESW.

In case of multiple sequential run in the same online session and the time-out happens in the middle of a sequential run, the next sequential run will be skipped and the execution ends with the current updated status of the registers reported above.

71.9.5.2 Register write-access watchdog timer

As explained in the STCU2 SK Code (SKC):

- A key mechanism protects STCU2 registers during the self-test configuration phase by preventing any unwanted access.
- A hardware watchdog timer locks register-write access after a number of STCU2 clock cycles. To refresh the hardware watchdog timer before it times out, write Key1 and Key2 sequences.

AUTOLOCK_VALUE is the number of STCU2 clock cycles after which the hardware watchdog timer locks register-write access. See the chip-specific STCU2 information for the value of AUTOLOCK_VALUE.

The hardware watchdog timer is particularly useful in case CFG[WRP] is 0 during the software self-test configuration. In this case, the software application might enable write access to the STCU2 registers.

71.10 Use cases and limitations

For details on use cases and limitations, contact NXP sales representative.

71.11 Glossary

BIST	Built-in self-test
BIST Clock	BIST controller clock corresponding to the specific BIST
CORE_CLK	Clock specified by the CFG register
FSM	Finite state machine
IPS	Internal peripheral system
LBIST	Logic BIST
MBIST	Memory BIST
NLBIST	Number of logic built-in self-test controller
NMCUT	Number of memory checked using memory built-in self-test controller
RF	Recoverable faults
UF	Unrecoverable faults
WDG FSM	Watchdog finite state machine

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 72
Safety by Software (SBSW)

72.1 Introduction

The SBSW is a module that supports the "safety by software" approach in automotive safety applications.

It provides a fault detection mechanism to be used by safety-related applications via two distinct safety mechanisms. SBSW monitors the health of an application by performing the following functions:

- It checks application software results by comparing the outcome of redundant computations in a configured manner with respect to both the values and their temporal relationship.
- It monitors the execution process of an application by comparing the application's real-time progress state with the configured expected progress program flow.

72.1.1 Acronyms and abbreviations

Text	Description
AC	Application core
CPU	Central processing unit
FCCU	Fault collection and control unit
FSM	Finite-state machine
INTC	Interrupt controller
IP	Intellectual property
IRQ	Interrupt request
NCF	Noncritical fault
R	Read
RO	Read only
RW	Read/write
SW	Software
TC	TMWDP core
TCM	Tightly-coupled memory
TMC	Time-monitored comparator
TMWDP	Timed multi-watchdog processor
TMWDPI	TMWDP interface
W	Write
WO	Write only
W1C	Write 1 to clear

72.2 Block diagram

The following figure shows the top-level block diagram of SBSW:

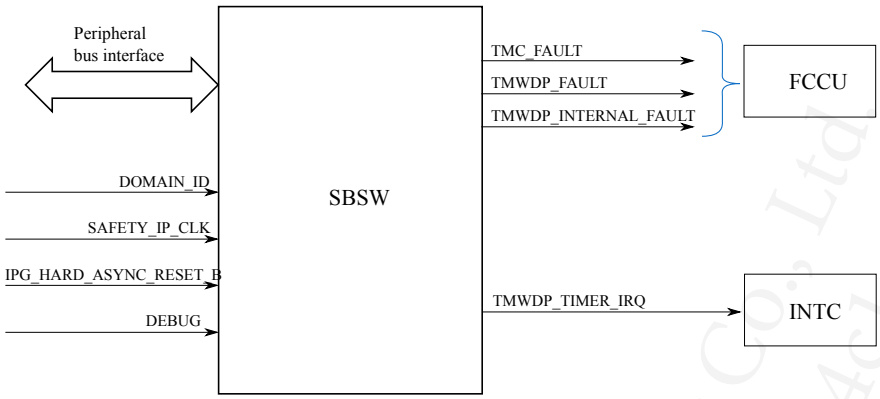


Figure 565. Top-level block diagram

The following figure shows the overall structure of the SBSW.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab6240d

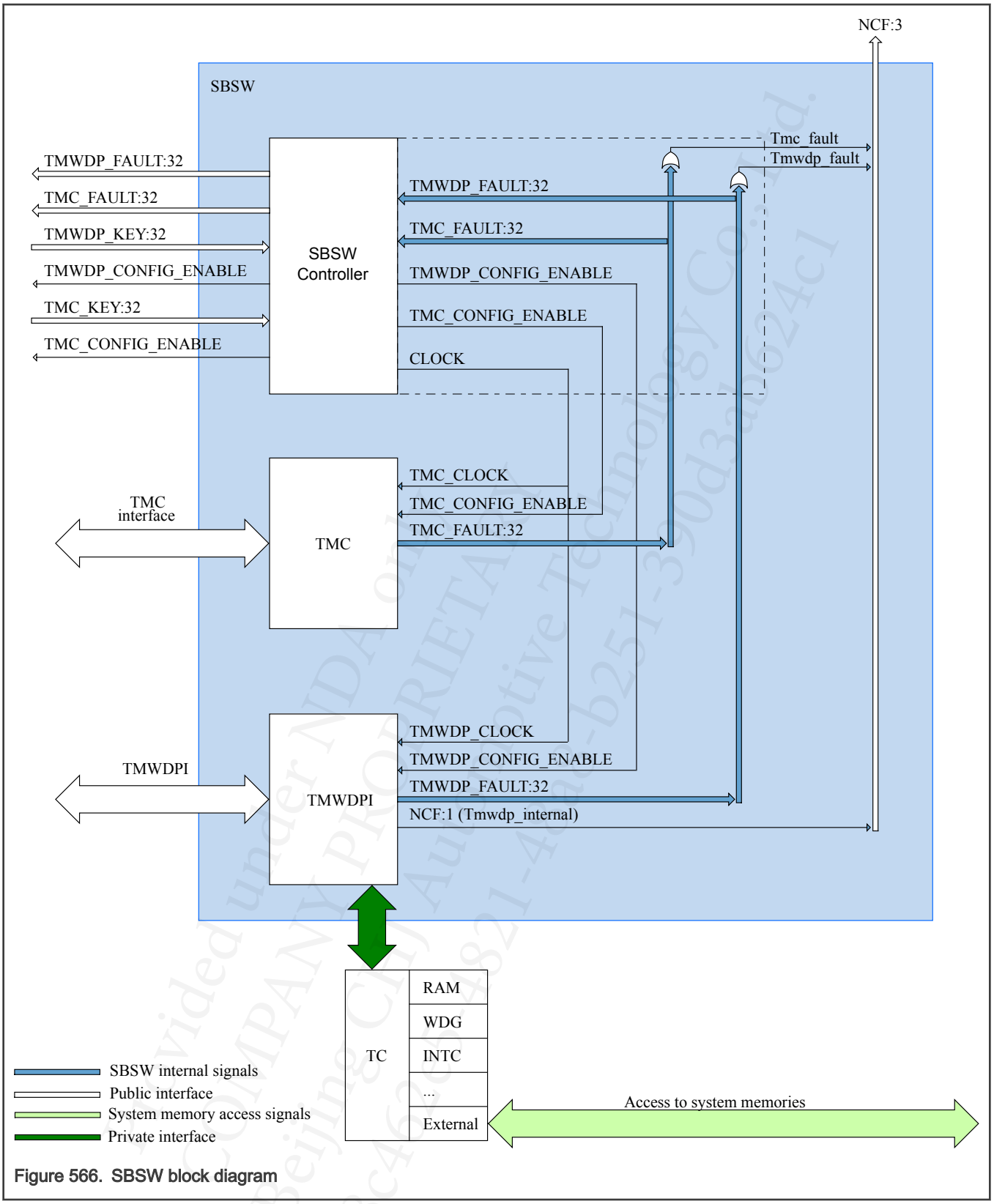


Figure 566. SBSW block diagram

The thin lines represent a single bit or signal, and the thick lines denote multiple-bit entities with names followed by colons and number of bits.

72.3 Signal descriptions

The table below describes the signals on the boundary of SBSW.

Table 566. Signal descriptions

Signal name	Direction	Description
Clock and reset		
SAFETY_IP_CLK	Input	SBSW has a single clock of 66.7 MHz, which is synchronous to the 133 MHz MODULE_CLK. The chip handles synchronization requirement of the clocks outside of the SBSW module.
IPG_HARD_ASYNC_RESET_B	Input	IP bus async functional reset
DEBUG	Input	SBSW enters the Debug mode when this signal is asserted high. The DEBUG_MODE[MODE] field configures the behavior of the SBSW when it is in the Debug mode.
DOMAIN_ID[3:0]	Input	These signals are used to distinguish whether the RW accesses are from TC or AC. If the value on this bus matches the value programmed in the TC_ID register, then RW accesses are from the TC, otherwise RW accesses are from the AC.
FCCU interface signals		
TMC_FAULT	Output	This signal is high when TMC detects a fault. The fault could be either because of false comparison or time violation.
TMWDP_FAULT	Output	This signal is high when TMWDP detects a fault. The fault could be either because of illegal transition or time violation in any automaton.
TMWDP_INTERNAL_FAULT	Output	This signal is high when TMWDP detects an error in the safety mechanism itself. The error could be either because of invalid configuration or an internal error.
INTC interface signals		
TMWDP_TIMER_IRQ	Output	This level interrupt is generated whenever the TMWDP timer reload expires.

72.4 Functional description

SBSW has three main functional blocks (or sub-modules):

- SBSW controller

- Array of TMCs
- TMWDPI

72.4.1 SBSW controller

The SBSW controller includes auxiliary parts of this module, and controls the operation of SBSW sub-modules (TMCs and TMWDPI), as shown in Figure 566. Some of the functions of the SBSW controller are as follows:

- Provides a locking/unlocking mechanism protecting configuration of the sub-modules.
- Provides the clock signals to the sub-modules.
- Provides aggregated fault outputs from the sub-modules.
- Generates NCF signals for FCCU to react.

SBSW controller protects sensitive configuration data by handling a locking/unlocking protocol. The unlocking mechanism is independent for each SBSW sub-module, using a dedicated register and a separate FSM implementing the protocol. You must unlock the individual sub-module's configuration before modifying it.

72.4.1.1 Unlocking/locking the TMC configuration

The following figure illustrates the unlocking protocol for the TMC.

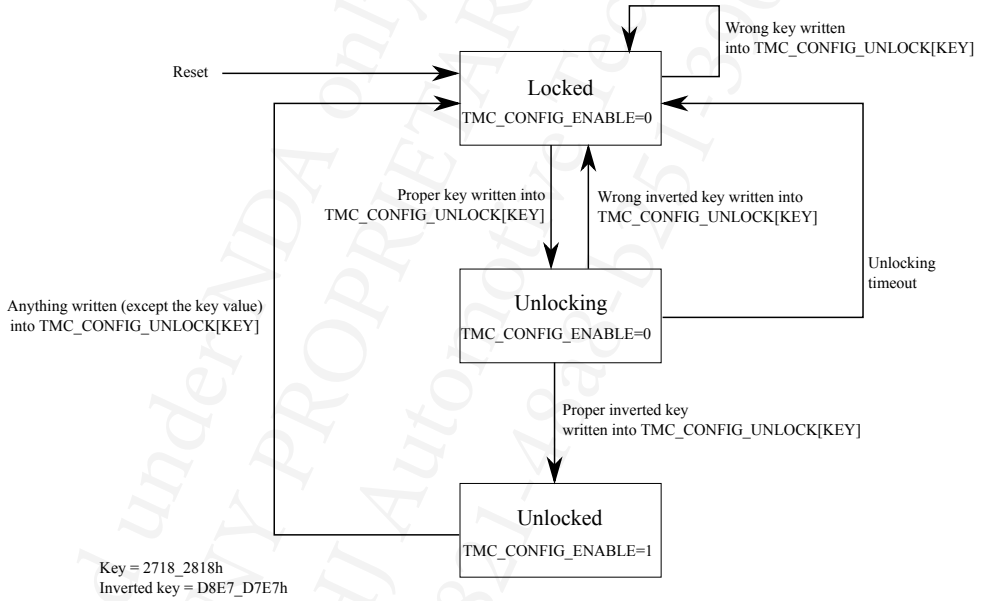


Figure 567. TMC configuration locking state diagram

To unlock the configuration, you must execute the following procedure by using the TMC_CONFIG_UNLOCK register:

1. Write a predefined key value (2718_2818h) to TMC_CONFIG_UNLOCK[KEY].
2. Write the inverted key value (D8E7_D7E7h) to TMC_CONFIG_UNLOCK[KEY] within a predefined timeout interval of 62 SAFETY_IP_CLK cycles.

On successful completion of the procedure described above, the SBSW controller unlocks the TMC configuration and asserts the TMC_CONFIG_ENABLE signal.

If you do not write the inverted key value within the predefined timeout, or if you write a wrong value, the SBSW controller aborts the unlocking operation.

After modifying the configuration, you must lock the TMC configuration by writing any value (except the unlocking key, 2718_2818h) into TMC_CONFIG_UNLOCK[KEY].

NOTE

Before locking the TMC configuration, you must ensure that it stays in the unlocked state for at least one SAFETY_IP_CLK cycle. You can ensure this, for example, by reading the TMC_CONFIG_STATUS[STATUS] field to see whether the TMC configuration unlocking procedure has succeeded.

A successful completion of step 1 (writing the predefined key value) reinitializes the counter and it starts decrementing. Successful completion of the step 2 (writing the inverted key value within a specified time) stops the counter. The counter also stops if the SBSW controller aborts the unlocking operation.

72.4.1.2 Unlocking/locking the TMWDP configuration

To unlock the configuration, you must execute the following procedure by using the TMWDP_CONFIG_UNLOCK register:

1. Write a predefined key value (3141_5926h) to TMWDP_CONFIG_UNLOCK[KEY].
2. Write the inverted key value (CEBE_A6D9h) to TMWDP_CONFIG_UNLOCK[KEY] within a predefined timeout interval of 62 SAFETY_IP_CLK cycles.

On successful completion of the procedure described above, the SBSW controller unlocks the TMWDP configuration and asserts the TMWDP_CONFIG_ENABLE signal.

If you do not write the inverted key value within the predefined timeout, or if you write a wrong value, the SBSW controller aborts the unlocking operation.

After modifying the configuration, you must lock the TMWDP configuration by writing any value (except the unlocking key, 3141_5926h) into TMWDP_CONFIG_UNLOCK[KEY].

NOTE

Before locking the TMWDP configuration, you must ensure that it stays in the unlocked state for at least one SAFETY_IP_CLK cycle. You can ensure this, for example, by reading the TMWDP_CONFIG_STATUS[STATUS] field to see whether the TMWDP configuration unlocking procedure has succeeded.

A successful completion of step 1 (writing the predefined key value) reinitializes the counter and it starts decrementing. Successful completion of the step 2 (writing the inverted key value within a specified time) stops the counter. The counter also stops if the SBSW controller aborts the unlocking operation.

72.4.2 TMC

TMC is an SBSW sub-module that:

- Compares results of two independent application software (SW) instances, SW1 and SW2
- Monitors the comparison timings
- Signals a fault if comparison does not succeed or timing violation occurs

This chip has 32 instances of TMC. Each instance is dedicated to a particular computation comparison. You can enable or disable a TMC instance by using the corresponding TMC_a_CONTROL register. The functionality of TMC is illustrated in the following figure:

Preliminary Proposal
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462b5-4821-48e8-9251-3901f3ab624c

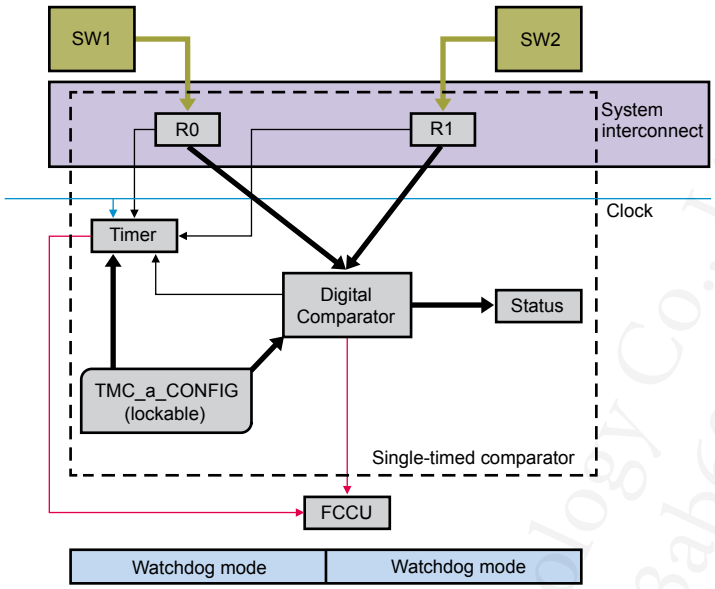


Figure 568. TMC functionality

There are two essential parts of a TMC:

- Digital comparator
- Timer

72.4.2.1 Digital comparator

The digital comparator performs a simple digital comparison (greater than, less than, or equal to) or a complex comparision (with distance value) of the values stored in R0 and R1 registers. The digital comparator signals a fault after any of the following cases:

- If the comparison does not succeed. You configure the comparison modes in the respective TMC configuration register by using TMC_a_CONFIG[COMPARE_MODE].
- If you overwrite either of the values programmed in R0 or R1 before the comparison.

You write the SW1 result in R0 by using TMC_a_R0[R0], and the SW2 result in R1 by using TMC_a_R1[R1].

NOTE

The core on which SW1 or SW2 executes is irrelevant.

The comparison occurs immediately after both R0 and R1 are written a value to compare. The comparison modes are:

- R0 == R1
- R0 > R1
- R0 < R1
- |R1 - R0| ≤ D, Distance

You program the value for Distance in TMC_a_DISTANCE[DISTANCE].

Often, you want to know whether a value (for example, from SW1) is within a certain range with respect to the other value. You can achieve this by comparing a calculation of |R1-R0| with a predefined distance value (D), |R1-R0| ≤ D.

72.4.2.2 Timer

The TMC monitors the comparisons (discussed in Digital comparator) by using a 32-bit count-down timer. Each TMC instance has its own dedicated timer. The TMC signals a fault if the timer expires without a comparison.

You configure a TMC instance by using TMC_a_CONFIG[TIMING_MODE] to run the corresponding timer in either of the following modes:

- **Watchdog mode:**

  The timer ensures that comparison occurs periodically within a specified time limit. If it does not, then TMC signals a fault. You specify the time limit in TMC_a_TIMEOUT[TIMEOUT].

  In this mode, the timer starts decrementing immediately after either of the following events:

  — TMC is enabled.

  — The comparison is complete (that is, when both R0 and R1 registers are written).

  The next comparison shall happen before the timer expires. After the comparison occurs, the timer is loaded again with the timeout value and it starts decrementing again.

- **Window mode:**

  The timer ensures that R0 and R1 registers are written within a specified time limit. If they are not, then TMC signals a fault. You specify the time limit in TMC_a_TIMEOUT[TIMEOUT].

  In this mode, the timer starts decrementing when either of the R0 or R1 register is written. The other register must be written before the timer expires.

The following figures illustrate both the modes. The DISABLED state is reachable from any other state (except from COMPARING); these transitions have been omitted for the sake of readability.

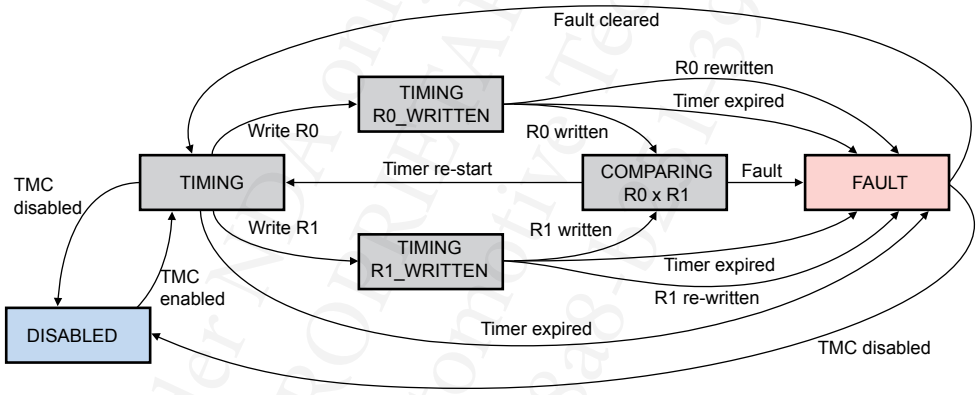


Figure 569. TMC in Watchdog mode

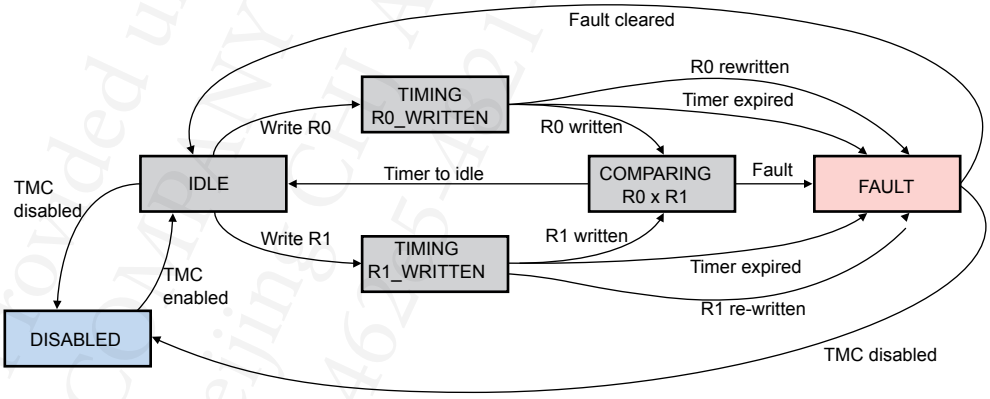


Figure 570. TMC in Window mode

See TMC Configuration (TMC_0_CONFIG - TMC_31_CONFIG) for the configuration registers, which you use to:

- Set the TMC behavior

- Set the TMC timeout value
- Enable a TMC instance

You may read the current value of the timer from TMC Timer (TMC_0_TIMER - TMC_31_TIMER), and the status of the respective TMC from TMC Fault Status (TMC_FAULT_STATUS).

72.4.2.3 Block diagram

The following figure is the top-level block diagram of the TMC array. Individual TMC_FAULT*n* signals merge to form the TMC_FAULT bus. All TMC instances share the TMC_CLOCK and TMC_CONFIG_ENABLE signals.

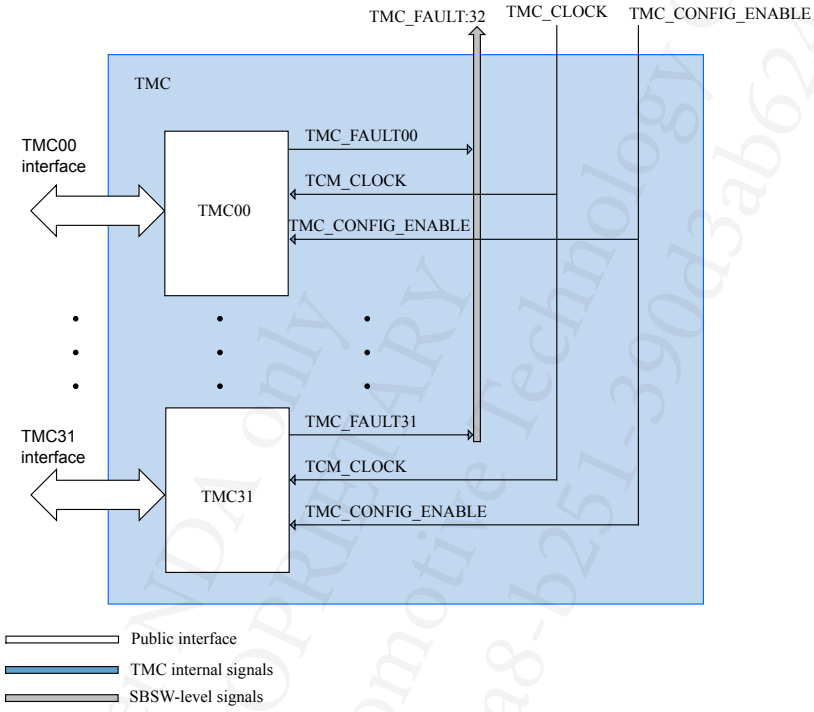


Figure 571. TMC block diagram

72.4.2.4 TMC states

You check the state of each TMC by using the corresponding TMC_a_STATUS register. Following are the possible states of a TMC:

- Timing: Indicates that the timer is currently running
- Idle: Indicates that the TMC is in an intermediate state—between enabling a TMC, and writing into R0 and R1
- R0 Written: Indicates that R0 is written
- R1 Written: Indicates that R1 is written
- Comparing: Indicates that both R0 and R1 were written and TMC is currently performing a comparison
- Fault: Indicates that TMC has detected a fault

There are four register fields determining the state of running of a TMC (see TMC Status (TMC_0_STATUS - TMC_31_STATUS)):

- R0_WRITTEN
- R1_WRITTEN
- FAULT

- OVERWRITTEN

The mapping between the TMC states and the flags are shown in the table below:

Table 567. TMC state mapping

Flags			States	
R0_Written	R1_Written	Fault	In Watchdog mode	In Window mode
0	0	00	Timing	Idle
1	0	00	R0 Written	R0 Written
0	1	00	R1 Written	R1 Written
1	1	00	Comparing	Comparing
0	0	01	Fault	Fault

72.4.2.5 TMC events

The TMC reports a fault on the following events:

- A false comparison fault is detected if the comparison result is not as configured.
- A timeout fault occurs if:
  - Both R0 and R1 are not written within the pre-configured time interval.
  - One of either R0 or R1 is rewritten before the other one is written.

A TMC reports a fault by asserting the TMC_FAULT*n* signal. These fault signals are aggregated by the module into a common status vector and presented to the SBSW controller. The controller is responsible for asserting the NCF signal to FCCU (via the TMC_FAULT signal). The TMC_FAULT*n* signal asserts the appropriate field in TMC_FAULT_STATUS[STATUSn] and stops the respective TMC timer. To learn whether the fault that a TMC instance reports is a false comparison or a timeout fault, see the respective field in TMC_a_STATUS[FAULT].

If the timer value is not configured properly (for example, if you specify a too small value because of design constraints), the timeout fault occurs almost immediately.

Use the following registers to analyze the details of a timeout:

- TMC_a_TIMEOUT[TIMEOUT] – To know the timeout value specified for the timer.
- TMC_a_TIMER[TIMER] – To know the current value of the timer and to calculate the remaining count before the expiration.

To clear the fault output:

- Disable the corresponding TMC instance.
- Clear the TMC_a_STATUS[FAULT] field which restarts the corresponding TMC instance. In this case, the comparator behaves exactly as if it were disabled and enabled immediately without any change in its configuration.

Clearing the fault output also eliminates contribution of the comparator to the TMC_FAULT signal. The TMC_FAULT NCF ceases when none of the TMCs reports any fault.

72.4.3 TMWDPI

TMWDPI is a sub-module of SBSW. It includes a set of registers that acts as a communication interface between the application software and the TMWDP. For details on the register interface, see TMWDPI registers.

The function of TMWDP is to monitor the execution of the application. You program a TMWDPI register to request for an application's execution progress state, and the TMWDP verifies the correctness of the execution progress. TMWDP also monitors that the application is reporting its execution progress in a timely fashion. For details on TMWDP, see TMWDP description.

72.4.3.1 TMWDP description

The TMWDP monitors the program flow using 32 timed finite state automata—timed state machines, and each state of an automaton is guarded by a specified time interval. The TMWDP ensures that a valid transition must take place within this time. A valid transition is a change in execution state of an application that matches the program flow configured in the respective automaton.

This algorithm for the TMWDP operation is implemented in firmware provided by NXP as a part of the SBSW software support package. The firmware runs on a dedicated core within the chip—TMWDP Core—hereafter referred to as the TC. The firmware is either loaded from the system memory to a local code RAM or executed directly from the system memory. In the latter case, the TMWDP firmware eventually ends up in the CPU cache. It completely fits typical code cache sizes.

Preliminary memory size estimates are as follows:

- 8 KB firmware
- 64 KB SRAM used for runtime data and a runtime copy of the TMWDP configuration

A TMWDP automaton reports the following faults:

- Time violation: TMWDP reports a time violation fault after either of the following two events:
  - If a valid transition does not complete within the time specified
  - If the application does not report change in its state for more than a specified time

  Each state of an automaton is guarded by a specified time interval.

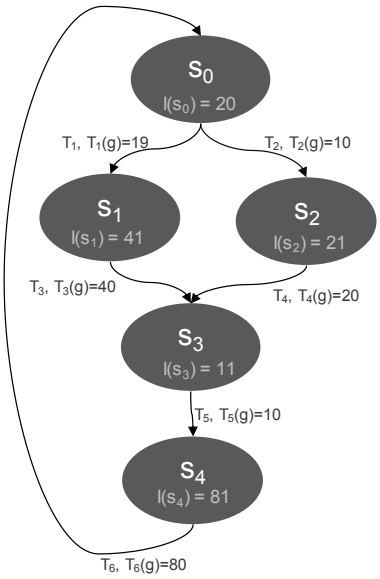
  A TMWDP automaton reports this fault by asserting the TMWDP_FAULT*n* signal.
- Illegal transition: TMWDP reports an illegal transition fault if the application reports its transition to a state that does not immediately follow the current state. This is because the application program flow must match the flow in each automaton.

  A TMWDP reports this fault by asserting the TMWDP_FAULT signal.
- Internal error: TMWDP reports an internal error if the automaton could not process its algorithm timely.

  A TMWDP reports this fault by asserting the TMWDP_INTERNAL_FAULT signal.

The following figure is an example of the automaton program flow.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b051-590d3a624c1



Legend:
S_n: Application states
$I(S_n)$: Maximum time before which the application must change its state
$T_n(g)$: Maximum time before which the given transition must occur

Figure 572. Timed automaton example

You statically generate the configuration for the TMWDP automata based on a particular application structure, stored in the application memory. You also pass the address of this application structure to the algorithm through the TMWDP_CONFIG_ADDR register. To prevent race conditions and common-cause faults during runtime, the TMWDP firmware copies the configuration data from the system memory to the local RAM during initialization.

A TMWDP timer, which is a hardware timer within SBSW, sends an interrupt (TMWDP_TIMER_IRQ signal) that executes the TMWDP algorithm periodically on TC. Only a single transition request is allowed per TMWDP timer period between the TMWDP algorithm runs.

72.4.3.2 Block diagram

The following figure is the top-level block diagram of TMWDP.

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-8255-9013ab624c1

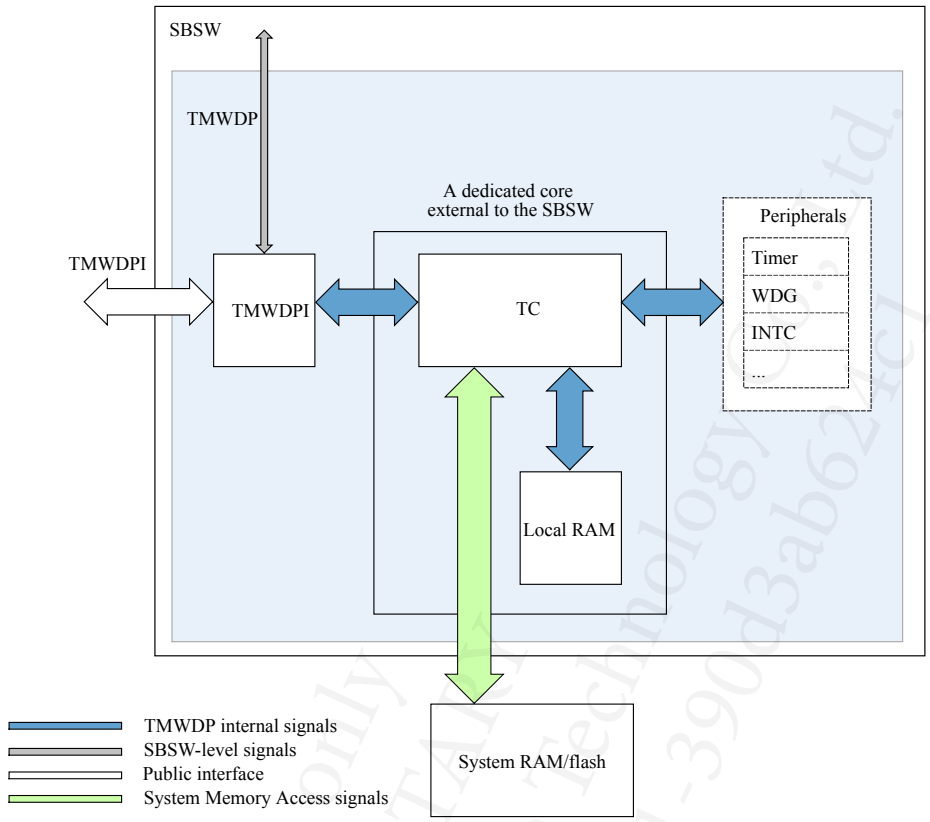


Figure 573. TMWDP block diagram

72.4.3.3 TMWDPI registers

The TMWDPI provides status indications concerning both TMWDP as a whole and each individual automaton.

The TMWDPI registers support the TMWDP algorithm to achieve a deterministic process of progress reporting. The register interface provides the following functionality:

- The application writes its state ID to the respective automaton register and the TMWDPI sets the written flag to indicate the automaton to process the request.
- The TMWDPI sets the overwritten flag if an application performs a new progress request while the written flag is set. Also, it does not write the new state ID to the respective automaton register.
- The TC reads the state ID and the flags from the progress request registers, and this also clears both the flags (written and overwritten).
- The TMWDPI performs arbitration when concurrent accesses are made by both the AC and TC. In general, write and read operations from both the cores are atomic.

The TMWDPI consists of the following register sets:

- **Configuration registers:**

  These registers are used for loading automata configuration and for enabling or disabling the entire TMWDP. These registers have write access privileges by the ACs, and are considered as input registers for the TMWDP. Write access to these registers is protected by the TMWDP_CONFIG_ENABLE signal from the SBSW controller. See Unlocking/locking the TMWDP configuration for details.

- **Status registers:**

  These registers report the status of the TMWDP as a whole.

- **Automaton registers:**

  These are the communication registers for the TMWDP. Each automaton has its own register for reporting progress of its corresponding application. Each write request to an automaton register is recorded by a write/overwrite detection logic. The TMWDP sets the written flag when an application performs a progress request, and it sets the overwritten flag when a new progress state is requested while the written flag is set. A TC read clears both the flags.

The configuration and status registers are common to the entire TMWDP sub-module.

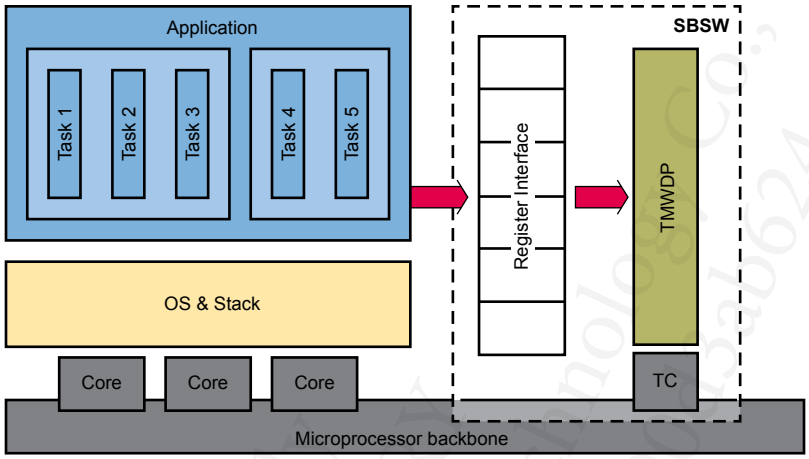


Figure 574. TMWDP register interface

All TMWDPI registers have two independent access ports (sides):

- The application side: accessed by the AC
- The internal side: manipulated by the dedicated TC exclusively

The access privileges are maintained by the SBSW controller.

72.4.3.4 TMWDP/TMWDPI events

The TMWDPI and the TMWDP algorithm report the following events:

- Invalid configuration data (invalid address, failed initialization or runtime consistency check)
- Internal issues (watchdog timeout, excessive runtime)
- Invalid transition request (wrong destination state ID, invalid automaton ID, more than one transition request per automaton in a TMWDP cycle)
- Timeout of an automaton

According to the status of individual automata, the TMWDPI maintains the corresponding bits of TMWDP_AUTOMATA_STATUS, TMWDP_AUTOMATA_ILLGL_TRANS, and TMWDP_AUTOMATA_TIME_VIOLATION as well as the status bits of each automaton application register. It also routes the aggregated fault vector to the SBSW controller.

When an automaton detects a fault or a timeout, it stops (AUT_a_STATUS[STATUS]=0). You need to reset the automaton to clear the error bits by requesting a transition to state 0 if the automaton is stopped.

72.4.4 SBSW interface summary

The SBSW exposes to the application:

- 32 instances (an array) of TMC with their own interface:
  - A configuration register and multiple status registers

  - Data registers R0 and R1
  - Timeout and timer registers
- Aggregated TMC fault register (32-bit wide)
- 32 state transition registers in the TMWDPI to mitigate a race condition bottleneck among AC and the TC executing the TMWDP firmware
- Aggregated TMWDP status and fault registers (32-bit wide; register width equals number of automata)
- A TMWDP configuration register
- A TMWDP status register
- Configuration locking mechanism

The SBSW module reports the faults detected by it to the FCCU through dedicated NCF lines.

72.5 Memory map and register definition

72.5.1 SBSW register descriptions

Only word (32-bit) accesses are allowed on all registers, and these registers can be read at any time.

Access to offset address 1Ch in the SBSW memory space does not result in a transfer error.

72.5.1.1 SBSW memory map

SBSW base address: 4031_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	TMC Configuration Unlock (TMC_CONFIG_UNLOCK)	32	WO	0000_0000h
4h	TMWDP Configuration Unlock (TMWDP_CONFIG_UNLOCK)	32	WO	0000_0000h
8h	TMC Configuration Status (TMC_CONFIG_STATUS)	32	RO	0000_0000h
Ch	TMWDP Configuration Status (TMWDP_CONFIG_STATUS)	32	RO	0000_0000h
10h	Debug Mode (DEBUG_MODE)	32	RW	0000_0000h
14h	TMC Fault Status (TMC_FAULT_STATUS)	32	RO	0000_0000h
18h	TMWDP Fault Status (TMWDP_FAULT_STATUS)	32	RO	0000_0000h
20h	TMC Configuration (TMC_0_CONFIG)	32	RW	0000_0000h
24h	TMC Distance (TMC_0_DISTANCE)	32	RW	0000_0000h
28h	TMC Timeout (TMC_0_TIMEOUT)	32	RW	0000_0000h
2Ch	TMC Control (TMC_0_CONTROL)	32	RW	0000_0000h
30h	TMC R0 Data (TMC_0_R0)	32	RW	0000_0000h
34h	TMC R1 Data (TMC_0_R1)	32	RW	0000_0000h
38h	TMC Status (TMC_0_STATUS)	32	W1C	0000_0000h
3Ch	TMC Timer (TMC_0_TIMER)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
40h	TMC Configuration (TMC_1_CONFIG)	32	RW	0000_0000h
44h	TMC Distance (TMC_1_DISTANCE)	32	RW	0000_0000h
48h	TMC Timeout (TMC_1_TIMEOUT)	32	RW	0000_0000h
4Ch	TMC Control (TMC_1_CONTROL)	32	RW	0000_0000h
50h	TMC R0 Data (TMC_1_R0)	32	RW	0000_0000h
54h	TMC R1 Data (TMC_1_R1)	32	RW	0000_0000h
58h	TMC Status (TMC_1_STATUS)	32	W1C	0000_0000h
5Ch	TMC Timer (TMC_1_TIMER)	32	RO	0000_0000h
60h	TMC Configuration (TMC_2_CONFIG)	32	RW	0000_0000h
64h	TMC Distance (TMC_2_DISTANCE)	32	RW	0000_0000h
68h	TMC Timeout (TMC_2_TIMEOUT)	32	RW	0000_0000h
6Ch	TMC Control (TMC_2_CONTROL)	32	RW	0000_0000h
70h	TMC R0 Data (TMC_2_R0)	32	RW	0000_0000h
74h	TMC R1 Data (TMC_2_R1)	32	RW	0000_0000h
78h	TMC Status (TMC_2_STATUS)	32	W1C	0000_0000h
7Ch	TMC Timer (TMC_2_TIMER)	32	RO	0000_0000h
80h	TMC Configuration (TMC_3_CONFIG)	32	RW	0000_0000h
84h	TMC Distance (TMC_3_DISTANCE)	32	RW	0000_0000h
88h	TMC Timeout (TMC_3_TIMEOUT)	32	RW	0000_0000h
8Ch	TMC Control (TMC_3_CONTROL)	32	RW	0000_0000h
90h	TMC R0 Data (TMC_3_R0)	32	RW	0000_0000h
94h	TMC R1 Data (TMC_3_R1)	32	RW	0000_0000h
98h	TMC Status (TMC_3_STATUS)	32	W1C	0000_0000h
9Ch	TMC Timer (TMC_3_TIMER)	32	RO	0000_0000h
A0h	TMC Configuration (TMC_4_CONFIG)	32	RW	0000_0000h
A4h	TMC Distance (TMC_4_DISTANCE)	32	RW	0000_0000h
A8h	TMC Timeout (TMC_4_TIMEOUT)	32	RW	0000_0000h
ACh	TMC Control (TMC_4_CONTROL)	32	RW	0000_0000h
B0h	TMC R0 Data (TMC_4_R0)	32	RW	0000_0000h
B4h	TMC R1 Data (TMC_4_R1)	32	RW	0000_0000h
B8h	TMC Status (TMC_4_STATUS)	32	W1C	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
BCh	TMC Timer (TMC_4_TIMER)	32	RO	0000_0000h
C0h	TMC Configuration (TMC_5_CONFIG)	32	RW	0000_0000h
C4h	TMC Distance (TMC_5_DISTANCE)	32	RW	0000_0000h
C8h	TMC Timeout (TMC_5_TIMEOUT)	32	RW	0000_0000h
CCh	TMC Control (TMC_5_CONTROL)	32	RW	0000_0000h
D0h	TMC R0 Data (TMC_5_R0)	32	RW	0000_0000h
D4h	TMC R1 Data (TMC_5_R1)	32	RW	0000_0000h
D8h	TMC Status (TMC_5_STATUS)	32	W1C	0000_0000h
DCh	TMC Timer (TMC_5_TIMER)	32	RO	0000_0000h
E0h	TMC Configuration (TMC_6_CONFIG)	32	RW	0000_0000h
E4h	TMC Distance (TMC_6_DISTANCE)	32	RW	0000_0000h
E8h	TMC Timeout (TMC_6_TIMEOUT)	32	RW	0000_0000h
ECh	TMC Control (TMC_6_CONTROL)	32	RW	0000_0000h
F0h	TMC R0 Data (TMC_6_R0)	32	RW	0000_0000h
F4h	TMC R1 Data (TMC_6_R1)	32	RW	0000_0000h
F8h	TMC Status (TMC_6_STATUS)	32	W1C	0000_0000h
FCh	TMC Timer (TMC_6_TIMER)	32	RO	0000_0000h
100h	TMC Configuration (TMC_7_CONFIG)	32	RW	0000_0000h
104h	TMC Distance (TMC_7_DISTANCE)	32	RW	0000_0000h
108h	TMC Timeout (TMC_7_TIMEOUT)	32	RW	0000_0000h
10Ch	TMC Control (TMC_7_CONTROL)	32	RW	0000_0000h
110h	TMC R0 Data (TMC_7_R0)	32	RW	0000_0000h
114h	TMC R1 Data (TMC_7_R1)	32	RW	0000_0000h
118h	TMC Status (TMC_7_STATUS)	32	W1C	0000_0000h
11Ch	TMC Timer (TMC_7_TIMER)	32	RO	0000_0000h
120h	TMC Configuration (TMC_8_CONFIG)	32	RW	0000_0000h
124h	TMC Distance (TMC_8_DISTANCE)	32	RW	0000_0000h
128h	TMC Timeout (TMC_8_TIMEOUT)	32	RW	0000_0000h
12Ch	TMC Control (TMC_8_CONTROL)	32	RW	0000_0000h
130h	TMC R0 Data (TMC_8_R0)	32	RW	0000_0000h
134h	TMC R1 Data (TMC_8_R1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
138h	TMC Status (TMC_8_STATUS)	32	W1C	0000_0000h
13Ch	TMC Timer (TMC_8_TIMER)	32	RO	0000_0000h
140h	TMC Configuration (TMC_9_CONFIG)	32	RW	0000_0000h
144h	TMC Distance (TMC_9_DISTANCE)	32	RW	0000_0000h
148h	TMC Timeout (TMC_9_TIMEOUT)	32	RW	0000_0000h
14Ch	TMC Control (TMC_9_CONTROL)	32	RW	0000_0000h
150h	TMC R0 Data (TMC_9_R0)	32	RW	0000_0000h
154h	TMC R1 Data (TMC_9_R1)	32	RW	0000_0000h
158h	TMC Status (TMC_9_STATUS)	32	W1C	0000_0000h
15Ch	TMC Timer (TMC_9_TIMER)	32	RO	0000_0000h
160h	TMC Configuration (TMC_10_CONFIG)	32	RW	0000_0000h
164h	TMC Distance (TMC_10_DISTANCE)	32	RW	0000_0000h
168h	TMC Timeout (TMC_10_TIMEOUT)	32	RW	0000_0000h
16Ch	TMC Control (TMC_10_CONTROL)	32	RW	0000_0000h
170h	TMC R0 Data (TMC_10_R0)	32	RW	0000_0000h
174h	TMC R1 Data (TMC_10_R1)	32	RW	0000_0000h
178h	TMC Status (TMC_10_STATUS)	32	W1C	0000_0000h
17Ch	TMC Timer (TMC_10_TIMER)	32	RO	0000_0000h
180h	TMC Configuration (TMC_11_CONFIG)	32	RW	0000_0000h
184h	TMC Distance (TMC_11_DISTANCE)	32	RW	0000_0000h
188h	TMC Timeout (TMC_11_TIMEOUT)	32	RW	0000_0000h
18Ch	TMC Control (TMC_11_CONTROL)	32	RW	0000_0000h
190h	TMC R0 Data (TMC_11_R0)	32	RW	0000_0000h
194h	TMC R1 Data (TMC_11_R1)	32	RW	0000_0000h
198h	TMC Status (TMC_11_STATUS)	32	W1C	0000_0000h
19Ch	TMC Timer (TMC_11_TIMER)	32	RO	0000_0000h
1A0h	TMC Configuration (TMC_12_CONFIG)	32	RW	0000_0000h
1A4h	TMC Distance (TMC_12_DISTANCE)	32	RW	0000_0000h
1A8h	TMC Timeout (TMC_12_TIMEOUT)	32	RW	0000_0000h
1ACh	TMC Control (TMC_12_CONTROL)	32	RW	0000_0000h
1B0h	TMC R0 Data (TMC_12_R0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1B4h	TMC R1 Data (TMC_12_R1)	32	RW	0000_0000h
1B8h	TMC Status (TMC_12_STATUS)	32	W1C	0000_0000h
1BCh	TMC Timer (TMC_12_TIMER)	32	RO	0000_0000h
1C0h	TMC Configuration (TMC_13_CONFIG)	32	RW	0000_0000h
1C4h	TMC Distance (TMC_13_DISTANCE)	32	RW	0000_0000h
1C8h	TMC Timeout (TMC_13_TIMEOUT)	32	RW	0000_0000h
1CCh	TMC Control (TMC_13_CONTROL)	32	RW	0000_0000h
1D0h	TMC R0 Data (TMC_13_R0)	32	RW	0000_0000h
1D4h	TMC R1 Data (TMC_13_R1)	32	RW	0000_0000h
1D8h	TMC Status (TMC_13_STATUS)	32	W1C	0000_0000h
1DCh	TMC Timer (TMC_13_TIMER)	32	RO	0000_0000h
1E0h	TMC Configuration (TMC_14_CONFIG)	32	RW	0000_0000h
1E4h	TMC Distance (TMC_14_DISTANCE)	32	RW	0000_0000h
1E8h	TMC Timeout (TMC_14_TIMEOUT)	32	RW	0000_0000h
1ECh	TMC Control (TMC_14_CONTROL)	32	RW	0000_0000h
1F0h	TMC R0 Data (TMC_14_R0)	32	RW	0000_0000h
1F4h	TMC R1 Data (TMC_14_R1)	32	RW	0000_0000h
1F8h	TMC Status (TMC_14_STATUS)	32	W1C	0000_0000h
1FCh	TMC Timer (TMC_14_TIMER)	32	RO	0000_0000h
200h	TMC Configuration (TMC_15_CONFIG)	32	RW	0000_0000h
204h	TMC Distance (TMC_15_DISTANCE)	32	RW	0000_0000h
208h	TMC Timeout (TMC_15_TIMEOUT)	32	RW	0000_0000h
20Ch	TMC Control (TMC_15_CONTROL)	32	RW	0000_0000h
210h	TMC R0 Data (TMC_15_R0)	32	RW	0000_0000h
214h	TMC R1 Data (TMC_15_R1)	32	RW	0000_0000h
218h	TMC Status (TMC_15_STATUS)	32	W1C	0000_0000h
21Ch	TMC Timer (TMC_15_TIMER)	32	RO	0000_0000h
220h	TMC Configuration (TMC_16_CONFIG)	32	RW	0000_0000h
224h	TMC Distance (TMC_16_DISTANCE)	32	RW	0000_0000h
228h	TMC Timeout (TMC_16_TIMEOUT)	32	RW	0000_0000h
22Ch	TMC Control (TMC_16_CONTROL)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
230h	TMC R0 Data (TMC_16_R0)	32	RW	0000_0000h
234h	TMC R1 Data (TMC_16_R1)	32	RW	0000_0000h
238h	TMC Status (TMC_16_STATUS)	32	W1C	0000_0000h
23Ch	TMC Timer (TMC_16_TIMER)	32	RO	0000_0000h
240h	TMC Configuration (TMC_17_CONFIG)	32	RW	0000_0000h
244h	TMC Distance (TMC_17_DISTANCE)	32	RW	0000_0000h
248h	TMC Timeout (TMC_17_TIMEOUT)	32	RW	0000_0000h
24Ch	TMC Control (TMC_17_CONTROL)	32	RW	0000_0000h
250h	TMC R0 Data (TMC_17_R0)	32	RW	0000_0000h
254h	TMC R1 Data (TMC_17_R1)	32	RW	0000_0000h
258h	TMC Status (TMC_17_STATUS)	32	W1C	0000_0000h
25Ch	TMC Timer (TMC_17_TIMER)	32	RO	0000_0000h
260h	TMC Configuration (TMC_18_CONFIG)	32	RW	0000_0000h
264h	TMC Distance (TMC_18_DISTANCE)	32	RW	0000_0000h
268h	TMC Timeout (TMC_18_TIMEOUT)	32	RW	0000_0000h
26Ch	TMC Control (TMC_18_CONTROL)	32	RW	0000_0000h
270h	TMC R0 Data (TMC_18_R0)	32	RW	0000_0000h
274h	TMC R1 Data (TMC_18_R1)	32	RW	0000_0000h
278h	TMC Status (TMC_18_STATUS)	32	W1C	0000_0000h
27Ch	TMC Timer (TMC_18_TIMER)	32	RO	0000_0000h
280h	TMC Configuration (TMC_19_CONFIG)	32	RW	0000_0000h
284h	TMC Distance (TMC_19_DISTANCE)	32	RW	0000_0000h
288h	TMC Timeout (TMC_19_TIMEOUT)	32	RW	0000_0000h
28Ch	TMC Control (TMC_19_CONTROL)	32	RW	0000_0000h
290h	TMC R0 Data (TMC_19_R0)	32	RW	0000_0000h
294h	TMC R1 Data (TMC_19_R1)	32	RW	0000_0000h
298h	TMC Status (TMC_19_STATUS)	32	W1C	0000_0000h
29Ch	TMC Timer (TMC_19_TIMER)	32	RO	0000_0000h
2A0h	TMC Configuration (TMC_20_CONFIG)	32	RW	0000_0000h
2A4h	TMC Distance (TMC_20_DISTANCE)	32	RW	0000_0000h
2A8h	TMC Timeout (TMC_20_TIMEOUT)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2ACh	TMC Control (TMC_20_CONTROL)	32	RW	0000_0000h
2B0h	TMC R0 Data (TMC_20_R0)	32	RW	0000_0000h
2B4h	TMC R1 Data (TMC_20_R1)	32	RW	0000_0000h
2B8h	TMC Status (TMC_20_STATUS)	32	W1C	0000_0000h
2BCh	TMC Timer (TMC_20_TIMER)	32	RO	0000_0000h
2C0h	TMC Configuration (TMC_21_CONFIG)	32	RW	0000_0000h
2C4h	TMC Distance (TMC_21_DISTANCE)	32	RW	0000_0000h
2C8h	TMC Timeout (TMC_21_TIMEOUT)	32	RW	0000_0000h
2CCh	TMC Control (TMC_21_CONTROL)	32	RW	0000_0000h
2D0h	TMC R0 Data (TMC_21_R0)	32	RW	0000_0000h
2D4h	TMC R1 Data (TMC_21_R1)	32	RW	0000_0000h
2D8h	TMC Status (TMC_21_STATUS)	32	W1C	0000_0000h
2DCh	TMC Timer (TMC_21_TIMER)	32	RO	0000_0000h
2E0h	TMC Configuration (TMC_22_CONFIG)	32	RW	0000_0000h
2E4h	TMC Distance (TMC_22_DISTANCE)	32	RW	0000_0000h
2E8h	TMC Timeout (TMC_22_TIMEOUT)	32	RW	0000_0000h
2ECh	TMC Control (TMC_22_CONTROL)	32	RW	0000_0000h
2F0h	TMC R0 Data (TMC_22_R0)	32	RW	0000_0000h
2F4h	TMC R1 Data (TMC_22_R1)	32	RW	0000_0000h
2F8h	TMC Status (TMC_22_STATUS)	32	W1C	0000_0000h
2FCh	TMC Timer (TMC_22_TIMER)	32	RO	0000_0000h
300h	TMC Configuration (TMC_23_CONFIG)	32	RW	0000_0000h
304h	TMC Distance (TMC_23_DISTANCE)	32	RW	0000_0000h
308h	TMC Timeout (TMC_23_TIMEOUT)	32	RW	0000_0000h
30Ch	TMC Control (TMC_23_CONTROL)	32	RW	0000_0000h
310h	TMC R0 Data (TMC_23_R0)	32	RW	0000_0000h
314h	TMC R1 Data (TMC_23_R1)	32	RW	0000_0000h
318h	TMC Status (TMC_23_STATUS)	32	W1C	0000_0000h
31Ch	TMC Timer (TMC_23_TIMER)	32	RO	0000_0000h
320h	TMC Configuration (TMC_24_CONFIG)	32	RW	0000_0000h
324h	TMC Distance (TMC_24_DISTANCE)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
328h	TMC Timeout (TMC_24_TIMEOUT)	32	RW	0000_0000h
32Ch	TMC Control (TMC_24_CONTROL)	32	RW	0000_0000h
330h	TMC R0 Data (TMC_24_R0)	32	RW	0000_0000h
334h	TMC R1 Data (TMC_24_R1)	32	RW	0000_0000h
338h	TMC Status (TMC_24_STATUS)	32	W1C	0000_0000h
33Ch	TMC Timer (TMC_24_TIMER)	32	RO	0000_0000h
340h	TMC Configuration (TMC_25_CONFIG)	32	RW	0000_0000h
344h	TMC Distance (TMC_25_DISTANCE)	32	RW	0000_0000h
348h	TMC Timeout (TMC_25_TIMEOUT)	32	RW	0000_0000h
34Ch	TMC Control (TMC_25_CONTROL)	32	RW	0000_0000h
350h	TMC R0 Data (TMC_25_R0)	32	RW	0000_0000h
354h	TMC R1 Data (TMC_25_R1)	32	RW	0000_0000h
358h	TMC Status (TMC_25_STATUS)	32	W1C	0000_0000h
35Ch	TMC Timer (TMC_25_TIMER)	32	RO	0000_0000h
360h	TMC Configuration (TMC_26_CONFIG)	32	RW	0000_0000h
364h	TMC Distance (TMC_26_DISTANCE)	32	RW	0000_0000h
368h	TMC Timeout (TMC_26_TIMEOUT)	32	RW	0000_0000h
36Ch	TMC Control (TMC_26_CONTROL)	32	RW	0000_0000h
370h	TMC R0 Data (TMC_26_R0)	32	RW	0000_0000h
374h	TMC R1 Data (TMC_26_R1)	32	RW	0000_0000h
378h	TMC Status (TMC_26_STATUS)	32	W1C	0000_0000h
37Ch	TMC Timer (TMC_26_TIMER)	32	RO	0000_0000h
380h	TMC Configuration (TMC_27_CONFIG)	32	RW	0000_0000h
384h	TMC Distance (TMC_27_DISTANCE)	32	RW	0000_0000h
388h	TMC Timeout (TMC_27_TIMEOUT)	32	RW	0000_0000h
38Ch	TMC Control (TMC_27_CONTROL)	32	RW	0000_0000h
390h	TMC R0 Data (TMC_27_R0)	32	RW	0000_0000h
394h	TMC R1 Data (TMC_27_R1)	32	RW	0000_0000h
398h	TMC Status (TMC_27_STATUS)	32	W1C	0000_0000h
39Ch	TMC Timer (TMC_27_TIMER)	32	RO	0000_0000h
3A0h	TMC Configuration (TMC_28_CONFIG)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
3A4h	TMC Distance (TMC_28_DISTANCE)	32	RW	0000_0000h
3A8h	TMC Timeout (TMC_28_TIMEOUT)	32	RW	0000_0000h
3ACh	TMC Control (TMC_28_CONTROL)	32	RW	0000_0000h
3B0h	TMC R0 Data (TMC_28_R0)	32	RW	0000_0000h
3B4h	TMC R1 Data (TMC_28_R1)	32	RW	0000_0000h
3B8h	TMC Status (TMC_28_STATUS)	32	W1C	0000_0000h
3BCh	TMC Timer (TMC_28_TIMER)	32	RO	0000_0000h
3C0h	TMC Configuration (TMC_29_CONFIG)	32	RW	0000_0000h
3C4h	TMC Distance (TMC_29_DISTANCE)	32	RW	0000_0000h
3C8h	TMC Timeout (TMC_29_TIMEOUT)	32	RW	0000_0000h
3CCh	TMC Control (TMC_29_CONTROL)	32	RW	0000_0000h
3D0h	TMC R0 Data (TMC_29_R0)	32	RW	0000_0000h
3D4h	TMC R1 Data (TMC_29_R1)	32	RW	0000_0000h
3D8h	TMC Status (TMC_29_STATUS)	32	W1C	0000_0000h
3DCh	TMC Timer (TMC_29_TIMER)	32	RO	0000_0000h
3E0h	TMC Configuration (TMC_30_CONFIG)	32	RW	0000_0000h
3E4h	TMC Distance (TMC_30_DISTANCE)	32	RW	0000_0000h
3E8h	TMC Timeout (TMC_30_TIMEOUT)	32	RW	0000_0000h
3ECh	TMC Control (TMC_30_CONTROL)	32	RW	0000_0000h
3F0h	TMC R0 Data (TMC_30_R0)	32	RW	0000_0000h
3F4h	TMC R1 Data (TMC_30_R1)	32	RW	0000_0000h
3F8h	TMC Status (TMC_30_STATUS)	32	W1C	0000_0000h
3FCh	TMC Timer (TMC_30_TIMER)	32	RO	0000_0000h
400h	TMC Configuration (TMC_31_CONFIG)	32	RW	0000_0000h
404h	TMC Distance (TMC_31_DISTANCE)	32	RW	0000_0000h
408h	TMC Timeout (TMC_31_TIMEOUT)	32	RW	0000_0000h
40Ch	TMC Control (TMC_31_CONTROL)	32	RW	0000_0000h
410h	TMC R0 Data (TMC_31_R0)	32	RW	0000_0000h
414h	TMC R1 Data (TMC_31_R1)	32	RW	0000_0000h
418h	TMC Status (TMC_31_STATUS)	32	W1C	0000_0000h
41Ch	TMC Timer (TMC_31_TIMER)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
420h	TMWDP Configuration Address (TMWDP_CONFIG_ADDR)	32	RW	0000_0000h
424h	TMWDP Control (TMWDP_CONTROL)	32	RW	0000_0000h
428h	TMWDP Status (TMWDP_STATUS)	32	RO	0000_0000h
42Ch	TMWDP Automata Status (TMWDP_AUTOMATA_STATUS)	32	RO	0000_0000h
430h	TMWDP Automata Illegal Transition (TMWDP_AUTOMATA_ILLGL_TRANS)	32	RO	0000_0000h
434h	TMWDP Automata Time Violation (TMWDP_AUTOMATA_TIME_VIOLATION)	32	RO	0000_0000h
450h	TMWDP Automaton Status (AUT_0_STATUS)	32	RO	0000_0000h
454h	TMWDP Automaton Progress Request (AUT_0_PRGS_REQ)	32	RW	0000_0000h
458h	TMWDP Automaton Status (AUT_1_STATUS)	32	RO	0000_0000h
45Ch	TMWDP Automaton Progress Request (AUT_1_PRGS_REQ)	32	RW	0000_0000h
460h	TMWDP Automaton Status (AUT_2_STATUS)	32	RO	0000_0000h
464h	TMWDP Automaton Progress Request (AUT_2_PRGS_REQ)	32	RW	0000_0000h
468h	TMWDP Automaton Status (AUT_3_STATUS)	32	RO	0000_0000h
46Ch	TMWDP Automaton Progress Request (AUT_3_PRGS_REQ)	32	RW	0000_0000h
470h	TMWDP Automaton Status (AUT_4_STATUS)	32	RO	0000_0000h
474h	TMWDP Automaton Progress Request (AUT_4_PRGS_REQ)	32	RW	0000_0000h
478h	TMWDP Automaton Status (AUT_5_STATUS)	32	RO	0000_0000h
47Ch	TMWDP Automaton Progress Request (AUT_5_PRGS_REQ)	32	RW	0000_0000h
480h	TMWDP Automaton Status (AUT_6_STATUS)	32	RO	0000_0000h
484h	TMWDP Automaton Progress Request (AUT_6_PRGS_REQ)	32	RW	0000_0000h
488h	TMWDP Automaton Status (AUT_7_STATUS)	32	RO	0000_0000h
48Ch	TMWDP Automaton Progress Request (AUT_7_PRGS_REQ)	32	RW	0000_0000h
490h	TMWDP Automaton Status (AUT_8_STATUS)	32	RO	0000_0000h
494h	TMWDP Automaton Progress Request (AUT_8_PRGS_REQ)	32	RW	0000_0000h
498h	TMWDP Automaton Status (AUT_9_STATUS)	32	RO	0000_0000h
49Ch	TMWDP Automaton Progress Request (AUT_9_PRGS_REQ)	32	RW	0000_0000h
4A0h	TMWDP Automaton Status (AUT_10_STATUS)	32	RO	0000_0000h
4A4h	TMWDP Automaton Progress Request (AUT_10_PRGS_REQ)	32	RW	0000_0000h
4A8h	TMWDP Automaton Status (AUT_11_STATUS)	32	RO	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4ACh	TMWDP Automaton Progress Request (AUT_11_PRGS_REQ)	32	RW	0000_0000h
4B0h	TMWDP Automaton Status (AUT_12_STATUS)	32	RO	0000_0000h
4B4h	TMWDP Automaton Progress Request (AUT_12_PRGS_REQ)	32	RW	0000_0000h
4B8h	TMWDP Automaton Status (AUT_13_STATUS)	32	RO	0000_0000h
4BCh	TMWDP Automaton Progress Request (AUT_13_PRGS_REQ)	32	RW	0000_0000h
4C0h	TMWDP Automaton Status (AUT_14_STATUS)	32	RO	0000_0000h
4C4h	TMWDP Automaton Progress Request (AUT_14_PRGS_REQ)	32	RW	0000_0000h
4C8h	TMWDP Automaton Status (AUT_15_STATUS)	32	RO	0000_0000h
4CCh	TMWDP Automaton Progress Request (AUT_15_PRGS_REQ)	32	RW	0000_0000h
4D0h	TMWDP Automaton Status (AUT_16_STATUS)	32	RO	0000_0000h
4D4h	TMWDP Automaton Progress Request (AUT_16_PRGS_REQ)	32	RW	0000_0000h
4D8h	TMWDP Automaton Status (AUT_17_STATUS)	32	RO	0000_0000h
4DCh	TMWDP Automaton Progress Request (AUT_17_PRGS_REQ)	32	RW	0000_0000h
4E0h	TMWDP Automaton Status (AUT_18_STATUS)	32	RO	0000_0000h
4E4h	TMWDP Automaton Progress Request (AUT_18_PRGS_REQ)	32	RW	0000_0000h
4E8h	TMWDP Automaton Status (AUT_19_STATUS)	32	RO	0000_0000h
4ECh	TMWDP Automaton Progress Request (AUT_19_PRGS_REQ)	32	RW	0000_0000h
4F0h	TMWDP Automaton Status (AUT_20_STATUS)	32	RO	0000_0000h
4F4h	TMWDP Automaton Progress Request (AUT_20_PRGS_REQ)	32	RW	0000_0000h
4F8h	TMWDP Automaton Status (AUT_21_STATUS)	32	RO	0000_0000h
4FCh	TMWDP Automaton Progress Request (AUT_21_PRGS_REQ)	32	RW	0000_0000h
500h	TMWDP Automaton Status (AUT_22_STATUS)	32	RO	0000_0000h
504h	TMWDP Automaton Progress Request (AUT_22_PRGS_REQ)	32	RW	0000_0000h
508h	TMWDP Automaton Status (AUT_23_STATUS)	32	RO	0000_0000h
50Ch	TMWDP Automaton Progress Request (AUT_23_PRGS_REQ)	32	RW	0000_0000h
510h	TMWDP Automaton Status (AUT_24_STATUS)	32	RO	0000_0000h
514h	TMWDP Automaton Progress Request (AUT_24_PRGS_REQ)	32	RW	0000_0000h
518h	TMWDP Automaton Status (AUT_25_STATUS)	32	RO	0000_0000h
51Ch	TMWDP Automaton Progress Request (AUT_25_PRGS_REQ)	32	RW	0000_0000h
520h	TMWDP Automaton Status (AUT_26_STATUS)	32	RO	0000_0000h
524h	TMWDP Automaton Progress Request (AUT_26_PRGS_REQ)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
528h	TMWDP Automaton Status (AUT_27_STATUS)	32	RO	0000_0000h
52Ch	TMWDP Automaton Progress Request (AUT_27_PRGS_REQ)	32	RW	0000_0000h
530h	TMWDP Automaton Status (AUT_28_STATUS)	32	RO	0000_0000h
534h	TMWDP Automaton Progress Request (AUT_28_PRGS_REQ)	32	RW	0000_0000h
538h	TMWDP Automaton Status (AUT_29_STATUS)	32	RO	0000_0000h
53Ch	TMWDP Automaton Progress Request (AUT_29_PRGS_REQ)	32	RW	0000_0000h
540h	TMWDP Automaton Status (AUT_30_STATUS)	32	RO	0000_0000h
544h	TMWDP Automaton Progress Request (AUT_30_PRGS_REQ)	32	RW	0000_0000h
548h	TMWDP Automaton Status (AUT_31_STATUS)	32	RO	0000_0000h
54Ch	TMWDP Automaton Progress Request (AUT_31_PRGS_REQ)	32	RW	0000_0000h
FFCh	TMWDP Core Domain ID (TC_ID)	32	RW	0000_0000h

72.5.1.2 TMC Configuration Unlock (TMC_CONFIG_UNLOCK)

Offset

Register	Offset
TMC_CONFIG_UNLOCK	0h

Function

This register is used for locking and unlocking the TMC configuration. See Unlocking/locking the TMC configuration for the unlocking and locking procedures.

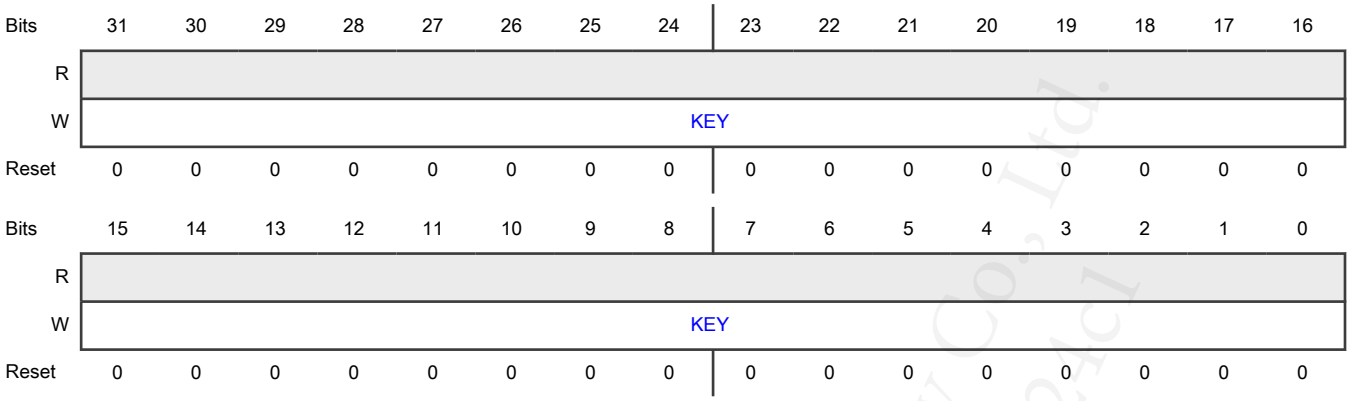
TMC configuration registers stay locked (that is, any write access is not permitted) by default or after functional reset (assertion of IPG_HARD_ASYNC_RESET_B signal).

NOTE

This register temporarily holds the key value and its bit-inverted value.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CXU Automotive Technology Co.,Ltd.
c8c462e5-4921-48a8-b251-3900abb6a1a1

Diagram



Fields

Field	Function
31-0 KEY	Key Write the key value (2718_2818h) and its inverted value (D8E7_D7E7h) in sequence within the pre-defined time interval (62 SAFETY_IP_CLK cycles) to unlock the TMC configuration. Writing any other value or a timeout (that is, writing the second value after 62 SAFETY_IP_CLK cycles) aborts the TMC configuration unlocking procedure.

72.5.1.3 TMWDP Configuration Unlock (TMWDP_CONFIG_UNLOCK)

Offset

Register	Offset
TMWDP_CONFIG_UNLOCK	4h

Function

This register is used for locking and unlocking the TMWDP configuration. See Unlocking/locking the TMWDP configuration for the unlocking and locking procedures.

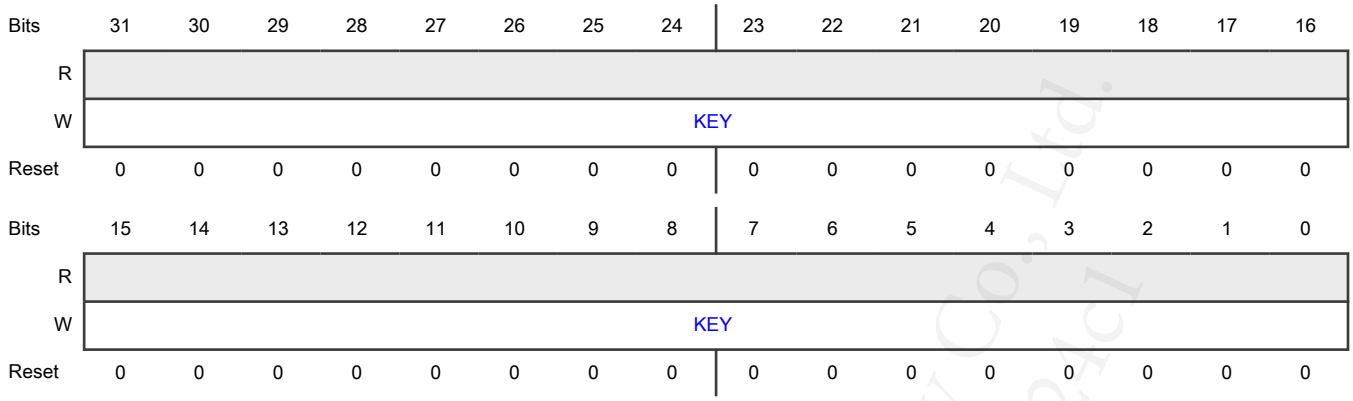
TMWDP configuration registers stay locked (that is, any write access is not permitted) by default or after functional reset (assertion of IPG_HARD_ASYNC_RESET_B signal).

NOTE

This register temporarily holds the key value and its bit-inverted value.

Provided under NDA of COMPANY PROPRIETARY Beijing CIU Automotive Technology Co., Ltd. c8c4626-4821-48a8-b251-3d8ab624c1

Diagram



Fields

Field	Function
31-0 KEY	Key Write the key value (3141_5926h) and its inverted value (CEBE_A6D9h) in sequence within the pre-defined time interval (62 SAFETY_IP_CLK cycles) to unlock the TMWDP configuration. Writing any other value or a timeout (that is, writing the second value after 62 SAFETY_IP_CLK cycles) aborts the TMWDP configuration unlocking procedure.

72.5.1.4 TMC Configuration Status (TMC_CONFIG_STATUS)

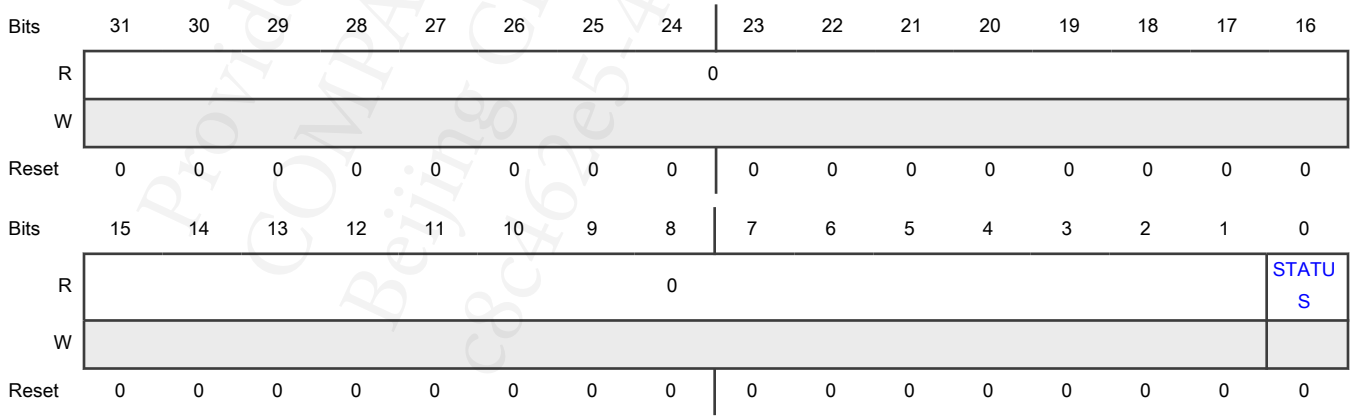
Offset

Register	Offset
TMC_CONFIG_STATUS	8h

Function

This register indicates the current status (locked or unlocked) of TMC configuration. You can configure the TMC only if the configuration is in the unlocked state. See Unlocking/locking the TMC configuration for the unlocking procedure.

Diagram



Provided under NDA of COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-482-48a8-b251-3d8ab624c1

Fields

Field	Function
31-1 —	Reserved
0 STATUS	Status This bit indicates the state of TMC configuration. 0b - Locked 1b - Unlocked

72.5.1.5 TMWDP Configuration Status (TMWDP_CONFIG_STATUS)

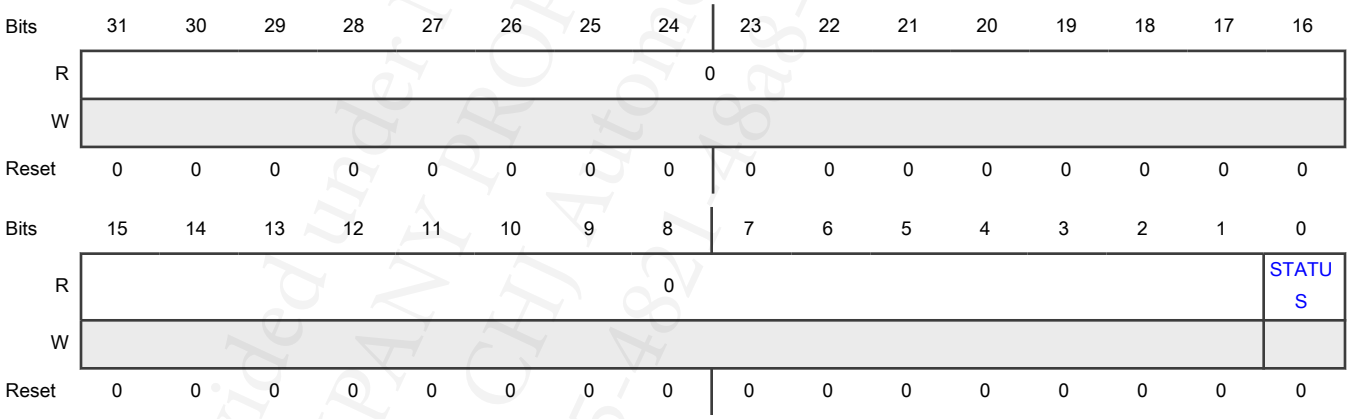
Offset

Register	Offset
TMWDP_CONFIG_STATUS	Ch

Function

This register indicates the current status (locked or unlocked) of TMWDP configuration. You can configure the TMWDP only if the configuration is in the unlocked state. See Unlocking/locking the TMWDP configuration for the unlocking procedure.

Diagram



Fields

Field	Function
31-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 STATUS	Status This bit indicates the state of TMWDP configuration. 0b - Locked 1b - Unlocked

72.5.1.6 Debug Mode (DEBUG_MODE)

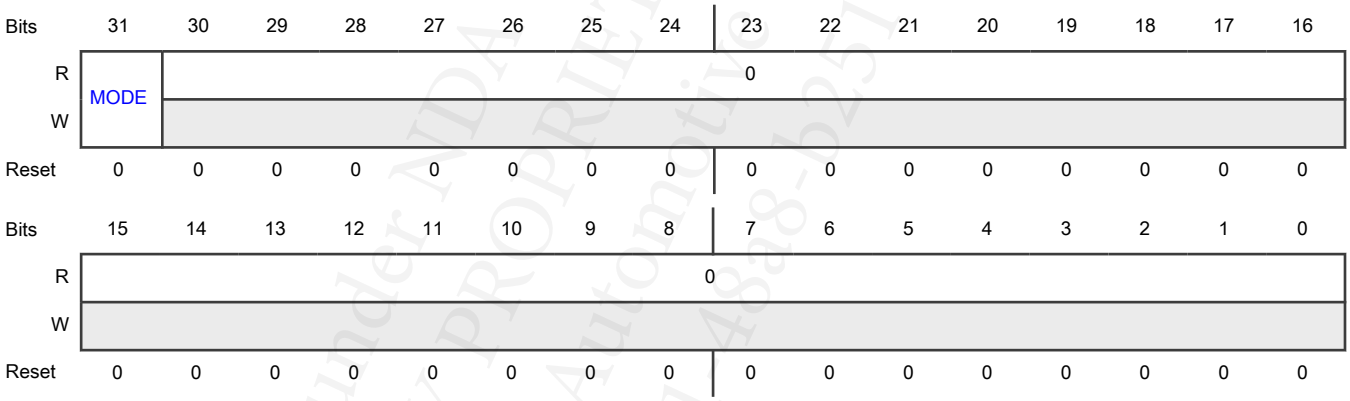
Offset

Register	Offset
DEBUG_MODE	10h

Function

This register is used for configuring the behavior of the SBSW when the chip is in the Debug mode.

Diagram



Fields

Field	Function
31 MODE	Mode 0b - Debug mode does not affect the SBSW behavior. 1b - All counters/timers within the SBSW stop in the Debug mode. The list of counters comprises of two unlocking timers in the controller, one timer in each TMC instance, and the other timer in the TMWDPI.
30-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4205-5482-1482-b251-390a3ab624c1

72.5.1.7 TMC Fault Status (TMC_FAULT_STATUS)

Offset

Register	Offset
TMC_FAULT_STATUS	14h

Function

See TMC_FAULT_STATUS[STATUSn].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATUS31	STATUS30	STATUS29	STATUS28	STATUS27	STATUS26	STATUS25	STATUS24	STATUS23	STATUS22	STATUS21	STATUS20	STATUS19	STATUS18	STATUS17	STATUS16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STATUS15	STATUS14	STATUS13	STATUS12	STATUS11	STATUS10	STATUS9	STATUS8	STATUS7	STATUS6	STATUS5	STATUS4	STATUS3	STATUS2	STATUS1	STATUS0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STATUSn	Status n This field indicates whether the corresponding TMC instance (n) has detected a fault. A TMC reports a fault by asserting the TMC_FAULT*n* signal. 0b - Fault is not detected. 1b - Fault is detected.

72.5.1.8 TMWDP Fault Status (TMWDP_FAULT_STATUS)

Offset

Register	Offset
TMWDP_FAULT_STATUS	18h

Function

See TMWDP_FAULT_STATUS[STATUSn].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATUS31	STATUS30	STATUS29	STATUS28	STATUS27	STATUS26	STATUS25	STATUS24	STATUS23	STATUS22	STATUS21	STATUS20	STATUS19	STATUS18	STATUS17	STATUS16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STATUS15	STATUS14	STATUS13	STATUS12	STATUS11	STATUS10	STATUS9	STATUS8	STATUS7	STATUS6	STATUS5	STATUS4	STATUS3	STATUS2	STATUS1	STATUS0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STATUSn	Status n This field indicates whether the corresponding TMWDP automaton instance (n) has detected a fault. A TMWDP automaton reports a fault by asserting the TMWDP_FAULT*n* signal. 0b - Fault is not detected. 1b - Fault is detected.

72.5.1.9 TMC Configuration (TMC_0_CONFIG - TMC_31_CONFIG)

Offset

For a = 0 to 31:

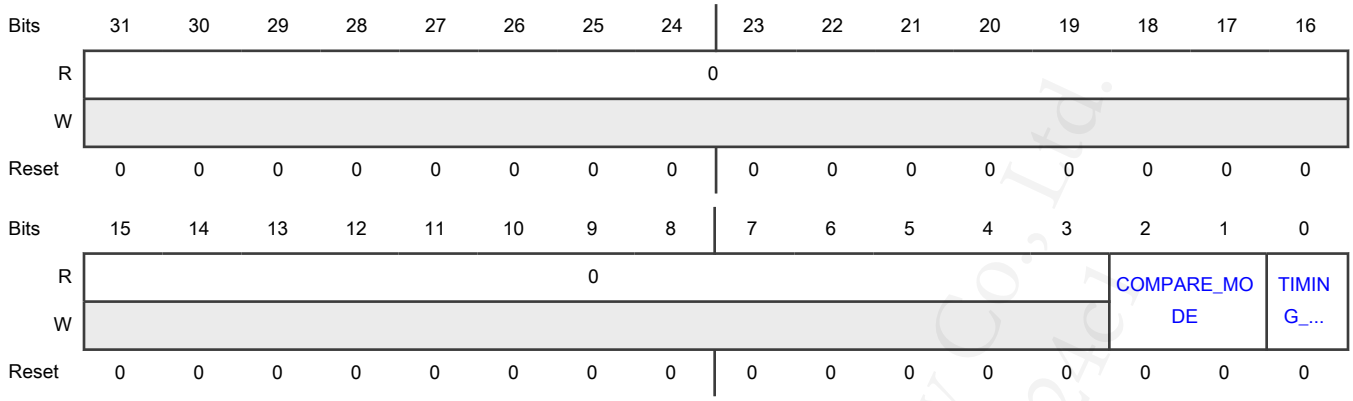
Register	Offset
TMC_a_CONFIG	20h + (a × 20h)

Function

This register is used for configuring the TMC timing and the comparison modes.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-30d3ab624

Diagram



Fields

Field	Function
31-3 —	Reserved
2-1 COMPARE_MODE	Compare mode Select one of the following modes. **NOTE** You program the value for Distance (D) in TMC_a_DISTANCE[DISTANCE]. 00b - (R0 == R1) 01b - (R0 < R1) 10b - (R0 > R1) 11b - (|R1 – R0| ≤ D)
0 TIMING_MODE	Timing mode Select one of the following modes. 0b - Watchdog Mode 1b - Window Mode

72.5.1.10 TMC Distance (TMC_0_DISTANCE - TMC_31_DISTANCE)

Offset

For a = 0 to 31:

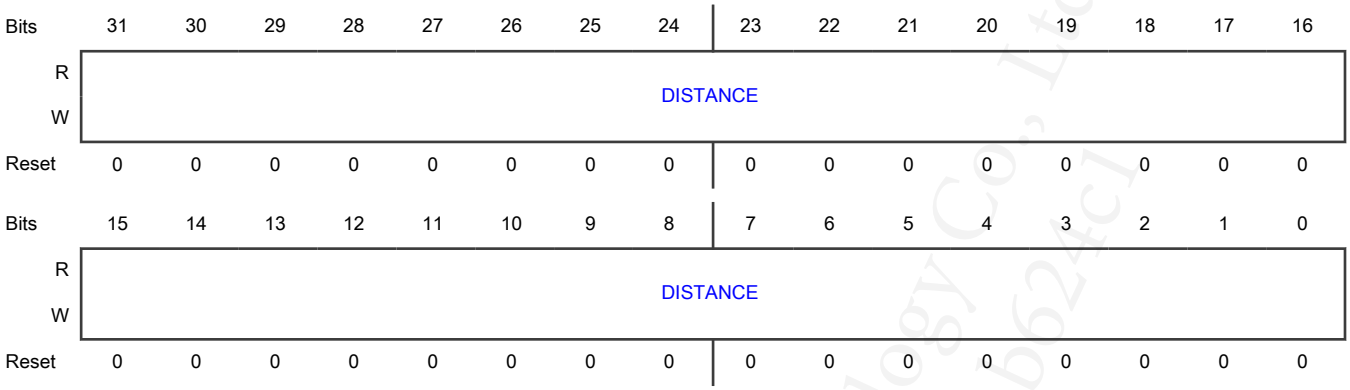
Register	Offset
TMC_a_DISTANCE	24h + (a × 20h)

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab62402

Function

See TMC_a_DISTANCE[DISTANCE].

Diagram



Fields

Field	Function
31-0 DISTANCE	Distance value You configure the maximum distance between the R0 and R1 values for comparison. **NOTE** Distance is a 32-bit unsigned value.

72.5.1.11 TMC Timeout (TMC_0_TIMEOUT - TMC_31_TIMEOUT)

Offset

For a = 0 to 31:

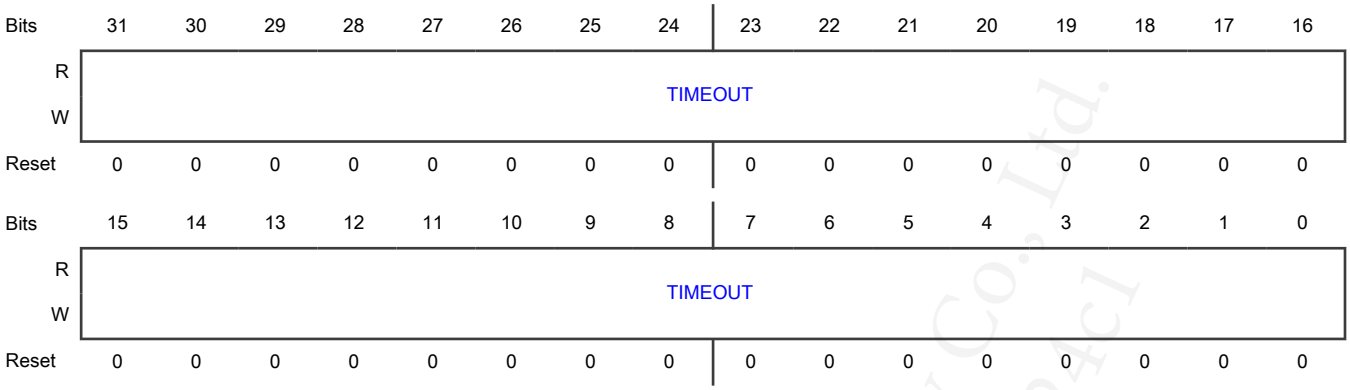
Register	Offset
TMC_a_TIMEOUT	28h + (a × 20h)

Function

See TMC_a_TIMEOUT[TIMEOUT].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b521-390d3ab624c1

Diagram



Fields

Field	Function
31-0 TIMEOUT	Timeout interval Functioning of this field varies with the timer modes. You select the timer mode using the TMC_a_CONFIG[TIMING_MODE] field. In Watchdog mode, you use this field to configure the timeout interval in which the timer periodically compares the values written to the R0 and R1 registers . In Window mode, you use this field to configure the timeout interval within which the R0 and R1 registers must be written. **NOTE** Each unit represents one tick of SAFETY_IP_CLK cycle (15 ns). See Timer for more information about the timer modes.

72.5.1.12 TMC Control (TMC_0_CONTROL - TMC_31_CONTROL)

Offset

For a = 0 to 31:

Register	Offset
TMC_a_CONTROL	2Ch + (a × 20h)

Function

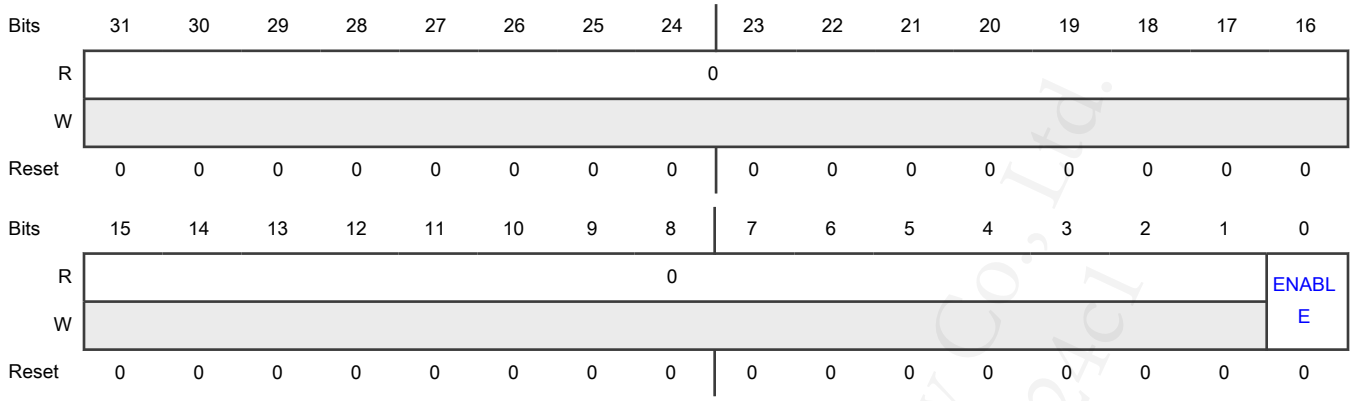
This register is used for enabling and disabling the corresponding TMC instance. For proper functioning of the TMC, enable the TMC_a_CONTROL[ENABLE] field only after configuring the corresponding TMC (by using TMC_a_CONFIG, TMC_a_DISTANCE and TMC_a_TIMEOUT).

NOTE

Clear the corresponding TMC_a_STATUS[FAULT] field (by writing 11b to it) before enabling the TMC instance.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
ca46e5-4821-4da8-8251-590d3ab624c1

Diagram



Fields

Field	Function
31-1 —	Reserved
0 ENABLE	Enable 0b - TMC is disabled. 1b - TMC is enabled.

72.5.1.13 TMC R0 Data (TMC_0_R0 - TMC_31_R0)

Offset

For a = 0 to 31:

Register	Offset
TMC_a_R0	30h + (a × 20h)

Function

See TMC_a_R0[R0].

NOTE

This is a 32-bit signed register (2's complement).

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	R0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 R0	R0 data value Write the TMC's R0 data value for comparison.

72.5.1.14 TMC R1 Data (TMC_0_R1 - TMC_31_R1)

Offset

For a = 0 to 31:

Register	Offset
TMC_a_R1	34h + (a × 20h)

Function

See TMC_a_R1[R1].

NOTE

This is a 32-bit signed register (2's complement).

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	R1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	R1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e54-82f4-48a8-b251-390d3ab624c1

Fields

Field	Function
31-0 R1	R1 data value Write the TMC's R1 data value for comparison.

72.5.1.15 TMC Status (TMC_0_STATUS - TMC_31_STATUS)

Offset

For a = 0 to 31:

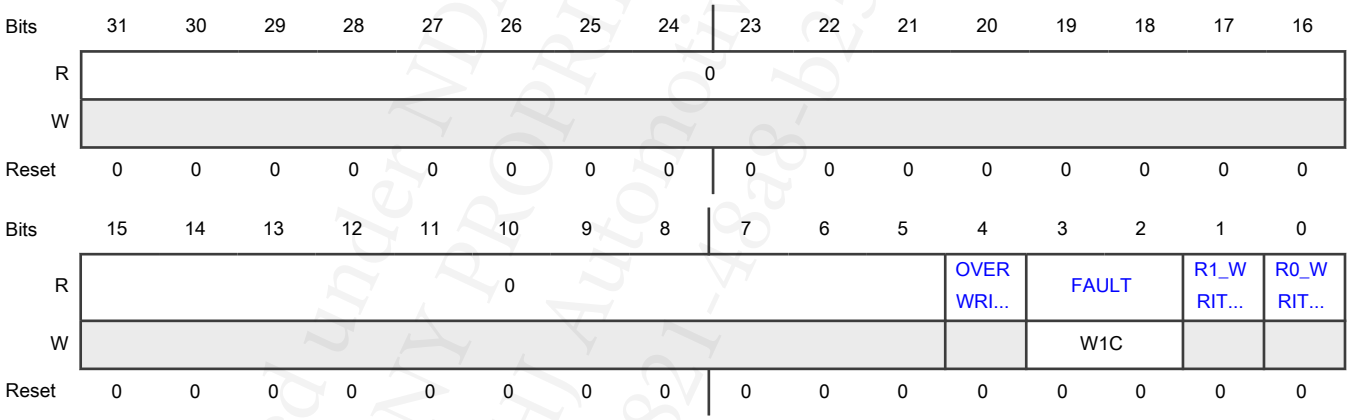
Register	Offset
TMC_a_STATUS	38h + (a × 20h)

Function

This register indicates the current state of the corresponding TMC.

When the second register is written and the comparison succeeds, the TMC immediately enters its initial state ("Timing" or "Idle"). Otherwise, it first transitions to "Comparing" state and then to "Fault" state with successive clock pulses. Thus, the TMC_a_STATUS[FAULT] field is updated one clock cycle after the last data write is complete.

Diagram



Fields

Field	Function
31-5 —	Reserved
4 OVERWRITTEN	Overwritten Indicates whether timeout has occurred because R0 or R1 was written more than one time during a single TMC cycle

Table continues on the next page...

Table continued from the previous page...

Field	Function
	You can also learn whether the timeout that has occurred is real (because of timer elapse) or not (because of R0/R1 overwritten) from the value of TMC_a_TIMER[TIMER]. **NOTE** You clear this field by clearing the TMC_a_STATUS[FAULT] field. 0b - Neither R0 nor R1 is overwritten. 1b - R0 or R1 is overwritten.
3-2 FAULT	Fault indicator Indicates the cause if the corresponding TMC has detected a fault. **NOTE** When you clear this field (by writing 11b), the other three fields in this register are also cleared (OVERWRITTEN, R0_WRITTEN, and R1_WRITTEN). 00b - No fault 01b - False comparison 10b - Timeout or overwritten R0/R1
1 R1_WRITTEN	R1 written Indicates whether R1 data value has been written to TMC_a_R1[R1] **NOTE** A successful comparison clears this field automatically. In case of a fault, you must clear TMC_a_STATUS[FAULT] to clear this field. 0b - R1 has not been written in the current cycle. 1b - TMC is in the Timing state.
0 R0_WRITTEN	R0 written Indicates whether R0 data value has been written to TMC_a_R0[R0] **NOTE** A successful comparison clears this field automatically. In case of a fault, you must clear TMC_a_STATUS[FAULT] to clear this field. 0b - R0 has not been written in the current cycle. 1b - TMC is in the Timing state.

72.5.1.16 TMC Timer (TMC_0_TIMER - TMC_31_TIMER)

Offset

For a = 0 to 31:

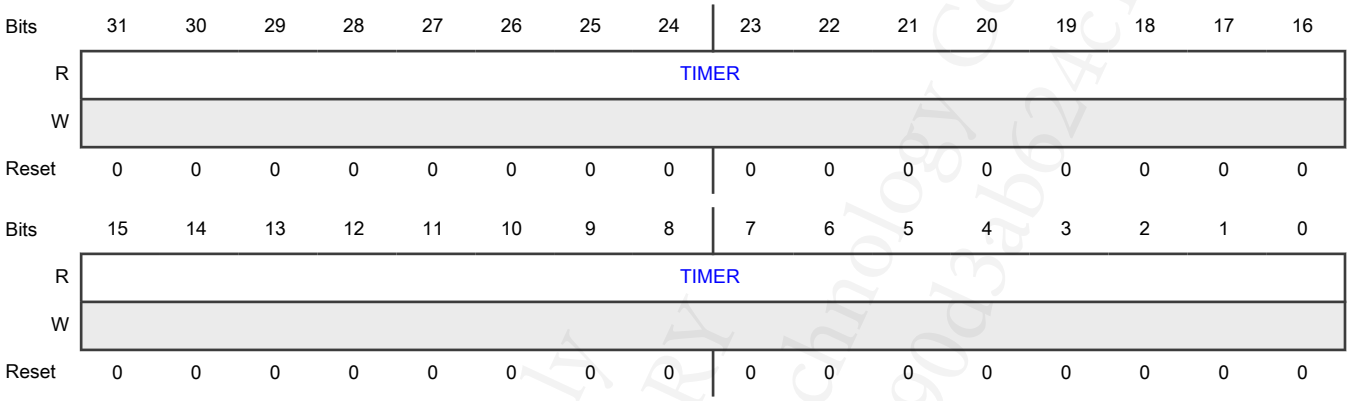
Provided under NDA only
COMPANY PROPRIETARY
Beijing CJ Automotive Technology Co., Ltd.
c8c42b5421-4848-8231-390d3ab624c1

Register	Offset
TMC_a_TIMER	3Ch + (a × 20h)

Function

See TMC_a_TIMER[TIMER].

Diagram



Fields

Field	Function
31-0 TIMER	Timer value Indicates the current timer value of the corresponding TMC

72.5.1.17 TMWDP Configuration Address (TMWDP_CONFIG_ADDR)

Offset

Register	Offset
TMWDP_CONFIG_ADDR	420h

Function

See TMWDP_CONFIG_ADDR[ADDRESS].

NOTE

This register is RW for the AC and RO for the TC.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e5-4821-4883-8251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	ADDRESS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	ADDRESS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 ADDRESS	Configuration address This field contains the address of the TMWDP configuration. You must unlock the configuration before writing to this field. See Unlocking/locking the TMWDP configuration for the unlocking procedure.

72.5.1.18 TMWDP Control (TMWDP_CONTROL)

Offset

Register	Offset
TMWDP_CONTROL	424h

Function

This register is for the applications to enable or disable the TMWDP. It is configured by the AC and read by the TC after the TMWDP configuration is complete (locked). You must unlock the TMWDP configuration before this register can be modified.

NOTE

This register is RW for the AC and RO for the TC.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															ENABLE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-1 —	Reserved
0 ENABLE	Enable 0b - TMWDP is disabled. 1b - TMWDP is enabled.

72.5.1.19 TMWDP Status (TMWDP_STATUS)

Offset

Register	Offset
TMWDP_STATUS	428h

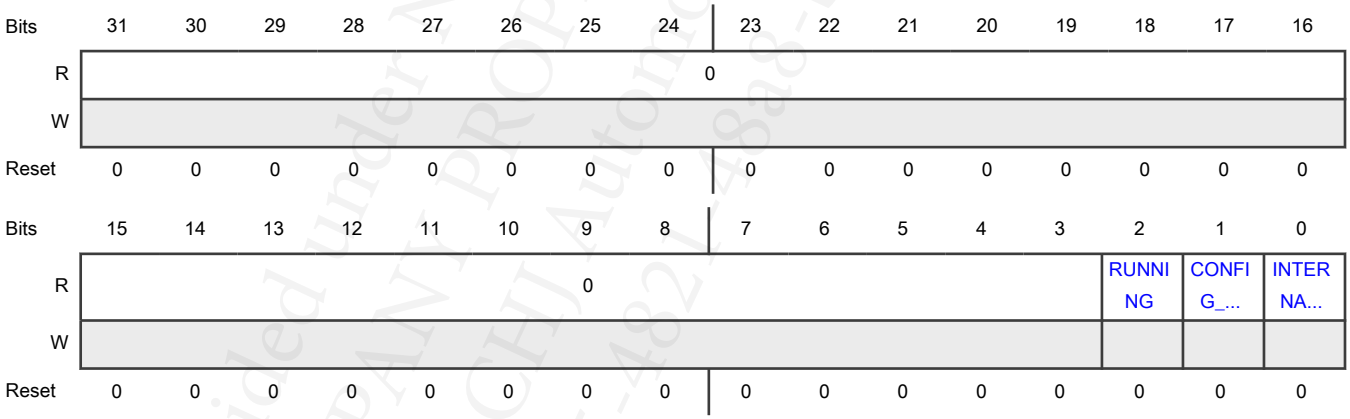
Function

This register indicates whether TMWDP is running or if a fault is detected.

NOTE

This register is RO for the AC and RW for the TC.

Diagram



Fields

Field	Function
31-3 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a6-b251-390a3ab624c1

Table continued from the previous page...

Field	Function
2 RUNNING	TMWDP running Indicates whether the TMWDP is running and ready to accept application requests 0b - TMWDP is stopped, or has encountered an error, or is processing a new configuration. 1b - TMWDP is running.
1 CONFIG_ERROR	Configuration error Indicates whether TMWDP has detected a configuration error, and triggers the TMWDP internal fault signal (NCF signal to FCCU: TMWDP_INTERNAL_FAULT) 0b - No TMWDP configuration error. 1b - TMWDP configuration error is detected.
0 INTERNAL_ERROR	TMWDP internal error Indicates whether TMWDP has detected an internal error, and triggers the TMWDP internal fault signal (NCF signal to FCCU: TMWDP_INTERNAL_FAULT) 0b - No TMWDP internal error. 1b - TMWDP internal error is detected.

72.5.1.20 TMWDP Automata Status (TMWDP_AUTOMATA_STATUS)

Offset

Register	Offset
TMWDP_AUTOMATA_STATUS	42Ch

Function

See TMWDP_AUTOMATA_STATUS[STATUSn].

NOTE

The TC cannot access this register.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIVI Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39013ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATUS31	STATUS30	STATUS29	STATUS28	STATUS27	STATUS26	STATUS25	STATUS24	STATUS23	STATUS22	STATUS21	STATUS20	STATUS19	STATUS18	STATUS17	STATUS16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STATUS15	STATUS14	STATUS13	STATUS12	STATUS11	STATUS10	STATUS9	STATUS8	STATUS7	STATUS6	STATUS5	STATUS4	STATUS3	STATUS2	STATUS1	STATUS0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STATUSn	Status n Indicates whether the corresponding TMWDP automaton instance (n) is running or has stopped 0b - Automaton has stopped (initial or fault state). 1b - Automaton is running.

72.5.1.21 TMWDP Automata Illegal Transition (TMWDP_AUTOMATA_ILLGL_TRANS)

Offset

Register	Offset
TMWDP_AUTOMATA_ILLGL_TRANS	430h

Function

See TMWDP_AUTOMATA_ILLGL_TRANS[STATUSn].

NOTE

The TC cannot access this register.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-1821-48a8-b251-390d3ab624

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATUS31	STATUS30	STATUS29	STATUS28	STATUS27	STATUS26	STATUS25	STATUS24	STATUS23	STATUS22	STATUS21	STATUS20	STATUS19	STATUS18	STATUS17	STATUS16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STATUS15	STATUS14	STATUS13	STATUS12	STATUS11	STATUS10	STATUS9	STATUS8	STATUS7	STATUS6	STATUS5	STATUS4	STATUS3	STATUS2	STATUS1	STATUS0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STATUSn	Status n Indicates whether the corresponding TMWDP automaton instance (n) has detected a faulty transition 0b - Automaton has not detected a faulty transition. 1b - Automaton has detected a faulty transition.

72.5.1.22 TMWDP Automata Time Violation (TMWDP_AUTOMATA_TIME_VIOLATION)

Offset

Register	Offset
TMWDP_AUTOMATA_TIME_VIOLATION	434h

Function

See TMWDP_AUTOMATA_TIME_VIOLATION[STATUSn].

NOTE

The TC cannot access this register.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-1821-48a8-b251-390d3ab624

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATUS31	STATUS30	STATUS29	STATUS28	STATUS27	STATUS26	STATUS25	STATUS24	STATUS23	STATUS22	STATUS21	STATUS20	STATUS19	STATUS18	STATUS17	STATUS16
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STATUS15	STATUS14	STATUS13	STATUS12	STATUS11	STATUS10	STATUS9	STATUS8	STATUS7	STATUS6	STATUS5	STATUS4	STATUS3	STATUS2	STATUS1	STATUS0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 STATUSn	Status n Indicates whether the corresponding TMWDP automaton instance (n) has detected a time violation 0b - Automaton has not detected a time violation. 1b - Automaton has detected a time violation.

72.5.1.23 TMWDP Automaton Status (AUT_0_STATUS - AUT_31_STATUS)

Offset

For a = 0 to 31:

Register	Offset
AUT_a_STATUS	450h + (a × 8h)

Function

This register indicates the status of the corresponding TMWDP automaton.

Following are the four types of status that it captures:

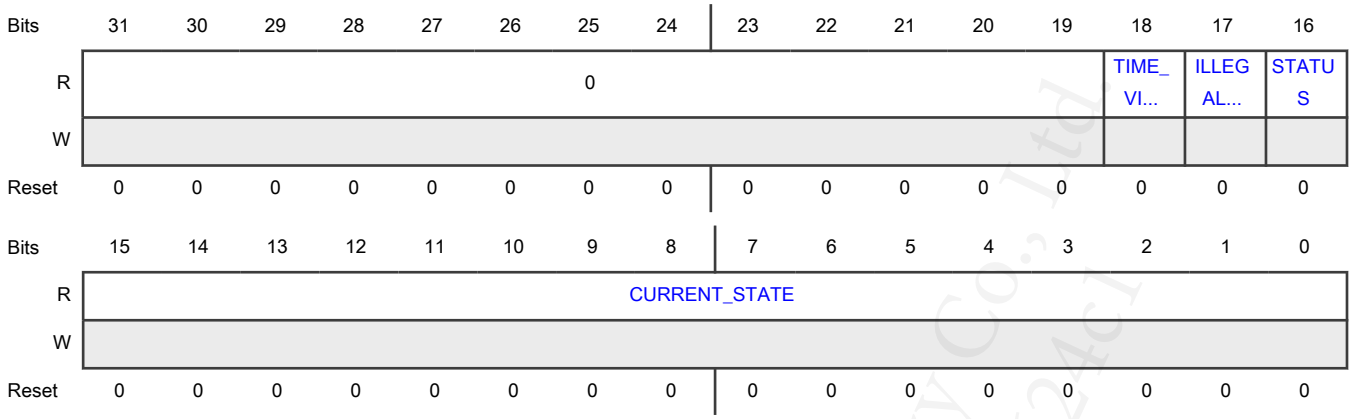
- Current state
- Running or stopped status
- Illegal transition fault status
- Time violation fault status

NOTE

This register is RO for the AC and RW for the TC.

Printed under NDA only
COMPANY PROPRIETARY
Beijing C-J Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-390d3ab624

Diagram



Fields

Field	Function
31-19 —	Reserved
18 TIME_VIOLATION	Time violation Indicates whether the corresponding automaton has detected a time violation 0b - Time violation is not detected. 1b - Time violation is detected.
17 ILLEGAL_TRANSITION	Illegal transition Indicates whether the corresponding automaton has detected an illegal transition 0b - Illegal transition is not detected. 1b - Illegal transition is detected.
16 STATUS	Automaton status Indicates whether the corresponding automaton is running or has stopped (initial or fault) 0b - Stopped (initial or fault state) 1b - Running state
15-0 CURRENT_STATE	Automaton current state Indicates the state ID of the corresponding automaton (any integer value from 0 to 65535).

72.5.1.24 TMWDP Automaton Progress Request (AUT_0_PRGS_REQ - AUT_31_PRGS_REQ)

Offset

For a = 0 to 31:

Provided to NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e_482_48a8_b251_390a3ab624c1

Register	Offset
AUT_a_PRGS_REQ	454h + (a × 8h)

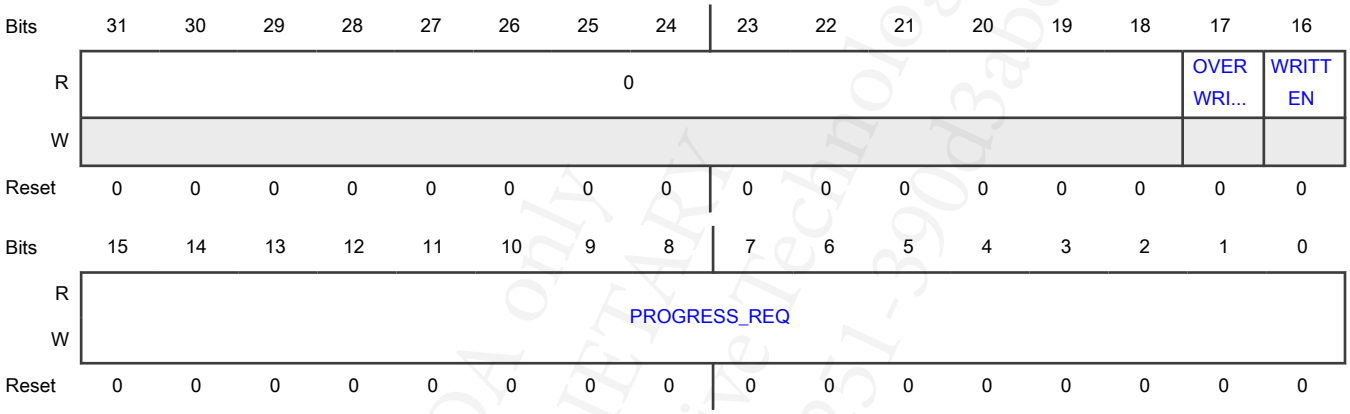
Function

This register is used as a communication channel between the application, which reports its progress by successively writing its state IDs, and the SBSW firmware running on TC, which reads and processes these requests.

NOTE

This register is RW for the AC and RO for the TC. When TC reads this register, the TMWDPI sets the WRITTEN and OVERWRITTEN fields to 0.

Diagram



Fields

Field	Function
31-18 —	Reserved
17 OVERWRITTEN	Overwritten Indicates whether the PROGRESS_REQ field has been written more than one time after the TMWDP last serviced this field **NOTE** When the TC reads this field, the TMWDPI sets this field to 0. 0b - PROGRESS_REQ field is not overwritten. 1b - PROGRESS_REQ field is overwritten.
16 WRITTEN	Written Indicates whether the PROGRESS_REQ field has been written after the TMWDP last serviced this field

Table continues on the next page...

Table continued from the previous page...

Field	Function
	NOTE When the TC reads this field, the TMWDPI sets this field to 0. 0b - PROGRESS_REQ field is not written. 1b - PROGRESS_REQ field is written.
15-0 PROGRESS_REQ	Application progress request You use this field to request an application to report its progress state.

72.5.1.25 TMWDP Core Domain ID (TC_ID)

Offset

Register	Offset
TC_ID	FFCh

Function

This register is used to set the TC domain ID. You can write to this register only one time after reset.

The TC boot initializes this register. You can read this register at any time. SBSW controller compares the domain ID value with the value on the sideband signal of the IPS interface, and the comparison result determines the core that performs the read or write operation. If the values match, the TC initiates the operation, otherwise the AC initiates the operation.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							WRITTEN	0				ID			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-9	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
8 WRITTEN	TC ID status Indicates whether you have written the TC domain ID to the TC_ID[ID] field 0b - Domain ID is not written. 1b - Domain ID is written.
7-4 —	Reserved
3-0 ID	TC domain ID value You set the value of the TC domain ID in this field. **NOTE** You must write to this field, even if the TC domain ID that you want to set is 0h.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 73
Error Injection Module (EIM)

73.1 Chip-specific EIM information

73.1.1 EIM instances

The following table summarizes the features of EIM instances on this chip.

Table 568. EIM instance summary

EIM instance	Number of used memory channels	Targets
ACP_EIM	31	Memory arrays for: • SRAM controllers • eDMA controllers • Cortex-M7 cores Comparison signals used by RCCU regarding lockstep operation of: • eDMA controllers • Cortex-M7 cores • Cortex-A53 cores • GIC-500
EIM_0	1	Memory arrays for: • Cortex-A53 cores • GIC-500
EIM_1	2	Memory arrays for QuadSPI
EIM_2	6	Memory arrays for SerDes subsystem
EIM_3	4	Memory arrays for DDR subsystem

73.1.2 ACP_EIM channel assignments

Each EIM channel corresponds to a specific target. The EIM instance can inject errors on access for two types of targets:

- Memory arrays: errors on accesses to the memory data bits and checkbits
- Comparison signals used by the RCCU: errors on data bits

It is recommended that the error injection configuration is enabled when the target is in quiescent state.

The following table shows details about the channel assignments and corresponding targets.

- In the EIM programming model, each channel has a set of EICHD*n*_WORD registers for injecting errors. This table correlates the WORD register bits with target bits.
- EIM channels 0–27 inject errors on memory arrays. Writing 1 to a WORD register bit causes the controller of the corresponding memory array to generate an error event.
- EIM channels 28–30 inject errors into comparison signals used by the RCCU. Writing 1 to any combination of these WORD bits triggers RCCU alarms.

— For the mapping of RCCU alarms to EIM channel bits, see Table 570.

— Each RCCU alarm corresponds to an FCCU fault.

Table 569. EICHD*n*_WORD register targets

Channel	Target	Data bits	Checkbits	Error injection path
0	SRAM controller 1 bank1–0	Bank1 bits 63–32: WORD1[31:0] Bank1 bits 31–0: WORD2[31:0]	Bank1 bits 7–0: WORD0[31:24]	Read
		Bank0 bits 63–32: WORD3[31:0] Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 7–0: WORD0[23:16]	Read
	SRAM Controller 0 bank1-0	Bank1 bits 63–32: WORD5[31:0] Bank1 bits 31–0: WORD6[31:0]	Bank1 bits 7–0: WORD0[15:8]	Read
		Bank0 bits 63–32: WORD7[31:0] Bank0 bits 31–0: WORD8[31:0]	Bank0 bits 7–0: WORD0[7:0]	Read
1	SRAM Controller 3 bank1-0	Bank1 bits 63–32: WORD1[31:0] Bank1 bits 31–0: WORD2[31:0]	Bank1 bits 7–0: WORD0[31:24]	Read
		Bank0 bits 63–32: WORD3[31:0] Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 7–0: WORD0[23:16]	Read
	SRAM Controller 2 bank1–0	Bank1 bits 63–32: WORD5[31:0] Bank1 bits 31–0: WORD6[31:0]	Bank1 bits 7–0: WORD0[15:8]	Read
		Bank0 bits 63–32: WORD7[31:0] Bank0 bits 31–0: WORD8[31:0]	Bank0 bits 7–0: WORD0[7:0]	Read
2	SRAM Controller 5 bank1-0	Bank1 bits 63–32: WORD1[31:0] Bank1 bits 31-0: WORD2[31:0]	Bank1 bits 7–0: WORD0[31:24]	Read
		Bank0 bits 63–32: WORD3[31:0] Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 7–0: WORD0[23:16]	Read
	SRAM Controller 4 bank1–0	Bank1 bits 63–32: WORD5[31:0] Bank1 bits 31–0: WORD6[31:0]	Bank1 bits 7–0: WORD0[15:8]	Read
		Bank0 bits 63–32: WORD7[31:0] Bank0 bits 31–0: WORD8[31:0]	Bank0 bits 7–0: WORD0[7:0]	Read
3	SRAM Controller 7 bank1–0	Bank1 bits 63–32: WORD1[31:0] Bank1 bits 31–0: WORD2[31:0]	Bank1 bits 7–0: WORD0[31:24]	Read

Table continues on the next page...

Table 569. EICHD*n*_WORD register targets (continued)

Channel	Target	Data bits	Checkbits	Error injection path
		Bank0 bits 63–32: WORD3[31:0] Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 7–0: WORD0[23:16]	Read
	SRAM Controller 6 bank1–0	Bank1 bits 63–32: WORD5[31:0] Bank1 bits 31–0: WORD6[31:0]	Bank1 bits 7–0: WORD0[15:8]	Read
		Bank0 bits 63–32: WORD7[31:0] Bank0 bits 31–0: WORD8[31:0]	Bank0 bits 7–0: WORD0[7:0]	Read
4	SRAM Controller 9 bank1–0	Bank1 bits 63–32: WORD1[31:0] Bank1 bits 31–0: WORD2[31:0]	Bank1 bits 7–0: WORD0[31:24]	Read
		Bank0 bits 63–32: WORD3[31:0] Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 7–0: WORD0[23:16]	Read
	SRAM Controller 8 bank1–0	Bank1 bits 63–32: WORD5[31:0] Bank1 bits 31–0: WORD6[31:0]	Bank1 bits 7–0: WORD0[15:8]	Read
		Bank0 bits 63–32: WORD7[31:0] Bank0 bits 31–0: WORD8[31:0]	Bank0 bits 7–0: WORD0[7:0]	Read
5	SRAM Controller 11 bank1–0	Bank1 bits 63–32: WORD1[31:0] Bank1 bits 31–0: WORD2[31:0]	Bank1 bits 7–0: WORD0[31:24]	Read
		Bank0 bits 63–32: WORD3[31:0] Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 7–0: WORD0[23:16]	Read
	SRAM Controller 10 bank1–0	Bank1 bits 63–32: WORD5[31:0] Bank1 bits 31–0: WORD6[31:0]	Bank1 bits 7–0: WORD0[15:8]	Read
		Bank0 bits 63–32: WORD7[31:0] Bank0 bits 31–0: WORD8[31:0]	Bank0 bits 7–0: WORD0[7:0]	Read
6	SRAM Controller 13 bank1–0	Bank1 bits 63–32: WORD1[31:0] Bank1 bits 31–0: WORD2[31:0]	Bank1 bits 7–0: WORD0[31:24]	Read
		Bank0 bits 63–32: WORD3[31:0] Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 7–0: WORD0[23:16]	Read
	SRAM Controller 12 bank1–0	Bank1 bits 63–32: WORD5[31:0] Bank1 bits 31–0: WORD6[31:0]	Bank1 bits 7–0: WORD0[15:8]	Read

Table continues on the next page...

Table 569. EICHD*n*_WORD register targets (continued)

Channel	Target	Data bits	Checkbits	Error injection path
		Bank0 bits 63–32: WORD7[31:0] Bank0 bits 31–0: WORD8[31:0]	Bank0 bits 7–0: WORD0[7:0]	Read
7	SRAM Controller 15 bank1–0	Bank1 bits 63–32: WORD1[31:0] Bank1 bits 31–0: WORD2[31:0]	Bank1 bits 7–0: WORD0[31:24]	Read
		Bank0 bits 63–32: WORD3[31:0] Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 7–0: WORD0[23:16]	Read
	SRAM Controller 14 bank1–0	Bank1 bits 63–32: WORD5[31:0] Bank1 bits 31-0: WORD6[31:0]	Bank1 bits 7–0: WORD0[15:8]	Read
		Bank0 bits 63–32: WORD7[31:0] Bank0 bits 31–0: WORD8[31:0]	Bank0 bits 7–0: WORD0[7:0]	Read
8	eDMA_0 TCD	TCD bits 63–32: WORD1[31:0] TCD bits 31–0: WORD2[31:0]	TCD bits 7–0: WORD0[31:24]	Read
9	eDMA_1 TCD	TCD bits 63–32: WORD1[31:0] TCD bits 31–0: WORD2[31:0]	TCD bits 7–0: WORD0[31:24]	Read
10	Cortex-M7 core 0 DTCM	D1TCM bits 31–0: WORD1[31:0]	D1TCM bits 7–0: WORD0[31:24]	Read
		D0TCM bits 31–0: WORD2[31:0]	D0TCM bits 7–0: WORD0[23:16]	Read
11	Cortex-M7 core 0 instruction cache tag1–0	Tag1 bits 21–0: WORD1[11:0], WORD2[31:22]	Tag1 bits 6–0: WORD0[31:25]	Read
		Tag0 bits 21–0: WORD2[21:0]	Tag0 bits 6–0: WORD0[24:18]	Read
12	Cortex-M7 core 0 instruction cache data1–0	Data1 bits 63–32: WORD1[31:0] Data1 bits 31–0: WORD2[31:0]	Data1 bits 7–0: WORD0[31:24]	Read
		Data0 bits 63–32: WORD3[31:0] Data0 bits 31–0: WORD4[31:0]	Data0 bits 7–0: WORD0[23:16]	Read
13	Cortex-M7 core 0 data cache tag3–0	Tag3 bits 25–0: WORD1[7:0], WORD2[31:14]	Tag3 bits 6–0: WORD0[31:25]	Read
		Tag2 bits 25–0: WORD2[13:0], WORD3[31:20]	Tag2 bits 6–0: WORD0[24:18]	Read
		Tag1 bits 25–0: WORD3[19:0], WORD4[31:26]	Tag1 bits 6–0: WORD0[17:11]	Read

Table continues on the next page...

Table 569. EICHD*n*_WORD register targets (continued)

Channel	Target	Data bits	Checkbits	Error injection path
		Tag0 bits 25–0: WORD4[25:0]	Tag0 bits 6–0: WORD0[10:4]	Read
14	Cortex-M7 core 0 data cache data0 bank3–0	Bank3 bits 31–0: WORD1[31:0]	Bank3 bits 6–0: WORD0[31:25]	Read
		Bank2 bits 31–0: WORD2[31:0]	Bank2 bits 6–0: WORD0[24:18]	Read
		Bank1 bits 31–0: WORD3[31:0]	Bank1 bits 6–0: WORD0[17:11]	Read
		Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 6–0: WORD0[10:4]	Read
15	Cortex-M7 core 0 data cache data1 bank3–0	Bank3 bits 31–0: WORD1[31:0]	Bank3 bits 6–0: WORD0[31:25]	Read
		Bank2 bits 31–0: WORD2[31:0]	Bank2 bits 6–0: WORD0[24:18]	Read
		Bank1 bits 31–0: WORD3[31:0]	Bank1 bits 6–0: WORD0[17:11]	Read
		Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 6–0: WORD0[10:4]	Read
16	Cortex-M7 core 1 DTCM	D1TCM bits 31–0: WORD1[31:0]	D1TCM bits 7–0: WORD0[31:24]	Read
		D0TCM bits 31–0: WORD2[31:0]	D0TCM bits 7–0: WORD0[23:16]	Read
17	Cortex-M7 core 1 instruction cache tag1–0	Tag1 bits 21–0: WORD1[11:0], WORD2[31:22]	Tag1 bits 6–0: WORD0[31:25]	Read
		Tag0 bits 21–0: WORD2[21:0]	Tag0 bits 6–0: WORD0[24:18]	Read
18	Cortex-M7 core 1 instruction cache data1–0	Data1 bits 63–32: WORD1[31:0] Data1 bits 31–0: WORD2[31:0]	Data1 bits 7–0: WORD0[31:24]	Read
		Data0 bits 63–32: WORD3[31:0] Data0 bits 31–0: WORD4[31:0]	Data0 bits 7–0: WORD0[23:16]	Read
19	Cortex-M7 core 1 data cache tag3–0	Tag3 bits 25–0: WORD1[7:0], WORD2[31:14]	Tag3 bits 6–0: WORD0[31:25]	Read
		Tag2 bits 25–0: WORD2[13:0], WORD3[31:20]	Tag2 bits 6–0: WORD0[24:18]	Read
		Tag1 bits 25–0: WORD3[19:0], WORD4[31:26]	Tag1 bits 6–0: WORD0[17:11]	Read

Table continues on the next page...

Table 569. EICHD*n*_WORD register targets (continued)

Channel	Target	Data bits	Checkbits	Error injection path
		Tag0 bits 25–0: WORD4[25:0]	Tag0 bits 6–0: WORD0[10:4]	Read
20	Cortex-M7 core 1 data cache data0 bank3–0	Bank3 bits 31–0: WORD1[31:0]	Bank3 bits 6–0: WORD0[31:25]	Read
		Bank2 bits 31–0: WORD2[31:0]	Bank2 bits 6–0: WORD0[24:18]	Read
		Bank1 bits 31–0: WORD3[31:0]	Bank1 bits 6–0: WORD0[17:11]	Read
		Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 6–0: WORD0[10:4]	Read
21	Cortex-M7 core 1 data cache data1 bank3–0	Bank3 bits 31–0: WORD1[31:0]	Bank3 bits 6–0: WORD0[31:25]	Read
		Bank2 bits 31–0: WORD2[31:0]	Bank2 bits 6–0: WORD0[24:18]	Read
		Bank1 bits 31–0: WORD3[31:0]	Bank1 bits 6–0: WORD0[17:11]	Read
		Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 6–0: WORD0[10:4]	Read
22	Cortex-M7 core 2 DTCM	D1TCM bits 31–0: WORD1[31:0]	D1TCM bits 7–0: WORD0[31:24]	Read
		D0TCM bits 31–0: WORD2[31:0]	D0TCM bits 7–0: WORD0[23:16]	Read
23	Cortex-M7 core 2 instruction cache tag1–0	Tag1 bits 21–0: WORD1[11:0], WORD2[31:22]	Tag1 bits 6–0: WORD0[31:25]	Read
		Tag0 bits 21–0: WORD2[21:0]	Tag0 bits 6–0: WORD0[24:18]	Read
24	Cortex-M7 core 2 instruction cache data1–0	Data1 bits 63–32: WORD1[31:0] Data1 bits 31–0: WORD2[31:0]	Data1 bits 7–0: WORD0[31:24]	Read
		Data0 bits 63–32: WORD3[31:0] Data0 bits 31–0: WORD4[31:0]	Data0 bits 7–0: WORD0[23:16]	Read
25	Cortex-M7 core 2 data cache tag3–0	Tag3 bits 25–0: WORD1[7:0], WORD2[31:14]	Tag3 bits 6–0: WORD0[31:25]	Read
		Tag2 bits 25–0: WORD2[13:0], WORD3[31:20]	Tag2 bits 6–0: WORD0[24:18]	Read
		Tag1 bits 25–0: WORD3[19:0], WORD4[31:26]	Tag1 bits 6–0: WORD0[17:11]	Read

Table continues on the next page...

Table 569. EICHD*n*_WORD register targets (continued)

Channel	Target	Data bits	Checkbits	Error injection path
		Tag0 bits 25–0: WORD4[25:0]	Tag0 bits 6–0: WORD0[10:4]	Read
26	Cortex-M7 core 2 data cache data0 bank3–0	Bank3 bits 31–0: WORD1[31:0]	Bank3 bits 6–0: WORD0[31:25]	Read
		Bank2 bits 31–0: WORD2[31:0]	Bank2 bits 6–0: WORD0[24:18]	Read
		Bank1 bits 31–0: WORD3[31:0]	Bank1 bits 6–0: WORD0[17:11]	Read
		Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 6–0: WORD0[10:4]	Read
27	Cortex-M7 core 2 data cache data1 bank3–0	Bank3 bits 31–0: WORD1[31:0]	Bank3 bits 6–0: WORD0[31:25]	Read
		Bank2 bits 31–0: WORD2[31:0]	Bank2 bits 6–0: WORD0[24:18]	Read
		Bank1 bits 31–0: WORD3[31:0]	Bank1 bits 6–0: WORD0[17:11]	Read
		Bank0 bits 31–0: WORD4[31:0]	Bank0 bits 6–0: WORD0[10:4]	Read
28	Cortex-M7 core 2 lockstep	Bits 29–0: WORD1[25:0], WORD2[31:28]	—	Read
	Cortex-M7 core 1 lockstep	Bits 29–0: WORD2[27:0], WORD3[31:30]	—	Read
	Cortex-M7 core 0 lockstep	Bits 29–0: WORD3[29:0]	—	Read
29	eDMA_1 lockstep	Bits 9–0: WORD1[19:10]	—	Read
	eDMA_0 lockstep	Bits 9–0: WORD1[9:0]	—	Read
30	GIC-500 lockstep	Bits 5–0: WORD1[17:12]	—	Read
	Cortex-A53 core lockstep	Bits 11–0: WORD1[11:0]	—	Read

Table 570. RCCU alarms corresponding to EIM channels 28-30

Channel	Target bit	RCCU alarms	Error injection path
28	Cortex-M7 cores 2, 1, and 0 lockstep bit 0	RCCU1 alarm and RCCU2 alarm for Cortex-M7 cores 2, 1, and 0	Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 1		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 2		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 3		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 4		Read

Table continues on the next page...

Table 570. RCCU alarms corresponding to EIM channels 28-30 (continued)

Channel	Target bit	RCCU alarms	Error injection path
	Cortex-M7 cores 2, 1, and 0 lockstep bit 5		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 6		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 7		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 8		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 9		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 10		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 11		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 12		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 13		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 14		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 15		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 16		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 17		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 18		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 19		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 20		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 21		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 22		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 23		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 24		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 25		Read

Table continues on the next page...

NXP Confidential NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8d162e5-4821-48a8-b251-390d3ab624c1

Table 570. RCCU alarms corresponding to EIM channels 28-30 (continued)

Channel	Target bit	RCCU alarms	Error injection path
	Cortex-M7 cores 2, 1, and 0 lockstep bit 26		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 27		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 28		Read
	Cortex-M7 cores 2, 1, and 0 lockstep bit 29		Read
29	eDMA_1 and eDMA_0 lockstep bit 0	RCCU1a alarm and RCCU1b alarm for eDMA_1 and eDMA_0	Read
	eDMA_1 and eDMA_0 lockstep bit 1		Read
	eDMA_1 and eDMA_0 lockstep bit 2		Read
	eDMA_1 and eDMA_0 lockstep bit 3		Read
	eDMA_1 and eDMA_0 lockstep bit 4		Read
	eDMA_1 and eDMA_0 lockstep bit 5		Read
	eDMA_1 and eDMA_0 lockstep bit 6		Read
	eDMA_1 and eDMA_0 lockstep bit 7		Read
	eDMA_1 and eDMA_0 lockstep bit 8	RCCU2a alarm and RCCU2b alarm for eDMA_1 and eDMA_0	Read
	eDMA_1 and eDMA_0 lockstep bit 9		Read
30	Cortex-A53 core lockstep bit 0	RCCU0a alarm and RCCU0b alarm for Cortex-A53 core	Read
	Cortex-A53 core lockstep bit 1		Read
	Cortex-A53 core lockstep bit 2		Read
	Cortex-A53 core lockstep bit 3		Read
	Cortex-A53 core lockstep bit 4	RCCU1a alarm and RCCU1b alarm for Cortex-A53 core	Read
	Cortex-A53 core lockstep bit 5		Read
	Cortex-A53 core lockstep bit 6	RCCU2a alarm and RCCU2b alarm for Cortex-A53 core	Read
	Cortex-A53 core lockstep bit 7		Read
	Cortex-A53 core lockstep bit 8		Read
	Cortex-A53 core lockstep bit 9		Read
	Cortex-A53 core lockstep bit 10	RCCU4a alarm and RCCU4b alarm for Cortex-A53 core	Read
	Cortex-A53 core lockstep bit 11		Read
	GIC-500 lockstep bit 0	RCCU0 alarm and RCCU1 alarm for GIC-500	Read
	GIC-500 lockstep bit 1		Read
	GIC-500 lockstep bit 2		Read
	GIC-500 lockstep bit 3		Read

Table continues on the next page...

Table 570. RCCU alarms corresponding to EIM channels 28-30 (continued)

Channel	Target bit	RCCU alarms	Error injection path
	GIC-500 lockstep bit 4		Read
	GIC-500 lockstep bit 5		Read

73.1.3 EIM_0 channel assignments

The single channel of the EIM_0 instance can inject errors on accesses to the two LSBs of target memory arrays.

The following table shows the assigned targets for channel 0.

- In the EIM programming model, channel 0 has a set of EICHD0_WORD registers for injecting errors.
- The table maps the two LSBs of each target to a pair of EICHD0_WORD register bits.

To inject a single-bit error on a target, write 1 to one of the two corresponding EICHD0_WORD bits. To inject a multi-bit error on a target, write 11b to the the pair of EICHD0_WORD data bits.

Table 571. Targets for the EIM_0 EICHD0_WORD registers

Target: two LSBs only	EICHD0_WORD data bits	Error injection path
Cortex-A53 cluster 0 core 0 data "dirty" RAM	WORD8[1:0]	Write
Cortex-A53 cluster 0 core 0 data cache tag bank0	WORD8[3:2]	Write
Cortex-A53 cluster 0 core 0 data cache tag bank1	WORD8[5:4]	Write
Cortex-A53 cluster 0 core 0 data cache tag bank2	WORD8[7:6]	Write
Cortex-A53 cluster 0 core 0 data cache tag bank3	WORD8[9:8]	Write
Cortex-A53 cluster 0 core 0 instruction cache data bank0 high bits	WORD8[11:10]	Write
Cortex-A53 cluster 0 core 0 instruction cache data bank0 low bits	WORD8[13:12]	Write
Cortex-A53 cluster 0 core 0 instruction cache data bank1 high bits	WORD8[15:14]	Write
Cortex-A53 cluster 0 core 0 instruction cache data bank1 low bits	WORD8[17:16]	Write
Cortex-A53 cluster 0 core 0 instruction cache tag RAM0	WORD8[19:18]	Write
Cortex-A53 cluster 0 core 0 instruction cache tag RAM1	WORD8[21:20]	Write
Cortex-A53 cluster 0 core 0 TLB bank0	WORD8[23:22]	Write
Cortex-A53 cluster 0 core 0 TLB bank1	WORD8[25:24]	Write
Cortex-A53 cluster 0 core 0 TLB bank2	WORD8[27:26]	Write
Cortex-A53 cluster 0 core 0 TLB bank3	WORD8[29:28]	Write
Cortex-A53 cluster 0 core 1 data "dirty" RAM	WORD8[31:30]	Write
Cortex-A53 cluster 0 core 1 data cache tag bank0	WORD7[1:0]	Write
Cortex-A53 cluster 0 core 1 data cache tag bank1	WORD7[3:2]	Write
Cortex-A53 cluster 0 core 1 data cache tag bank2	WORD7[5:4]	Write
Cortex-A53 cluster 0 core 1 data cache tag bank3	WORD7[7:6]	Write
Cortex-A53 cluster 0 core 1 instruction cache data bank0 high bits	WORD7[9:8]	Write

Table continues on the next page...

Table 571. Targets for the EIM_0 EICHD0_WORD registers (continued)

Target: two LSBs only	EICHD0_WORD data bits	Error injection path
Cortex-A53 cluster 0 core 1 instruction cache data bank0 low bits	WORD7[11:10]	Write
Cortex-A53 cluster 0 core 1 instruction cache data bank1 high bits	WORD7[13:12]	Write
Cortex-A53 cluster 0 core 1 instruction cache data bank1 low bits	WORD7[15:14]	Write
Cortex-A53 cluster 0 core 1 instruction cache tag RAM0	WORD7[17:16]	Write
Cortex-A53 cluster 0 core 1 instruction cache tag RAM1	WORD7[19:18]	Write
Cortex-A53 cluster 0 core 1 TLB bank0	WORD7[21:20]	Write
Cortex-A53 cluster 0 core 1 TLB bank1	WORD7[23:22]	Write
Cortex-A53 cluster 0 core 1 TLB bank2	WORD7[25:24]	Write
Cortex-A53 cluster 0 core 1 TLB bank3	WORD7[27:26]	Write
Cortex-A53 cluster 0 core 0 level 1 data cache tag RAM way0	WORD7[29:28]	Write
Cortex-A53 cluster 0 core 0 level 1 data cache tag RAM way1	WORD7[31:30]	Write
Cortex-A53 cluster 0 core 0 level 1 data cache tag RAM way2	WORD6[1:0]	Write
Cortex-A53 cluster 0 core 0 level 1 data cache tag RAM way3	WORD6[3:2]	Write
Cortex-A53 cluster 0 core 1 level 1 data cache tag RAM way0	WORD6[5:4]	Write
Cortex-A53 cluster 0 core 1 level 1 data cache tag RAM way1	WORD6[7:6]	Write
Cortex-A53 cluster 0 core 1 level 1 data cache tag RAM way2	WORD6[9:8]	Write
Cortex-A53 cluster 0 core 1 level 1 data cache tag RAM way3	WORD6[11:10]	Write
Cortex-A53 cluster 1 core 0 data "dirty" RAM	WORD6[13:12]	Write
Cortex-A53 cluster 1 core 0 data cache tag bank0	WORD6[15:14]	Write
Cortex-A53 cluster 1 core 0 data cache tag bank1	WORD6[17:16]	Write
Cortex-A53 cluster 1 core 0 data cache tag bank2	WORD6[19:18]	Write
Cortex-A53 cluster 1 core 0 data cache tag bank3	WORD6[21:20]	Write
Cortex-A53 cluster 1 core 0 instruction cache data bank0 high bits	WORD6[23:22]	Write
Cortex-A53 cluster 1 core 0 instruction cache data bank0 low bits	WORD6[25:24]	Write
Cortex-A53 cluster 1 core 0 instruction cache data bank1 high bits	WORD6[27:26]	Write
Cortex-A53 cluster 1 core 0 instruction cache data bank1 low bits	WORD6[29:28]	Write
Cortex-A53 cluster 1 core 0 instruction cache tag RAM0	WORD6[31:30]	Write
Cortex-A53 cluster 1 core 0 instruction cache tag RAM1	WORD5[1:0]	Write
Cortex-A53 cluster 1 core 0 TLB bank0	WORD5[3:2]	Write
Cortex-A53 cluster 1 core 0 TLB bank1	WORD5[5:4]	Write
Cortex-A53 cluster 1 core 0 TLB bank2	WORD5[7:6]	Write
Cortex-A53 cluster 1 core 0 TLB bank3	WORD5[9:8]	Write

Table continues on the next page...

Table 571. Targets for the EIM_0 EICHD0_WORD registers (continued)

Target: two LSBs only	EICHD0_WORD data bits	Error injection path
Cortex-A53 cluster 1 core 1 data "dirty" RAM	WORD5[11:10]	Write
Cortex-A53 cluster 1 core 1 data cache tag bank0	WORD5[13:12]	Write
Cortex-A53 cluster 1 core 1 data cache tag bank1	WORD5[15:14]	Write
Cortex-A53 cluster 1 core 1 data cache tag bank2	WORD5[17:16]	Write
Cortex-A53 cluster 1 core 1 data cache tag bank3	WORD5[19:18]	Write
Cortex-A53 cluster 1 core 1 instruction cache data bank0 high bits	WORD5[21:20]	Write
Cortex-A53 cluster 1 core 1 instruction cache data bank0 low bits	WORD5[23:22]	Write
Cortex-A53 cluster 1 core 1 instruction cache data bank1 high bits	WORD5[25:24]	Write
Cortex-A53 cluster 1 core 1 instruction cache data bank1 low bits	WORD5[27:26]	Write
Cortex-A53 cluster 1 core 1 instruction cache tag RAM0	WORD5[29:28]	Write
Cortex-A53 cluster 1 core 1 instruction cache tag RAM1	WORD5[31:30]	Write
Cortex-A53 cluster 1 core 1 TLB bank0	WORD4[1:0]	Write
Cortex-A53 cluster 1 core 1 TLB bank1	WORD4[3:2]	Write
Cortex-A53 cluster 1 core 1 TLB bank2	WORD4[5:4]	Write
Cortex-A53 cluster 1 core 1 TLB bank3	WORD4[7:6]	Write
Cortex-A53 cluster 1 core 0 level 1 data cache tag RAM way0	WORD4[9:8]	Write
Cortex-A53 cluster 1 core 0 level 1 data cache tag RAM way1	WORD4[11:10]	Write
Cortex-A53 cluster 1 core 0 level 1 data cache tag RAM way2	WORD4[13:12]	Write
Cortex-A53 cluster 1 core 0 level 1 data cache tag RAM way3	WORD4[15:14]	Write
Cortex-A53 cluster 1 core 1 level 1 data cache tag RAM way0	WORD4[17:16]	Write
Cortex-A53 cluster 1 core 1 level 1 data cache tag RAM way1	WORD4[19:18]	Write
Cortex-A53 cluster 1 core 1 level 1 data cache tag RAM way2	WORD4[21:20]	Write
Cortex-A53 cluster 1 core 1 level 1 data cache tag RAM way3	WORD4[23:22]	Write
GIC-500_0 ICB cache	WORD4[25:24]	Write
Reserved	WORD4[27:26]	Write
GIC-500_0 LPI cache	WORD4[29:28]	Write
GIC-500_1 ICB cache	WORD4[31:30]	Write
Reserved	WORD3[1:0]	Write
GIC-500_1 LPI cache	WORD3[3:2]	Write
Reserved	WORD3[31:4]	Write

73.1.4 EIM_1 channel assignments

The EIM_1 instance channels can inject errors on accesses to data bits of its target memory arrays.

The following table shows the channel assignments and corresponding targets.

- In the EIM programming model, each channel has a set of EICHD*n*_WORD registers for injecting errors.
- The table maps the targets to their corresponding WORD register bits.

To cause a memory array controller to generate an error event, write 1 to the designated WORD register bit.

Table 572. Targets for the EIM_1 EICHD*n*_WORD registers

Channel	Target	WORD data bits	Error injection path
0	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—
	Reserved	WORD4[31:0]	—
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—
1	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—
	Reserved	WORD4[31:0]	—
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—
2	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—
	Reserved	WORD4[31:0]	—
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—
3	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—
	Reserved	WORD4[31:0]	—

Table continues on the next page...

Table 572. Targets for the EIM_1 EICHD*n*_WORD registers (continued)

Channel	Target	WORD data bits	Error injection path
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—
4	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—
	Reserved	WORD4[31:0]	—
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—
5	QuadSPI transmit buffer memory	WORD8[31:0]	Read
	QuadSPI transmit buffer inject error on ECC bits	WORD7[7:0]	Read
	Reserved	WORD7[31:8]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—
	Reserved	WORD4[31:0]	—
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—
6	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—
	Reserved	WORD4[31:0]	—
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—
7	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—

Table continues on the next page...

Table 572. Targets for the EIM_1 EICHD*n*_WORD registers (continued)

Channel	Target	WORD data bits	Error injection path
	Reserved	WORD4[31:0]	—
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—
8	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—
	Reserved	WORD4[31:0]	—
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—
9	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—
	Reserved	WORD4[31:0]	—
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—
10	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—
	Reserved	WORD4[31:0]	—
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—
11	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—

Table continues on the next page...

Table 572. Targets for the EIM_1 EICHD*n*_WORD registers (continued)

Channel	Target	WORD data bits	Error injection path
	Reserved	WORD4[31:0]	—
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—
12	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
		WORD5[31:0]	—
		WORD4[7:0]	—
	Reserved	WORD4[31:8]	—
	QuadSPI AHB read buffer memory	WORD3[31:0]	Read
		WORD2[31:0]	Read
	QuadSPI AHB read buffer inject error on ECC bits	WORD1[7:0]	Read
	Reserved	WORD1[31:8]	—
13	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—
	Reserved	WORD4[31:0]	—
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—
14	Reserved	WORD8[31:0]	—
	Reserved	WORD7[31:0]	—
	Reserved	WORD6[31:0]	—
	Reserved	WORD5[31:0]	—
	Reserved	WORD4[31:0]	—
	Reserved	WORD3[31:0]	—
	Reserved	WORD2[31:0]	—
	Reserved	WORD1[31:0]	—

73.1.5 EIM_2 channel assignments

The EIM_2 instance channels can inject errors on accesses to data bits of its target memory arrays.

The following table shows the channel assignments and corresponding targets.

- In the EIM programming model, each channel has a set of EICHD*n*_WORD registers for injecting errors.
- The table maps the target bits to the WORD register bits.

To cause a memory array controller to generate an error event, write 1 to the designated WORD register bit.

Table 573. Targets for the EIM_2 EICHD*n*_WORD registers

Channel	Target	Data bits	Error injection path
0	SerDes subsystem master read request buffer_0	Buffer_0[31:0]: WORD8[31:0]	Write
		Buffer_0[63:32]: WORD7[31:0]	Write
		Buffer_0[95:64]: WORD6[31:0]	Write
	SerDes subsystem master read request buffer_1	Buffer_1[31:0]: WORD5[31:0]	Write
		Buffer_1[63:32]: WORD4[31:0]	Write
		Buffer_1[95:64]: WORD3[31:0]	Write
	SerDes subsystem transmit retry buffer	Buffer[31:0]: WORD2[31:0]	Write
		Buffer[46:32]: WORD1[14:0]	Write
	Reserved	WORD1[31:15]	
1	SerDes subsystem slave TLP completion data segment composer	Composer[31:0]: WORD8[31:0]	Write
		Composer[63:32]: WORD7[31:0]	Write
		Composer[87:64]: WORD6[23:0]	Write
	Reserved	WORD6[31:24]	
	SerDes subsystem DMA channel context RAM_0	RAM_0[31:0]: WORD5[31:0]	Write
		RAM_0[63:32]: WORD4[31:0]	Write
		RAM_0[95:64]: WORD3[31:0]	Write
		RAM_0[103:96]: WORD2[7:0]	Write
	Reserved	WORD2[31:8]	
	SerDes subsystem transmit SOT buffer	Buffer[15:0]: WORD1[15:0]	Write
	Reserved	WORD1[31:16]	
2	SerDes subsystem DMA channel context RAM_1	RAM_1[31:0]: WORD8[31:0]	Write
		RAM_1[63:32]: WORD7[31:0]	Write
		RAM_1[95:64]: WORD6[31:0]	Write
		RAM_1[103:96]: WORD5[7:0]	Write
	Reserved	WORD5[31:8]	
	SerDes subsystem DMA channel context RAM_2	RAM_2[31:0]: WORD4[31:0]	Write
		RAM_2[63:32]: WORD3[31:0]	Write
		RAM_2[95:64]: WORD2[31:0]	Write
		RAM_2[103:96]: WORD1[7:0]	Write

Table continues on the next page...

Table 573. Targets for the EIM_2 EICHD*n*_WORD registers (continued)

Channel	Target	Data bits	Error injection path
	Reserved	WORD1[31:8]	
3	SerDes subsystem slave write set-aside buffer_0	Buffer_0[31:0]: WORD8[31:0]	Write
		Buffer_0[63:32]: WORD7[31:0]	Write
		Buffer_0[85:64]: WORD6[21:0]	Write
	Reserved	WORD6[31:22]	
	SerDes subsystem slave write set-aside buffer_1	Buffer_1[31:0]: WORD5[31:0]	Write
		Buffer_1[63:32]: WORD4[31:0]	Write
		Buffer_1[85:64]: WORD3[21:0]	Write
	Reserved	WORD3[31:22]	
	SerDes subsystem receive queue data	Data[31:0]: WORD2[31:0]	Write
		Data[45:32]: WORD1[13:0]	Write
	Reserved	WORD1[31:14]	
4	SerDes subsystem receive queue header_0	Header_0[31:0]: WORD8[31:0]	Write
		Header_0[63:32]: WORD7[31:0]	Write
		Header_0[85:64]: WORD6[21:0]	Write
	Reserved	WORD6[31:22]	
	SerDes subsystem receive queue header_1	Header_1[31:0]: WORD5[31:0]	Write
		Header_1[63:32]: WORD4[31:0]	Write
		Header_1[85:64]: WORD3[21:0]	Write
	Reserved	WORD3[31:22]	
	SerDes subsystem DMA read buffer RAM	RAM[31:0]: WORD2[31:0]	Write
		RAM[38:32]: WORD1[6:0]	Write
	Reserved	WORD1[31:7]	
5	SerDes subsystem master completion buffer	Buffer[31:0]: WORD8[31:0]	Write
		Buffer[63:32]: WORD7[31:0]	Write
		Buffer[86:64]: WORD6[22:0]	Write
	Reserved	WORD6[31:23]	

73.1.6 EIM_3 channel assignments

The EIM_3 instance channels can inject errors on accesses to data bits of its target memory arrays.

The following table shows the channel assignments and corresponding targets.

- In the EIM programming model, each channel has a set of EICHD*n*_WORD registers for injecting errors.
- The table maps the target bits to the WORD register bits.

To cause a memory array controller to generate an error event, write 1 to the designated WORD register bit.

Table 574. Targets for the EIM_3 EICHD*n*_WORD registers

Channel	Target	Data bits	Error Injection Path
0	DDR subsystem read reorder data buffer_1	WORD8[31:0]	Write
		WORD7[31:0]	Write
		WORD6[31:0]	Write
		WORD5[31:0]	Write
		WORD4[17:0]	Write
	Reserved	WORD4[31:18]	
1	DDR subsystem read reorder data buffer_2	WORD8[31:0]	Write
		WORD7[31:0]	Write
		WORD6[31:0]	Write
		WORD5[31:0]	Write
		WORD4[17:0]	Write
	Reserved	WORD4[31:18]	
2	DDR subsystem read reorder data buffer_0	WORD8[31:0]	Write
		WORD7[31:0]	Write
		WORD6[31:0]	Write
		WORD5[31:0]	Write
		WORD4[17:0]	Write
	Reserved	WORD4[31:18]	
3	DDR subsystem write data buffer	WORD8[31:0]	Write
		WORD7[31:0]	Write
		WORD6[31:0]	Write
		WORD5[31:0]	Write
		WORD4[15:0]	Write
	Reserved	WORD4[31:16]	

73.1.7 Constraints to use EIM on A53 cache memories

Cortex-A53 processor protects against the soft errors in the cache memories upon detection of a single-bit ECC error. In such an event, the A53 processor attempts to correct the data and update the memory, then read the line to confirm the correction. In the case of a hard error, such as the persistent injection by the EIM, until cleared by the Software, the processor might enter into a continuous livelock loop. Thus, care must be taken when using EIM to inject errors in the A53 processor cache memories to narrow the scope of any such safety diagnostic software. Here are the steps for this:

1. Clear all cache

```
ca53_invalidate_icache_all();
```

```
ca53_flush_dcache_all();
```

2. Read data into cache

```
/*Start Performance Monitoring*/
```

```
ca53_api_start_pmu();
```

3. Enable EIM

```
ca53_eim_enable(ECC_TYPE[L1_ECC_REGIONS[i]].ECC_TARGET, ECC_TYPE[L1_ECC_REGIONS[i]].OFFSET,
A53_single_bit_error, L1_ECC_Channel);
```

4. Read from cache

```
data32 = *(volatile uint32_t*)(addr);
```

Read/Writes Data into Cache with error

```
for(n=addr; n>(addr+size);n=n+4){*(volatile uint32_t*)(n + 0x1000) = *(volatile uint32_t*)(n);}
```

5. Disable EIM

```
ca53_eim_disable(ECC_TYPE[L1_ECC_REGIONS[i]].ECC_TARGET, ECC_TYPE[L1_ECC_REGIONS[i]].OFFSET,
A53_single_bit_error, L1_ECC_Channel);
```

Enable and program EIM to inject A53 cache memory error, then a single cacheline can be read to induce ECC reaction. Then, disable EIM. Reading more than one cacheline with EIM error injection active can lead to A53 livelock scenario. For details on the A53 processor livelock behavior, see ARM Cortex-A53 MPCore Processor Technical Reference Manual.

73.2 Chip-specific EIM information

73.2.1 EIM instances

The following table summarizes the features of EIM instances on this chip.

Table 575. EIM instance summary

EIM instance	Number of channels	Targets
EIM_MISC	17	Memory arrays for: • Standby SRAM • PCIe_1
EIM_LLCE	6	Memory arrays for: • LLCE Data RAM_A • LLCE Data RAM_B • LLCE FRPE RAM • LLCE TxPPE RAM • LLCE RxPPE RAM • LLCE DTE RAM

Table continues on the next page...

Table 575. EIM instance summary (continued)

EIM instance	Number of channels	Targets
EIM_PFE (EIM_PFE0-EIM_PFE6)	155	Memory arrays for: • class_inq_mem • lmem • mac_hash • pe0_dram • pe0_dram_lpe0 • pe0_dram_lpe1 • pe0_iram • pe1_dram • pe1_dram_lpe0 • pe1_dram_lpe1 • pe1_iram • pe2_dram • pe2_dram_lpe0 • pe2_dram_lpe1 • pe2_iram • pe3_dram • pe3_dram_lpe0 • pe3_dram_lpe1 • pe3_iram • pe4_dram • pe4_dram_lpe0 • pe4_dram_lpe1 • pe4_iram • pe5_dram • pe5_dram_lpe0 • pe5_dram_lpe1 • pe5_iram • pe6_dram • pe6_dram_lpe0 • pe6_dram_lpe1 • pe6_iram • pe7_dram

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e2a-6621-42a8-b251-390d3ab624c1

Table 575. EIM instance summary (continued)

EIM instance	Number of channels	Targets
		• pe7_dram_lpe0 • pe7_dram_lpe1 • pe7_iram • pe_lmem • ro_mem • smem2arb_mem_rd • vlan_hash • egpix_rxf_rd • egpix_txf_rd • emac0_iq_class_entry • emac0_iq_lmem1_rd • emac0_iq_lmem2_rd • emac0_iq_lmem3_rd • emac0_iq_lmem4_rd • emac0_iq_lmem5_rd • emac0_iq_lmem6_rd • emac0_iq_lmem7_rd • emac0_iq_lmem8_rd • emac0_iq_lru_count • emac1_iq_class_entry • emac1_iq_lmem1_rd • emac1_iq_lmem2_rd • emac1_iq_lmem3_rd • emac1_iq_lmem4_rd • emac1_iq_lmem5_rd • emac1_iq_lmem6_rd • emac1_iq_lmem7_rd • emac1_iq_lmem8_rd • emac1_iq_lru_count • emac2_iq_class_entry • emac2_iq_lmem1_rd • emac2_iq_lmem2_rd • emac2_iq_lmem3_rd

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHIPUNION Technology Co., Ltd.
8c4622e 4a214a3a 0251-390d3ab624c1

Table 575. EIM instance summary (continued)

EIM instance	Number of channels	Targets
		• emac2_iq_lmem4_rd • emac2_iq_lmem5_rd • emac2_iq_lmem6_rd • emac2_iq_lmem7_rd • emac2_iq_lmem8_rd • emac2_iq_lru_count • etgpix_rxf_rd • hgpi1_rxf_rd • hgpi1_txf_rd • hif1_dma_dxr1_drx_rd • hif1_dma_dxr1_dtx_rd • hif1_dma_dxr2_drx_rd • hif1_dma_dxr2_dtx_rd • hif1_dma_dxr3_drx_rd • hif1_dma_dxr3_dtx_rd • hif1_dma_dxr4_drx_rd • hif1_dma_dxr4_dtx_rd • hif1_dma_dxr_drx_ctrl_word_rd • hif1_dma_dxr_dtx_ctrl_word_rd • hif1_rx_dma_bdp_rd_burst_rd • hif1_rx_dma_bdp_wr_burst_rd • hif1_tx_bd_ctrl_rd • hif1_tx_dma_bdp_rd_burst_rd • hif1_tx_dma_bdp_wr_burst_rd • bcnt • bcnt2 • bmu • bmu2 • reclaim_buf_rd • teq_qfifo_0 • teq_qfifo_1 • tlite_cntx • tlite_inq_rd

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHANGJIANG Innovative Technology Co., Ltd.
8c4625c4821e4c38e251-390d3ab624c1

Table 575. EIM instance summary (continued)

EIM instance	Number of channels	Targets
		• upe_dram • upe_dram_lpe0 • upe_dram_lpe1 • upe_inq_mem • upe_iram

73.2.2 EIM_MISC channel assignments

Table 2 Targets of EIM_S32G channel's EICHDn_WORD registers

Channel	Target (bus size)	Data bits
0	Standby SRAM bank0 (72 bits)	Bank0 bits 31-0: WORD1[31:0] Bank0 bits 63-32: WORD2[31:0] Bank0 bits 71-65: WORD3[7:0]
1	Standby SRAM bank1 (72 bits)	Bank1 bits 31-0: WORD1[31:0] Bank1 bits 63-32: WORD2[31:0] Bank1 bits 71-65: WORD3[7:0]
2	PCIe_1 DMA_Channel_Context_RAM_0 (104 bits)	31-0 : WORD1[31:0] 63-32: WORD2[31:0] 95-64: WORD3[31:0] 104-96: WORD4[7:0]
3	PCIe_1 DMA_Channel_Context_RAM_1 (104 bits)	31-0 : WORD1[31:0] 63-32: WORD2[31:0] 95-64: WORD3[31:0] 104-96: WORD4[7:0]
4	PCIe_1 DMA_Channel_Context_RAM_2 (104 bits)	31-0 : WORD1[31:0] 63-32: WORD2[31:0] 95-64: WORD3[31:0] 104-96: WORD4[7:0]
5	PCIe_1 DMA_Read_Buffer_RAM (39 bits)	31-0 : WORD1[31:0] 38-32: WORD2[6:0]
6	PCIe_1	31-0 : WORD1[31:0] 63-32: WORD2[31:0]

Table continues on the next page...

Table continued from the previous page...

	Master_Completion_Buffer (87 bits)	86-64: WORD3[22:0]
7	PCIe_1 Master_Read_Request_Buffer_0 (96 bits)	31-0 : WORD1[31:0] 63-32: WORD2[31:0] 95-64: WORD3[31:0]
8	PCIe_1 Master_Read_Request_Buffer_1 (96 bits)	31-0 : WORD1[31:0] 63-32: WORD2[31:0] 95-64: WORD3[31:0]
9	PCIe_1 Receive_Queue_Data (46 bits)	31-0 : WORD1[31:0] 45-32: WORD2[13:0]
10	PCIe_1 Receive_Queue_Header_0 (86 bits)	31-0 : WORD1[31:0] 63-32: WORD2[31:0] 85-64: WORD3[21:0]
11	PCIe_1 Receive_Queue_Header_1 (86 bits)	31-0 : WORD1[31:0] 63-32: WORD2[31:0] 85-64: WORD3[21:0]
12	PCIe_1 Slave_TLP_Completion_Data_Segment_Composer (88 bits)	31-0 : WORD1[31:0] 63-32: WORD2[31:0] 87-64: WORD3[23:0]
13	PCIe_1 Slave_Write_Set_Aside_Buffer_0 (86 bits)	31-0 : WORD1[31:0] 63-32: WORD2[31:0] 85-64: WORD3[21:0]
14	PCIe_1 Slave_Write_Set_Aside_Buffer_1 (86 bits)	31-0 : WORD1[31:0] 63-32: WORD2[31:0] 85-64: WORD3[21:0]
15	PCIe_1 Transmit_Retry_Buffer (47 bits)	31-0 : WORD1[31:0] 46-32: WORD2[14:0]
16	PCIe_1 Transmit_SOT_Buffer (16 bits)	15-0 : WORD1[15:0]

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESR Automotive Technology Co., Ltd.
e8462e54-8214-48ad-b251-3003e16d1d

73.2.3 EIM_PFE channel assignments

Table 2 Targets of PFE EIM channel's EICHDn_WORD registers

Channel	Target (bus size)	Data bits
EIM0-CH00	class_inq_mem (39 bits)	Bank0 bits 31-0: WORD0[31:0] Bank0 bits 38-32: WORD1[6:0]
EIM0-CH01	lmem (72 bits)	Bank1 bits 31-0: WORD0[31:0] Bank1 bits 63-32: WORD1[31:0] Bank1 bits 71-64: WORD2[7:0]
EIM0-CH02	mac_hash (140 bits)	Bank2 bits 31-0: WORD0[31:0] Bank2 bits 63-32: WORD1[31:0] Bank2 bits 95-64: WORD2[31:0] Bank2 bits 127-96: WORD3[31:0] Bank2 bits 139-128: WORD4[11:0]
EIM0-CH03	pe0_dram (78 bits)	Bank3 bits 31-0: WORD0[31:0] Bank3 bits 63-32: WORD1[31:0] Bank3 bits 77-64: WORD2[13:0]
EIM0-CH04	pe0_dram (26 bits)	Bank4 bits 25-0: WORD0[25:0]
EIM0-CH05	pe0_dram_lpe0 (52 bits)	Bank5 bits 31-0: WORD0[31:0] Bank5 bits 51-32: WORD1[19:0]
EIM0-CH06	pe0_dram_lpe1 (52 bits)	Bank6 bits 31-0: WORD0[31:0] Bank6 bits 51-32: WORD1[19:0]
EIM0-CH07	pe0_iram (39 bits)	Bank7 bits 31-0: WORD0[31:0] Bank7 bits 38-32: WORD1[6:0]
EIM0-CH08	pe1_dram (78 bits)	Bank8 bits 31-0: WORD0[31:0] Bank8 bits 63-32: WORD1[31:0] Bank8 bits 77-64: WORD2[13:0]
EIM0-CH09	pe1_dram (26 bits)	Bank9 bits 25-0: WORD0[25:0]
EIM0-CH10	pe1_dram_lpe0 (52 bits)	Bank10 bits 31-0: WORD0[31:0] Bank10 bits 51-32: WORD1[19:0]
EIM0-CH11	pe1_dram_lpe1 (52 bits)	Bank11 bits 31-0: WORD0[31:0] Bank11 bits 51-32: WORD1[19:0]
EIM0-CH12	pe1_iram (39 bits)	Bank12 bits 31-0: WORD0[31:0] Bank12 bits 38-32: WORD1[6:0]
EIM0-CH13	pe2_dram (78 bits)	Bank13 bits 31-0: WORD0[31:0] Bank13 bits 63-32: WORD1[31:0] Bank13 bits 77-64: WORD2[13:0]
EIM0-CH14	pe2_dram (26 bits)	Bank14 bits 25-0: WORD0[25:0]
EIM0-CH15	pe2_dram_lpe0 (52 bits)	Bank15 bits 31-0: WORD0[31:0] Bank15 bits 51-32: WORD1[19:0]
EIM0-CH16	pe2_dram_lpe1 (52 bits)	Bank16 bits 31-0: WORD0[31:0] Bank16 bits 51-32: WORD1[19:0]

Table continues on the next page...

Table continued from the previous page...

EIM0-CH17	pe2_iram (39 bits)	Bank17 bits 31-0: WORD0[31:0] Bank17 bits 38-32: WORD1[6:0]
EIM0-CH18	pe3_dram (78 bits)	Bank18 bits 31-0: WORD0[31:0] Bank18 bits 63-32: WORD1[31:0] Bank18 bits 77-64: WORD2[13:0]
EIM0-CH19	pe3_dram (26 bits)	Bank19 bits 25-0: WORD0[25:0]
EIM0-CH20	pe3_dram_lpe0 (52 bits)	Bank20 bits 31-0: WORD0[31:0] Bank20 bits 51-32: WORD1[19:0]
EIM0-CH21	pe3_dram_lpe1 (52 bits)	Bank21 bits 31-0: WORD0[31:0] Bank21 bits 51-32: WORD1[19:0]
EIM0-CH22	pe3_iram (39 bits)	Bank22 bits 31-0: WORD0[31:0] Bank22 bits 38-32: WORD1[6:0]
EIM0-CH23	pe4_dram (78 bits)	Bank23 bits 31-0: WORD0[31:0] Bank23 bits 63-32: WORD1[31:0] Bank23 bits 77-64: WORD2[13:0]
EIM0-CH24	pe4_dram (26 bits)	Bank24 bits 25-0: WORD0[25:0]
EIM0-CH25	pe4_dram_lpe0 (52 bits)	Bank25 bits 31-0: WORD0[31:0] Bank25 bits 51-32: WORD1[19:0]
EIM0-CH26	pe4_dram_lpe1 (52 bits)	Bank26 bits 31-0: WORD0[31:0] Bank26 bits 51-32: WORD1[19:0]
EIM0-CH27	pe4_iram (39 bits)	Bank27 bits 31-0: WORD0[31:0] Bank27 bits 38-32: WORD1[6:0]
EIM0-CH28	pe5_dram (78 bits)	Bank28 bits 31-0: WORD0[31:0] Bank28 bits 63-32: WORD1[31:0] Bank28 bits 77-64: WORD2[13:0]
EIM0-CH29	pe5_dram (26 bits)	Bank29 bits 25-0: WORD0[25:0]
EIM0-CH30	pe5_dram_lpe0 (52 bits)	Bank30 bits 31-0: WORD0[31:0] Bank30 bits 51-32: WORD1[19:0]
EIM0-CH31	pe5_dram_lpe1 (52 bits)	Bank31 bits 31-0: WORD0[31:0] Bank31 bits 51-32: WORD1[19:0]
EIM1-CH00	pe5_iram (39 bits)	Bank0 bits 31-0: WORD0[31:0] Bank0 bits 38-32: WORD1[6:0]
EIM1-CH01	pe6_dram (78 bits)	Bank1 bits 31-0: WORD0[31:0] Bank1 bits 63-32: WORD1[31:0] Bank1 bits 77-64: WORD2[13:0]
EIM1-CH02	pe6_dram (26 bits)	Bank2 bits 25-0: WORD0[25:0]
EIM1-CH03	pe6_dram_lpe0 (52 bits)	Bank3 bits 31-0: WORD0[31:0] Bank3 bits 51-32: WORD1[19:0]
EIM1-CH04	pe6_dram_lpe1 (52 bits)	Bank4 bits 31-0: WORD0[31:0] Bank4 bits 51-32: WORD1[19:0]

Table continues on the next page...

Table continued from the previous page...

EIM1-CH05	pe6_iram (39 bits)	Bank5 bits 31-0: WORD0[31:0] Bank5 bits 38-32: WORD1[6:0]
EIM1-CH06	pe7_dram (78 bits)	Bank6 bits 31-0: WORD0[31:0] Bank6 bits 63-32: WORD1[31:0] Bank6 bits 77-64: WORD2[13:0]
EIM1-CH07	pe7_dram (26 bits)	Bank7 bits 25-0: WORD0[25:0]
EIM1-CH08	pe7_dram_lpe0 (52 bits)	Bank8 bits 31-0: WORD0[31:0] Bank8 bits 51-32: WORD1[19:0]
EIM1-CH09	pe7_dram_lpe1 (52 bits)	Bank9 bits 31-0: WORD0[31:0] Bank9 bits 51-32: WORD1[19:0]
EIM1-CH10	pe7_iram (39 bits)	Bank10 bits 31-0: WORD0[31:0] Bank10 bits 38-32: WORD1[6:0]
EIM1-CH11	pe_lmem (52 bits)	Bank11 bits 31-0: WORD0[31:0] Bank11 bits 51-32: WORD1[19:0]
EIM1-CH12	ro (72 bits)	Bank12 bits 31-0: WORD0[31:0] Bank12 bits 63-32: WORD1[31:0] Bank12 bits 71-64: WORD2[7:0]
EIM1-CH13	smem2arb_mem_rd (144 bits)	Bank13 bits 31-0: WORD0[31:0] Bank13 bits 63-32: WORD1[31:0] Bank13 bits 95-64: WORD2[31:0] Bank13 bits 127-96: WORD3[31:0] Bank13 bits 143-128: WORD4[15:0]
EIM1-CH14	vlan_hash (116 bits)	Bank14 bits 31-0: WORD0[31:0] Bank14 bits 63-32: WORD1[31:0] Bank14 bits 95-64: WORD2[31:0] Bank14 bits 115-96: WORD3[19:0]
EIM2-CH00	EIM_indata_axi2npu_m1_rdata (16 bits)	Bank0 bits 15-0: WORD0[15:0]
EIM2-CH01	EIM_indata_axi2npu_m2_rdata (16 bits)	Bank1 bits 15-0: WORD0[15:0]
EIM2-CH02	EIM_indata_axi2npu_m3_rdata (16 bits)	Bank2 bits 15-0: WORD0[15:0]
EIM2-CH03	EIM_indata_axi2npu_m4_rdata (16 bits)	Bank3 bits 15-0: WORD0[15:0]
EIM2-CH04	EIM_indata_npu2axi_m1_araddr (8 bits)	Bank4 bits 7-0: WORD0[7:0]
EIM2-CH05	EIM_indata_npu2axi_m1_awaddr (8 bits)	Bank5 bits 7-0: WORD0[7:0]
EIM2-CH06	EIM_indata_npu2axi_m1_wdata (16 bits)	Bank6 bits 15-0: WORD0[15:0]
EIM2-CH07	EIM_indata_npu2axi_m2_araddr (8 bits)	Bank7 bits 7-0: WORD0[7:0]
EIM2-CH08	EIM_indata_npu2axi_m2_awaddr (8 bits)	Bank8 bits 7-0: WORD0[7:0]
EIM2-CH09	EIM_indata_npu2axi_m2_wdata (16 bits)	Bank9 bits 15-0: WORD0[15:0]

Table continues on the next page...

Table continued from the previous page...

EIM2-CH10	EIM_indata_npu2axi_m3_araddr (8 bits)	Bank10 bits 7-0: WORD0[7:0]
EIM2-CH11	EIM_indata_npu2axi_m3_awaddr (8 bits)	Bank11 bits 7-0: WORD0[7:0]
EIM2-CH12	EIM_indata_npu2axi_m3_wdata (16 bits)	Bank12 bits 15-0: WORD0[15:0]
EIM2-CH13	EIM_indata_npu2axi_m4_araddr (8 bits)	Bank13 bits 7-0: WORD0[7:0]
EIM2-CH14	EIM_indata_npu2axi_m4_awaddr (8 bits)	Bank14 bits 7-0: WORD0[7:0]
EIM2-CH15	EIM_indata_npu2axi_m4_wdata (16 bits)	Bank15 bits 15-0: WORD0[15:0]
EIM2-CH16	egpix_rxf_rd (72 bits)	Bank16 bits 31-0: WORD0[31:0] Bank16 bits 63-32: WORD1[31:0] Bank16 bits 71-64: WORD2[7:0]
EIM2-CH17	egpix_rxf_rd (72 bits)	Bank17 bits 31-0: WORD0[31:0] Bank17 bits 63-32: WORD1[31:0] Bank17 bits 71-64: WORD2[7:0]
EIM2-CH18	egpix_rxf_rd (72 bits)	Bank18 bits 31-0: WORD0[31:0] Bank18 bits 63-32: WORD1[31:0] Bank18 bits 71-64: WORD2[7:0]
EIM2-CH19	egpix_txf_rd (72 bits)	Bank19 bits 31-0: WORD0[31:0] Bank19 bits 63-32: WORD1[31:0] Bank19 bits 71-64: WORD2[7:0]
EIM2-CH20	egpix_txf_rd (72 bits)	Bank20 bits 31-0: WORD0[31:0] Bank20 bits 63-32: WORD1[31:0] Bank20 bits 71-64: WORD2[7:0]
EIM2-CH21	egpix_txf_rd (72 bits)	Bank21 bits 31-0: WORD0[31:0] Bank21 bits 63-32: WORD1[31:0] Bank21 bits 71-64: WORD2[7:0]
EIM2-CH22	emac0_iq_class_entry (288 bits)	Bank22 bits 31-0: WORD0[31:0] Bank22 bits 63-32: WORD1[31:0] Bank22 bits 95-64: WORD2[31:0] Bank22 bits 127-96: WORD3[31:0] Bank22 bits 159-128: WORD4[31:0] Bank22 bits 191-160: WORD5[31:0] Bank22 bits 223-192: WORD6[31:0] Bank22 bits 255-224: WORD7[31:0] Bank22 bits 287-256: WORD8[31:0]
EIM2-CH23	emac0_iq_lmem1_rd (72 bits)	Bank23 bits 31-0: WORD0[31:0] Bank23 bits 63-32: WORD1[31:0] Bank23 bits 71-64: WORD2[7:0]
EIM2-CH24	emac0_iq_lmem2_rd (72 bits)	Bank24 bits 31-0: WORD0[31:0] Bank24 bits 63-32: WORD1[31:0] Bank24 bits 71-64: WORD2[7:0]

Table continues on the next page...

Table continued from the previous page...

EIM2-CH25	emac0_iq_lmem3_rd (72 bits)	Bank25 bits 31-0: WORD0[31:0] Bank25 bits 63-32: WORD1[31:0] Bank25 bits 71-64: WORD2[7:0]
EIM2-CH26	emac0_iq_lmem4_rd (72 bits)	Bank26 bits 31-0: WORD0[31:0] Bank26 bits 63-32: WORD1[31:0] Bank26 bits 71-64: WORD2[7:0]
EIM2-CH27	emac0_iq_lmem5_rd (72 bits)	Bank27 bits 31-0: WORD0[31:0] Bank27 bits 63-32: WORD1[31:0] Bank27 bits 71-64: WORD2[7:0]
EIM2-CH28	emac0_iq_lmem6_rd (72 bits)	Bank28 bits 31-0: WORD0[31:0] Bank28 bits 63-32: WORD1[31:0] Bank28 bits 71-64: WORD2[7:0]
EIM2-CH29	emac0_iq_lmem7_rd (72 bits)	Bank29 bits 31-0: WORD0[31:0] Bank29 bits 63-32: WORD1[31:0] Bank29 bits 71-64: WORD2[7:0]
EIM2-CH30	emac0_iq_lmem8_rd (72 bits)	Bank30 bits 31-0: WORD0[31:0] Bank30 bits 63-32: WORD1[31:0] Bank30 bits 71-64: WORD2[7:0]
EIM2-CH31	emac0_iq_lru_count (23 bits)	Bank31 bits 22-0: WORD0[22:0]
EIM3-CH00	emac1_iq_class_entry (288 bits)	Bank0 bits 31-0: WORD0[31:0] Bank0 bits 63-32: WORD1[31:0] Bank0 bits 95-64: WORD2[31:0] Bank0 bits 127-96: WORD3[31:0] Bank0 bits 159-128: WORD4[31:0] Bank0 bits 191-160: WORD5[31:0] Bank0 bits 223-192: WORD6[31:0] Bank0 bits 255-224: WORD7[31:0] Bank0 bits 287-256: WORD8[31:0]
EIM3-CH01	emac1_iq_lmem1_rd (72 bits)	Bank1 bits 31-0: WORD0[31:0] Bank1 bits 63-32: WORD1[31:0] Bank1 bits 71-64: WORD2[7:0]
EIM3-CH02	emac1_iq_lmem2_rd (72 bits)	Bank2 bits 31-0: WORD0[31:0] Bank2 bits 63-32: WORD1[31:0] Bank2 bits 71-64: WORD2[7:0]
EIM3-CH03	emac1_iq_lmem3_rd (72 bits)	Bank3 bits 31-0: WORD0[31:0] Bank3 bits 63-32: WORD1[31:0] Bank3 bits 71-64: WORD2[7:0]
EIM3-CH04	emac1_iq_lmem4_rd (72 bits)	Bank4 bits 31-0: WORD0[31:0] Bank4 bits 63-32: WORD1[31:0] Bank4 bits 71-64: WORD2[7:0]
EIM3-CH05	emac1_iq_lmem5_rd (72 bits)	Bank5 bits 31-0: WORD0[31:0] Bank5 bits 63-32: WORD1[31:0] Bank5 bits 71-64: WORD2[7:0]

Table continues on the next page...

Table continued from the previous page...

EIM3-CH06	emac1_iq_lmem6_rd (72 bits)	Bank6 bits 31-0: WORD0[31:0] Bank6 bits 63-32: WORD1[31:0] Bank6 bits 71-64: WORD2[7:0]
EIM3-CH07	emac1_iq_lmem7_rd (72 bits)	Bank7 bits 31-0: WORD0[31:0] Bank7 bits 63-32: WORD1[31:0] Bank7 bits 71-64: WORD2[7:0]
EIM3-CH08	emac1_iq_lmem8_rd (72 bits)	Bank8 bits 31-0: WORD0[31:0] Bank8 bits 63-32: WORD1[31:0] Bank8 bits 71-64: WORD2[7:0]
EIM3-CH09	emac1_iq_lru_count (23 bits)	Bank9 bits 22-0: WORD0[22:0]
EIM3-CH10	emac2_iq_class_entry (288 bits)	Bank10 bits 31-0: WORD0[31:0] Bank10 bits 63-32: WORD1[31:0] Bank10 bits 95-64: WORD2[31:0] Bank10 bits 127-96: WORD3[31:0] Bank10 bits 159-128: WORD4[31:0] Bank10 bits 191-160: WORD5[31:0] Bank10 bits 223-192: WORD6[31:0] Bank10 bits 255-224: WORD7[31:0] Bank10 bits 287-256: WORD8[31:0]
EIM3-CH11	emac2_iq_lmem1_rd (72 bits)	Bank11 bits 31-0: WORD0[31:0] Bank11 bits 63-32: WORD1[31:0] Bank11 bits 71-64: WORD2[7:0]
EIM3-CH12	emac2_iq_lmem2_rd (72 bits)	Bank12 bits 31-0: WORD0[31:0] Bank12 bits 63-32: WORD1[31:0] Bank12 bits 71-64: WORD2[7:0]
EIM3-CH13	emac2_iq_lmem3_rd (72 bits)	Bank13 bits 31-0: WORD0[31:0] Bank13 bits 63-32: WORD1[31:0] Bank13 bits 71-64: WORD2[7:0]
EIM3-CH14	emac2_iq_lmem4_rd (72 bits)	Bank14 bits 31-0: WORD0[31:0] Bank14 bits 63-32: WORD1[31:0] Bank14 bits 71-64: WORD2[7:0]
EIM3-CH15	emac2_iq_lmem5_rd (72 bits)	Bank15 bits 31-0: WORD0[31:0] Bank15 bits 63-32: WORD1[31:0] Bank15 bits 71-64: WORD2[7:0]
EIM3-CH16	emac2_iq_lmem6_rd (72 bits)	Bank16 bits 31-0: WORD0[31:0] Bank16 bits 63-32: WORD1[31:0] Bank16 bits 71-64: WORD2[7:0]
EIM3-CH17	emac2_iq_lmem7_rd (72 bits)	Bank17 bits 31-0: WORD0[31:0] Bank17 bits 63-32: WORD1[31:0] Bank17 bits 71-64: WORD2[7:0]
EIM3-CH18	emac2_iq_lmem8_rd (72 bits)	Bank18 bits 31-0: WORD0[31:0] Bank18 bits 63-32: WORD1[31:0] Bank18 bits 71-64: WORD2[7:0]

Table continues on the next page...

Table continued from the previous page...

EIM3-CH19	emac2_iq_lru_count (23 bits)	Bank19 bits 22-0: WORD0[22:0]
EIM3-CH20	etgpix_rxf_rd (72 bits)	Bank20 bits 31-0: WORD0[31:0] Bank20 bits 63-32: WORD1[31:0] Bank20 bits 71-64: WORD2[7:0]
EIM3-CH21	etgpix_rxf_rd (72 bits)	Bank21 bits 31-0: WORD0[31:0] Bank21 bits 63-32: WORD1[31:0] Bank21 bits 71-64: WORD2[7:0]
EIM3-CH22	etgpix_rxf_rd (72 bits)	Bank22 bits 31-0: WORD0[31:0] Bank22 bits 63-32: WORD1[31:0] Bank22 bits 71-64: WORD2[7:0]
EIM3-CH23	hgpi1_rxf_rd (72 bits)	Bank23 bits 31-0: WORD0[31:0] Bank23 bits 63-32: WORD1[31:0] Bank23 bits 71-64: WORD2[7:0]
EIM3-CH24	hgpi1_txf_rd (72 bits)	Bank24 bits 31-0: WORD0[31:0] Bank24 bits 63-32: WORD1[31:0] Bank24 bits 71-64: WORD2[7:0]
EIM3-CH25	hif1_dma_dxr1_drx_rd (72 bits)	Bank25 bits 31-0: WORD0[31:0] Bank25 bits 63-32: WORD1[31:0] Bank25 bits 71-64: WORD2[7:0]
EIM3-CH26	hif1_dma_dxr1_dtx_rd (72 bits)	Bank26 bits 31-0: WORD0[31:0] Bank26 bits 63-32: WORD1[31:0] Bank26 bits 71-64: WORD2[7:0]
EIM3-CH27	hif1_dma_dxr2_drx_rd (72 bits)	Bank27 bits 31-0: WORD0[31:0] Bank27 bits 63-32: WORD1[31:0] Bank27 bits 71-64: WORD2[7:0]
EIM3-CH28	hif1_dma_dxr2_dtx_rd (72 bits)	Bank28 bits 31-0: WORD0[31:0] Bank28 bits 63-32: WORD1[31:0] Bank28 bits 71-64: WORD2[7:0]
EIM3-CH29	hif1_dma_dxr3_drx_rd (72 bits)	Bank29 bits 31-0: WORD0[31:0] Bank29 bits 63-32: WORD1[31:0] Bank29 bits 71-64: WORD2[7:0]
EIM3-CH30	hif1_dma_dxr3_dtx_rd (72 bits)	Bank30 bits 31-0: WORD0[31:0] Bank30 bits 63-32: WORD1[31:0] Bank30 bits 71-64: WORD2[7:0]
EIM3-CH31	hif1_dma_dxr4_drx_rd (72 bits)	Bank31 bits 31-0: WORD0[31:0] Bank31 bits 63-32: WORD1[31:0] Bank31 bits 71-64: WORD2[7:0]
EIM4-CH00	hif1_dma_dxr4_dtx_rd (72 bits)	Bank0 bits 31-0: WORD0[31:0] Bank0 bits 63-32: WORD1[31:0] Bank0 bits 71-64: WORD2[7:0]
EIM4-CH01	hif1_dma_dxr_drx_ctrl_word_rd (39 bits)	Bank1 bits 31-0: WORD0[31:0] Bank1 bits 38-32: WORD1[6:0]

Table continues on the next page...

Table continued from the previous page...

EIM4-CH02	hif1_dma_dxr_dtx_ctrl_word_rd (72 bits)	Bank2 bits 31-0: WORD0[31:0] Bank2 bits 63-32: WORD1[31:0] Bank2 bits 71-64: WORD2[7:0]
EIM4-CH03	hif1_rx_dma_bdp_rd_burst_rd (72 bits)	Bank3 bits 31-0: WORD0[31:0] Bank3 bits 63-32: WORD1[31:0] Bank3 bits 71-64: WORD2[7:0]
EIM4-CH04	hif1_rx_dma_bdp_wr_burst_rd (72 bits)	Bank4 bits 31-0: WORD0[31:0] Bank4 bits 63-32: WORD1[31:0] Bank4 bits 71-64: WORD2[7:0]
EIM4-CH05	hif1_tx_bd_ctrl_rd (72 bits)	Bank5 bits 31-0: WORD0[31:0] Bank5 bits 63-32: WORD1[31:0] Bank5 bits 71-64: WORD2[7:0]
EIM4-CH06	hif1_tx_bd_ctrl_rd (72 bits)	Bank6 bits 31-0: WORD0[31:0] Bank6 bits 63-32: WORD1[31:0] Bank6 bits 71-64: WORD2[7:0]
EIM4-CH07	hif1_tx_bd_ctrl_rd (72 bits)	Bank7 bits 31-0: WORD0[31:0] Bank7 bits 63-32: WORD1[31:0] Bank7 bits 71-64: WORD2[7:0]
EIM4-CH08	hif1_tx_dma_bdp_rd_burst_rd (72 bits)	Bank8 bits 31-0: WORD0[31:0] Bank8 bits 63-32: WORD1[31:0] Bank8 bits 71-64: WORD2[7:0]
EIM4-CH09	hif1_tx_dma_bdp_wr_burst_rd (72 bits)	Bank9 bits 31-0: WORD0[31:0] Bank9 bits 63-32: WORD1[31:0] Bank9 bits 71-64: WORD2[7:0]
EIM5-CH00	EIM_indata_ddr2npu_rdata (16 bits)	Bank0 bits 15-0: WORD0[15:0]
EIM5-CH01	EIM_indata_hdbus2upe_rdata (16 bits)	Bank1 bits 15-0: WORD0[15:0]
EIM5-CH02	EIM_indata_npu2ddr_araddr (8 bits)	Bank2 bits 7-0: WORD0[7:0]
EIM5-CH03	EIM_indata_npu2ddr_awaddr (8 bits)	Bank3 bits 7-0: WORD0[7:0]
EIM5-CH04	EIM_indata_npu2ddr_wdata (16 bits)	Bank4 bits 15-0: WORD0[15:0]
EIM5-CH05	EIM_indata_upe2hdbus_araddr (8 bits)	Bank5 bits 7-0: WORD0[7:0]
EIM5-CH06	EIM_indata_upe2hdbus_awaddr (8 bits)	Bank6 bits 7-0: WORD0[7:0]
EIM5-CH07	EIM_indata_upe2hdbus_wdata (16 bits)	Bank7 bits 15-0: WORD0[15:0]
EIM5-CH08	bcnt2 (72 bits)	Bank8 bits 31-0: WORD0[31:0] Bank8 bits 63-32: WORD1[31:0] Bank8 bits 71-64: WORD2[7:0]
EIM5-CH09	bcnt (72 bits)	Bank9 bits 31-0: WORD0[31:0] Bank9 bits 63-32: WORD1[31:0] Bank9 bits 71-64: WORD2[7:0]
EIM5-CH10	bmu2 (72 bits)	Bank10 bits 31-0: WORD0[31:0] Bank10 bits 63-32: WORD1[31:0] Bank10 bits 71-64: WORD2[7:0]

Table continues on the next page...

Table continued from the previous page...

EIM5-CH11	bmu (72 bits)	Bank11 bits 31-0: WORD0[31:0] Bank11 bits 63-32: WORD1[31:0] Bank11 bits 71-64: WORD2[7:0]
EIM5-CH12	reclaim_buf_rd (72 bits)	Bank12 bits 31-0: WORD0[31:0] Bank12 bits 63-32: WORD1[31:0] Bank12 bits 71-64: WORD2[7:0]
EIM5-CH13	teq_qfifo_0 (72 bits)	Bank13 bits 31-0: WORD0[31:0] Bank13 bits 63-32: WORD1[31:0] Bank13 bits 71-64: WORD2[7:0]
EIM5-CH14	teq_qfifo_1 (39 bits)	Bank14 bits 31-0: WORD0[31:0] Bank14 bits 38-32: WORD1[6:0]
EIM5-CH15	teq_qfifo_0 (72 bits)	Bank15 bits 31-0: WORD0[31:0] Bank15 bits 63-32: WORD1[31:0] Bank15 bits 71-64: WORD2[7:0]
EIM5-CH16	teq_qfifo_1 (39 bits)	Bank16 bits 31-0: WORD0[31:0] Bank16 bits 38-32: WORD1[6:0]
EIM5-CH17	teq_qfifo_0 (72 bits)	Bank17 bits 31-0: WORD0[31:0] Bank17 bits 63-32: WORD1[31:0] Bank17 bits 71-64: WORD2[7:0]
EIM5-CH18	teq_qfifo_1 (39 bits)	Bank18 bits 31-0: WORD0[31:0] Bank18 bits 38-32: WORD1[6:0]
EIM5-CH19	teq_qfifo_0 (72 bits)	Bank19 bits 31-0: WORD0[31:0] Bank19 bits 63-32: WORD1[31:0] Bank19 bits 71-64: WORD2[7:0]
EIM5-CH20	teq_qfifo_1 (39 bits)	Bank20 bits 31-0: WORD0[31:0] Bank20 bits 38-32: WORD1[6:0]
EIM5-CH21	teq_qfifo_0 (72 bits)	Bank21 bits 31-0: WORD0[31:0] Bank21 bits 63-32: WORD1[31:0] Bank21 bits 71-64: WORD2[7:0]
EIM5-CH22	teq_qfifo_1 (39 bits)	Bank22 bits 31-0: WORD0[31:0] Bank22 bits 38-32: WORD1[6:0]
EIM5-CH23	teq_qfifo_0 (72 bits)	Bank23 bits 31-0: WORD0[31:0] Bank23 bits 63-32: WORD1[31:0] Bank23 bits 71-64: WORD2[7:0]
EIM5-CH24	teq_qfifo_1 (39 bits)	Bank24 bits 31-0: WORD0[31:0] Bank24 bits 38-32: WORD1[6:0]
EIM5-CH25	tlite_cntx (156 bits)	Bank25 bits 31-0: WORD0[31:0] Bank25 bits 63-32: WORD1[31:0] Bank25 bits 95-64: WORD2[31:0] Bank25 bits 127-96: WORD3[31:0] Bank25 bits 155-128: WORD4[27:0]

Table continues on the next page...

Table continued from the previous page...

EIM5-CH26	tlite_cntx (78 bits)	Bank26 bits 31-0: WORD0[31:0] Bank26 bits 63-32: WORD1[31:0] Bank26 bits 77-64: WORD2[13:0]
EIM5-CH27	tlite_inq_rd (72 bits)	Bank27 bits 31-0: WORD0[31:0] Bank27 bits 63-32: WORD1[31:0] Bank27 bits 71-64: WORD2[7:0]
EIM5-CH28	upe_dram (78 bits)	Bank28 bits 31-0: WORD0[31:0] Bank28 bits 63-32: WORD1[31:0] Bank28 bits 77-64: WORD2[13:0]
EIM5-CH29	upe_dram (26 bits)	Bank29 bits 25-0: WORD0[25:0]
EIM5-CH30	upe_dram_lpe0 (52 bits)	Bank30 bits 31-0: WORD0[31:0] Bank30 bits 51-32: WORD1[19:0]
EIM5-CH31	upe_dram_lpe1 (52 bits)	Bank31 bits 31-0: WORD0[31:0] Bank31 bits 51-32: WORD1[19:0]
EIM6-CH00	upe_inq_mem (39 bits)	Bank0 bits 31-0: WORD0[31:0] Bank0 bits 38-32: WORD1[6:0]
EIM6-CH01	upe_iram (39 bits)	Bank1 bits 31-0: WORD0[31:0] Bank1 bits 38-32: WORD1[6:0]

73.2.4 EIM_LLCE channel assignments

Table 2 Targets of LLCE EIM channel's EICHDn_WORD registers

Channel	Target (bus size)	Data bits
0	LLCE Data RAM_A (39 bits)	WORD[31:0] +7 parity bits
1	LLCE Data RAM_B (39 bits)	WORD[31:0] +7 parity bits
2	Reserved	NA
3	LLCE FRPE RAM (39 bits)	WORD[31:0] +7 parity bits
4	LLCE RxPPE RAM (39 bits)	WORD[31:0] +7 parity bits
5	LLCE TxPPE RAM (39 bits)	WORD[31:0] +7 parity bits
6	LLCE DTE RAM (39 bits)	WORD[31:0]

Table continues on the next page...

Table continued from the previous page...

		+7 parity bits
7	Reserved	NA
8	Reserved	NA

73.3 Introduction

The Error Injection Module (EIM) is mainly used for diagnostic purposes. It provides a method for diagnostic coverage of internal memories (for example, system RAM, cache RAMs, and peripheral memories). See the chip-specific EIM information to determine which functional safety features are supported by this method.

73.3.1 Overview

EIM provides support for inducing single-bit and multi-bit inversions on read data when accessing peripheral RAMs. Injecting faults on memory accesses can be used to exercise the SEC-DED ECC function of the related system.

NOTE

The following diagram shows an example of EIM implementation with a 64-bit read data bus and an 8-bit checkbit bus.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39da3ab624c1

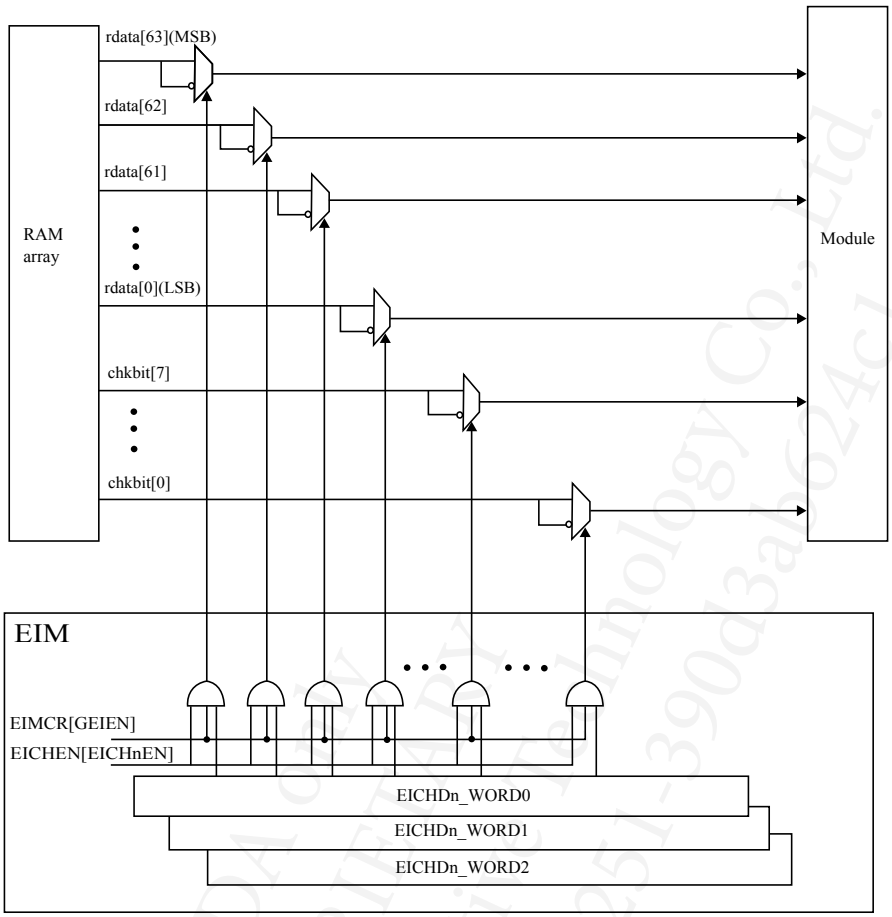


Figure 575. EIM functional block diagram (64-bit read data bus and 8-bit check bit bus)

Several memory elements are implemented within a device, which may not only be the large memory blocks (Flash and SRAM) but also smaller memories like caches, the TCD blocks, and the embedded peripheral memories. Some larger memories may actually be built from multiple memory elements, dependent on their size or function. Each of these memory elements implements its own control logic, the memory controller, that performs the accesses to the actual memory, the memory array. An EIM channel is associated with a memory controller and provides the capability to alter one or multiple signals in the read access path from the corresponding memory array(s). Only memory controllers controlling a safety related memory may be associated with an EIM channel.

73.3.2 Features

The EIM includes these features:

- Supports 31 error injection channels
- Protection against accidental enable and reconfiguration error injection function via two-stage enable mechanism

73.4 EIM register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.
  - Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 576. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	256	255-224	223-192	191-160
1	256	255-224	223-192	191-160
2	256	255-224	223-192	191-160
3	256	255-224	223-192	191-160
4	256	255-224	223-192	191-160
5	256	255-224	223-192	191-160
6	256	255-224	223-192	191-160
7	256	255-224	223-192	191-160
8	64	63-32	31-0	—
9	64	63-32	31-0	—
10	64	63-32	31-0	—

Table continues on the next page...

Table 576. Error injection channel descriptor: DATA_MASK details (continued)

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
11	44	43-32	31-0	—
12	128	127-96	95-64	63-32
13	104	103-96	95-64	63-32
14	128	127-96	95-64	63-32
15	128	127-96	95-64	63-32
16	64	63-32	31-0	—
17	44	43-32	31-0	—
18	128	127-96	95-64	63-32
19	104	103-96	95-64	63-32
20	128	127-96	95-64	63-32
21	128	127-96	95-64	63-32
22	64	63-32	31-0	—
23	44	43-32	31-0	—
24	128	127-96	95-64	63-32
25	104	103-96	95-64	63-32
26	128	127-96	95-64	63-32
27	128	127-96	95-64	63-32
28	90	89-64	63-32	31-0
29	20	19-0	—	—
30	18	17-0	—	—

Table 577. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
0	159-128	127-96	95-64	63-32	31-0
1	159-128	127-96	95-64	63-32	31-0

Table continues on the next page...

Table 577. DATA_MASK bit: Channel-word mapping (continued)

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
2	159-128	127-96	95-64	63-32	31-0
3	159-128	127-96	95-64	63-32	31-0
4	159-128	127-96	95-64	63-32	31-0
5	159-128	127-96	95-64	63-32	31-0
6	159-128	127-96	95-64	63-32	31-0
7	159-128	127-96	95-64	63-32	31-0
12	31-0	—	—	—	—
13	31-0	—	—	—	—
14	31-0	—	—	—	—
15	31-0	—	—	—	—
18	31-0	—	—	—	—
19	31-0	—	—	—	—
20	31-0	—	—	—	—
21	31-0	—	—	—	—
24	31-0	—	—	—	—
25	31-0	—	—	—	—
26	31-0	—	—	—	—
27	31-0	—	—	—	—

73.4.1 EIM memory map

EIM base address: 4030_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
100h	Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
10Ch	Error Injection Channel Descriptor 0, Word3 (EICHD0_WORD3)	32	RW	0000_0000h
110h	Error Injection Channel Descriptor 0, Word4 (EICHD0_WORD4)	32	RW	0000_0000h
114h	Error Injection Channel Descriptor 0, Word5 (EICHD0_WORD5)	32	RW	0000_0000h
118h	Error Injection Channel Descriptor 0, Word6 (EICHD0_WORD6)	32	RW	0000_0000h
11Ch	Error Injection Channel Descriptor 0, Word7 (EICHD0_WORD7)	32	RW	0000_0000h
120h	Error Injection Channel Descriptor 0, Word8 (EICHD0_WORD8)	32	RW	0000_0000h
140h	Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h
14Ch	Error Injection Channel Descriptor 1, Word3 (EICHD1_WORD3)	32	RW	0000_0000h
150h	Error Injection Channel Descriptor 1, Word4 (EICHD1_WORD4)	32	RW	0000_0000h
154h	Error Injection Channel Descriptor 1, Word5 (EICHD1_WORD5)	32	RW	0000_0000h
158h	Error Injection Channel Descriptor 1, Word6 (EICHD1_WORD6)	32	RW	0000_0000h
15Ch	Error Injection Channel Descriptor 1, Word7 (EICHD1_WORD7)	32	RW	0000_0000h
160h	Error Injection Channel Descriptor 1, Word8 (EICHD1_WORD8)	32	RW	0000_0000h
180h	Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
18Ch	Error Injection Channel Descriptor 2, Word3 (EICHD2_WORD3)	32	RW	0000_0000h
190h	Error Injection Channel Descriptor 2, Word4 (EICHD2_WORD4)	32	RW	0000_0000h
194h	Error Injection Channel Descriptor 2, Word5 (EICHD2_WORD5)	32	RW	0000_0000h
198h	Error Injection Channel Descriptor 2, Word6 (EICHD2_WORD6)	32	RW	0000_0000h
19Ch	Error Injection Channel Descriptor 2, Word7 (EICHD2_WORD7)	32	RW	0000_0000h
1A0h	Error Injection Channel Descriptor 2, Word8 (EICHD2_WORD8)	32	RW	0000_0000h
1C0h	Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
1CCh	Error Injection Channel Descriptor 3, Word3 (EICHD3_WORD3)	32	RW	0000_0000h
1D0h	Error Injection Channel Descriptor 3, Word4 (EICHD3_WORD4)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
1D4h	Error Injection Channel Descriptor 3, Word5 (EICHD3_WORD5)	32	RW	0000_0000h
1D8h	Error Injection Channel Descriptor 3, Word6 (EICHD3_WORD6)	32	RW	0000_0000h
1DCh	Error Injection Channel Descriptor 3, Word7 (EICHD3_WORD7)	32	RW	0000_0000h
1E0h	Error Injection Channel Descriptor 3, Word8 (EICHD3_WORD8)	32	RW	0000_0000h
200h	Error Injection Channel Descriptor 4, Word0 (EICHD4_WORD0)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
208h	Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)	32	RW	0000_0000h
20Ch	Error Injection Channel Descriptor 4, Word3 (EICHD4_WORD3)	32	RW	0000_0000h
210h	Error Injection Channel Descriptor 4, Word4 (EICHD4_WORD4)	32	RW	0000_0000h
214h	Error Injection Channel Descriptor 4, Word5 (EICHD4_WORD5)	32	RW	0000_0000h
218h	Error Injection Channel Descriptor 4, Word6 (EICHD4_WORD6)	32	RW	0000_0000h
21Ch	Error Injection Channel Descriptor 4, Word7 (EICHD4_WORD7)	32	RW	0000_0000h
220h	Error Injection Channel Descriptor 4, Word8 (EICHD4_WORD8)	32	RW	0000_0000h
240h	Error Injection Channel Descriptor 5, Word0 (EICHD5_WORD0)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
248h	Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)	32	RW	0000_0000h
24Ch	Error Injection Channel Descriptor 5, Word3 (EICHD5_WORD3)	32	RW	0000_0000h
250h	Error Injection Channel Descriptor 5, Word4 (EICHD5_WORD4)	32	RW	0000_0000h
254h	Error Injection Channel Descriptor 5, Word5 (EICHD5_WORD5)	32	RW	0000_0000h
258h	Error Injection Channel Descriptor 5, Word6 (EICHD5_WORD6)	32	RW	0000_0000h
25Ch	Error Injection Channel Descriptor 5, Word7 (EICHD5_WORD7)	32	RW	0000_0000h
260h	Error Injection Channel Descriptor 5, Word8 (EICHD5_WORD8)	32	RW	0000_0000h
280h	Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0)	32	RW	0000_0000h
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h
288h	Error Injection Channel Descriptor 6, Word2 (EICHD6_WORD2)	32	RW	0000_0000h
28Ch	Error Injection Channel Descriptor 6, Word3 (EICHD6_WORD3)	32	RW	0000_0000h
290h	Error Injection Channel Descriptor 6, Word4 (EICHD6_WORD4)	32	RW	0000_0000h
294h	Error Injection Channel Descriptor 6, Word5 (EICHD6_WORD5)	32	RW	0000_0000h
298h	Error Injection Channel Descriptor 6, Word6 (EICHD6_WORD6)	32	RW	0000_0000h
29Ch	Error Injection Channel Descriptor 6, Word7 (EICHD6_WORD7)	32	RW	0000_0000h
2A0h	Error Injection Channel Descriptor 6, Word8 (EICHD6_WORD8)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
2C0h	Error Injection Channel Descriptor 7, Word0 (EICHD7_WORD0)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
2C8h	Error Injection Channel Descriptor 7, Word2 (EICHD7_WORD2)	32	RW	0000_0000h
2CCh	Error Injection Channel Descriptor 7, Word3 (EICHD7_WORD3)	32	RW	0000_0000h
2D0h	Error Injection Channel Descriptor 7, Word4 (EICHD7_WORD4)	32	RW	0000_0000h
2D4h	Error Injection Channel Descriptor 7, Word5 (EICHD7_WORD5)	32	RW	0000_0000h
2D8h	Error Injection Channel Descriptor 7, Word6 (EICHD7_WORD6)	32	RW	0000_0000h
2DCh	Error Injection Channel Descriptor 7, Word7 (EICHD7_WORD7)	32	RW	0000_0000h
2E0h	Error Injection Channel Descriptor 7, Word8 (EICHD7_WORD8)	32	RW	0000_0000h
300h	Error Injection Channel Descriptor 8, Word0 (EICHD8_WORD0)	32	RW	0000_0000h
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
308h	Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)	32	RW	0000_0000h
340h	Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
348h	Error Injection Channel Descriptor 9, Word2 (EICHD9_WORD2)	32	RW	0000_0000h
380h	Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0)	32	RW	0000_0000h
384h	Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)	32	RW	0000_0000h
388h	Error Injection Channel Descriptor 10, Word2 (EICHD10_WORD2)	32	RW	0000_0000h
3C0h	Error Injection Channel Descriptor 11, Word0 (EICHD11_WORD0)	32	RW	0000_0000h
3C4h	Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)	32	RW	0000_0000h
3C8h	Error Injection Channel Descriptor 11, Word2 (EICHD11_WORD2)	32	RW	0000_0000h
400h	Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0)	32	RW	0000_0000h
404h	Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)	32	RW	0000_0000h
408h	Error Injection Channel Descriptor 12, Word2 (EICHD12_WORD2)	32	RW	0000_0000h
40Ch	Error Injection Channel Descriptor 12, Word3 (EICHD12_WORD3)	32	RW	0000_0000h
410h	Error Injection Channel Descriptor 12, Word4 (EICHD12_WORD4)	32	RW	0000_0000h
440h	Error Injection Channel Descriptor 13, Word0 (EICHD13_WORD0)	32	RW	0000_0000h
444h	Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)	32	RW	0000_0000h
448h	Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)	32	RW	0000_0000h
44Ch	Error Injection Channel Descriptor 13, Word3 (EICHD13_WORD3)	32	RW	0000_0000h
450h	Error Injection Channel Descriptor 13, Word4 (EICHD13_WORD4)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
480h	Error Injection Channel Descriptor 14, Word0 (EICHD14_WORD0)	32	RW	0000_0000h
484h	Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)	32	RW	0000_0000h
488h	Error Injection Channel Descriptor 14, Word2 (EICHD14_WORD2)	32	RW	0000_0000h
48Ch	Error Injection Channel Descriptor 14, Word3 (EICHD14_WORD3)	32	RW	0000_0000h
490h	Error Injection Channel Descriptor 14, Word4 (EICHD14_WORD4)	32	RW	0000_0000h
4C0h	Error Injection Channel Descriptor 15, Word0 (EICHD15_WORD0)	32	RW	0000_0000h
4C4h	Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)	32	RW	0000_0000h
4C8h	Error Injection Channel Descriptor 15, Word2 (EICHD15_WORD2)	32	RW	0000_0000h
4CCh	Error Injection Channel Descriptor 15, Word3 (EICHD15_WORD3)	32	RW	0000_0000h
4D0h	Error Injection Channel Descriptor 15, Word4 (EICHD15_WORD4)	32	RW	0000_0000h
500h	Error Injection Channel Descriptor 16, Word0 (EICHD16_WORD0)	32	RW	0000_0000h
504h	Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)	32	RW	0000_0000h
508h	Error Injection Channel Descriptor 16, Word2 (EICHD16_WORD2)	32	RW	0000_0000h
540h	Error Injection Channel Descriptor 17, Word0 (EICHD17_WORD0)	32	RW	0000_0000h
544h	Error Injection Channel Descriptor 17, Word1 (EICHD17_WORD1)	32	RW	0000_0000h
548h	Error Injection Channel Descriptor 17, Word2 (EICHD17_WORD2)	32	RW	0000_0000h
580h	Error Injection Channel Descriptor 18, Word0 (EICHD18_WORD0)	32	RW	0000_0000h
584h	Error Injection Channel Descriptor 18, Word1 (EICHD18_WORD1)	32	RW	0000_0000h
588h	Error Injection Channel Descriptor 18, Word2 (EICHD18_WORD2)	32	RW	0000_0000h
58Ch	Error Injection Channel Descriptor 18, Word3 (EICHD18_WORD3)	32	RW	0000_0000h
590h	Error Injection Channel Descriptor 18, Word4 (EICHD18_WORD4)	32	RW	0000_0000h
5C0h	Error Injection Channel Descriptor 19, Word0 (EICHD19_WORD0)	32	RW	0000_0000h
5C4h	Error Injection Channel Descriptor 19, Word1 (EICHD19_WORD1)	32	RW	0000_0000h
5C8h	Error Injection Channel Descriptor 19, Word2 (EICHD19_WORD2)	32	RW	0000_0000h
5CCh	Error Injection Channel Descriptor 19, Word3 (EICHD19_WORD3)	32	RW	0000_0000h
5D0h	Error Injection Channel Descriptor 19, Word4 (EICHD19_WORD4)	32	RW	0000_0000h
600h	Error Injection Channel Descriptor 20, Word0 (EICHD20_WORD0)	32	RW	0000_0000h
604h	Error Injection Channel Descriptor 20, Word1 (EICHD20_WORD1)	32	RW	0000_0000h
608h	Error Injection Channel Descriptor 20, Word2 (EICHD20_WORD2)	32	RW	0000_0000h
60Ch	Error Injection Channel Descriptor 20, Word3 (EICHD20_WORD3)	32	RW	0000_0000h
610h	Error Injection Channel Descriptor 20, Word4 (EICHD20_WORD4)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
640h	Error Injection Channel Descriptor 21, Word0 (EICHD21_WORD0)	32	RW	0000_0000h
644h	Error Injection Channel Descriptor 21, Word1 (EICHD21_WORD1)	32	RW	0000_0000h
648h	Error Injection Channel Descriptor 21, Word2 (EICHD21_WORD2)	32	RW	0000_0000h
64Ch	Error Injection Channel Descriptor 21, Word3 (EICHD21_WORD3)	32	RW	0000_0000h
650h	Error Injection Channel Descriptor 21, Word4 (EICHD21_WORD4)	32	RW	0000_0000h
680h	Error Injection Channel Descriptor 22, Word0 (EICHD22_WORD0)	32	RW	0000_0000h
684h	Error Injection Channel Descriptor 22, Word1 (EICHD22_WORD1)	32	RW	0000_0000h
688h	Error Injection Channel Descriptor 22, Word2 (EICHD22_WORD2)	32	RW	0000_0000h
6C0h	Error Injection Channel Descriptor 23, Word0 (EICHD23_WORD0)	32	RW	0000_0000h
6C4h	Error Injection Channel Descriptor 23, Word1 (EICHD23_WORD1)	32	RW	0000_0000h
6C8h	Error Injection Channel Descriptor 23, Word2 (EICHD23_WORD2)	32	RW	0000_0000h
700h	Error Injection Channel Descriptor 24, Word0 (EICHD24_WORD0)	32	RW	0000_0000h
704h	Error Injection Channel Descriptor 24, Word1 (EICHD24_WORD1)	32	RW	0000_0000h
708h	Error Injection Channel Descriptor 24, Word2 (EICHD24_WORD2)	32	RW	0000_0000h
70Ch	Error Injection Channel Descriptor 24, Word3 (EICHD24_WORD3)	32	RW	0000_0000h
710h	Error Injection Channel Descriptor 24, Word4 (EICHD24_WORD4)	32	RW	0000_0000h
740h	Error Injection Channel Descriptor 25, Word0 (EICHD25_WORD0)	32	RW	0000_0000h
744h	Error Injection Channel Descriptor 25, Word1 (EICHD25_WORD1)	32	RW	0000_0000h
748h	Error Injection Channel Descriptor 25, Word2 (EICHD25_WORD2)	32	RW	0000_0000h
74Ch	Error Injection Channel Descriptor 25, Word3 (EICHD25_WORD3)	32	RW	0000_0000h
750h	Error Injection Channel Descriptor 25, Word4 (EICHD25_WORD4)	32	RW	0000_0000h
780h	Error Injection Channel Descriptor 26, Word0 (EICHD26_WORD0)	32	RW	0000_0000h
784h	Error Injection Channel Descriptor 26, Word1 (EICHD26_WORD1)	32	RW	0000_0000h
788h	Error Injection Channel Descriptor 26, Word2 (EICHD26_WORD2)	32	RW	0000_0000h
78Ch	Error Injection Channel Descriptor 26, Word3 (EICHD26_WORD3)	32	RW	0000_0000h
790h	Error Injection Channel Descriptor 26, Word4 (EICHD26_WORD4)	32	RW	0000_0000h
7C0h	Error Injection Channel Descriptor 27, Word0 (EICHD27_WORD0)	32	RW	0000_0000h
7C4h	Error Injection Channel Descriptor 27, Word1 (EICHD27_WORD1)	32	RW	0000_0000h
7C8h	Error Injection Channel Descriptor 27, Word2 (EICHD27_WORD2)	32	RW	0000_0000h
7CCh	Error Injection Channel Descriptor 27, Word3 (EICHD27_WORD3)	32	RW	0000_0000h
7D0h	Error Injection Channel Descriptor 27, Word4 (EICHD27_WORD4)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
804h	Error Injection Channel Descriptor 28, Word1 (EICHD28_WORD1)	32	RW	0000_0000h
808h	Error Injection Channel Descriptor 28, Word2 (EICHD28_WORD2)	32	RW	0000_0000h
80Ch	Error Injection Channel Descriptor 28, Word3 (EICHD28_WORD3)	32	RW	0000_0000h
844h	Error Injection Channel Descriptor 29, Word1 (EICHD29_WORD1)	32	RW	0000_0000h
884h	Error Injection Channel Descriptor 30, Word1 (EICHD30_WORD1)	32	RW	0000_0000h

73.4.2 Error Injection Module Configuration Register (EIMCR)

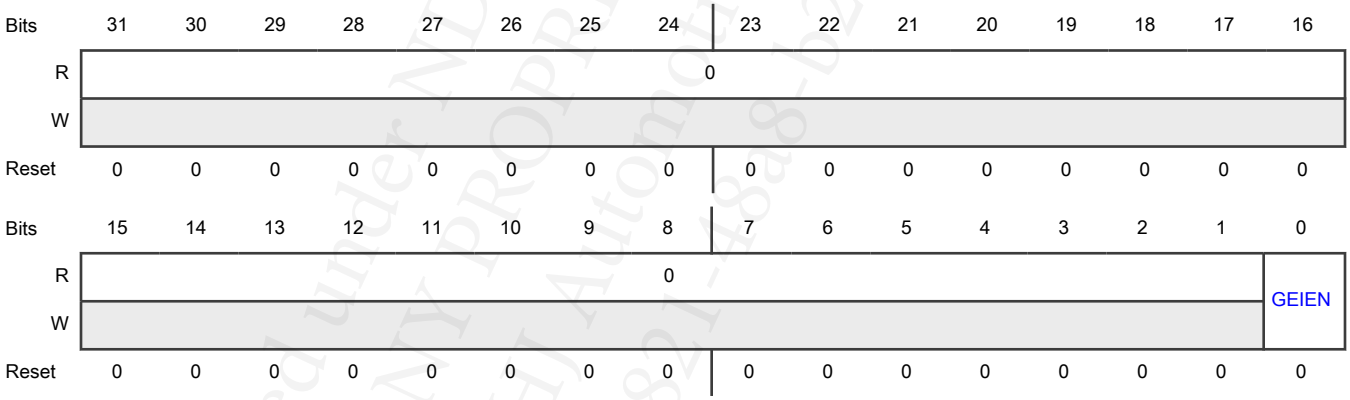
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled

73.4.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0	EICH1	EICH2	EICH3	EICH4	EICH5	EICH6	EICH7	EICH8	EICH9	EICH1	EICH1	EICH1	EICH1	EICH1	EICH1
W	EN	EN	EN	EN	EN	EN	EN	EN	EN	EN	0EN	1EN	2EN	3EN	4EN	5EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EICH1	EICH1	EICH1	EICH1	EICH2	EICH2	EICH2	EICH2	EICH2	EICH2	EICH2	EICH2	EICH2	EICH2	EICH3	0
W	6EN	7EN	8EN	9EN	0EN	1EN	2EN	3EN	4EN	5EN	6EN	7EN	8EN	9EN	0EN	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7

Table continues on the next page...

Table continued from the previous page...

Field	Function
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9
21 EICH10EN	Error Injection Channel 10 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 10 1b - Error injection is enabled on Error Injection Channel 10
20 EICH11EN	Error Injection Channel 11 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 11 1b - Error injection is enabled on Error Injection Channel 11
19 EICH12EN	Error Injection Channel 12 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 12 1b - Error injection is enabled on Error Injection Channel 12
18 EICH13EN	Error Injection Channel 13 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 13 1b - Error injection is enabled on Error Injection Channel 13
17 EICH14EN	Error Injection Channel 14 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 14 1b - Error injection is enabled on Error Injection Channel 14
16 EICH15EN	Error Injection Channel 15 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 15 1b - Error injection is enabled on Error Injection Channel 15
15 EICH16EN	Error Injection Channel 16 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 16 1b - Error injection is enabled on Error Injection Channel 16
14 EICH17EN	Error Injection Channel 17 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 17 1b - Error injection is enabled on Error Injection Channel 17
13 EICH18EN	Error Injection Channel 18 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 18 1b - Error injection is enabled on Error Injection Channel 18

Table continues on the next page...

Table continued from the previous page...

Field	Function
12 EICH19EN	Error Injection Channel 19 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 19 1b - Error injection is enabled on Error Injection Channel 19
11 EICH20EN	Error Injection Channel 20 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 20 1b - Error injection is enabled on Error Injection Channel 20
10 EICH21EN	Error Injection Channel 21 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 21 1b - Error injection is enabled on Error Injection Channel 21
9 EICH22EN	Error Injection Channel 22 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 22 1b - Error injection is enabled on Error Injection Channel 22
8 EICH23EN	Error Injection Channel 23 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 23 1b - Error injection is enabled on Error Injection Channel 23
7 EICH24EN	Error Injection Channel 24 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 24 1b - Error injection is enabled on Error Injection Channel 24
6 EICH25EN	Error Injection Channel 25 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 25 1b - Error injection is enabled on Error Injection Channel 25
5 EICH26EN	Error Injection Channel 26 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 26 1b - Error injection is enabled on Error Injection Channel 26
4 EICH27EN	Error Injection Channel 27 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 27 1b - Error injection is enabled on Error Injection Channel 27
3 EICH28EN	Error Injection Channel 28 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 28 1b - Error injection is enabled on Error Injection Channel 28
2 EICH29EN	Error Injection Channel 29 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 29 1b - Error injection is enabled on Error Injection Channel 29

Table continues on the next page...

Table continued from the previous page...

Field	Function
1 EICH30EN	Error Injection Channel 30 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 30 1b - Error injection is enabled on Error Injection Channel 30
0 —	Reserved

73.4.4 Error Injection Channel Descriptor n, Word0 (EICHD0_WORD0 - EICHD7_WORD0)

Offset

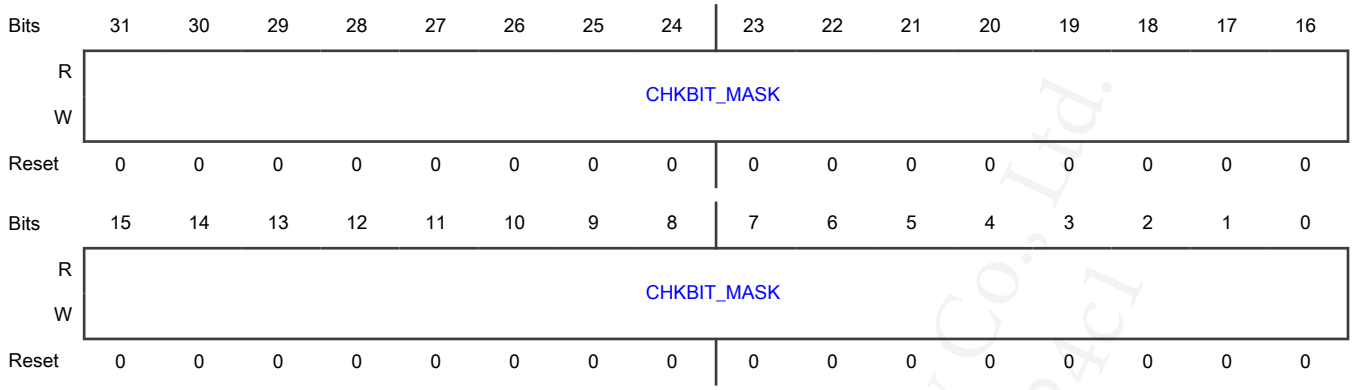
Register	Offset
EICHD0_WORD0	100h
EICHD1_WORD0	140h
EICHD2_WORD0	180h
EICHD3_WORD0	1C0h
EICHD4_WORD0	200h
EICHD5_WORD0	240h
EICHD6_WORD0	280h
EICHD7_WORD0	2C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462c-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[31:0] (32 bits wide), CHKBIT_MASK[31] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.

73.4.5 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD10_WORD1)

Offset

Register	Offset
EICHD0_WORD1	104h
EICHD1_WORD1	144h
EICHD2_WORD1	184h
EICHD3_WORD1	1C4h
EICHD4_WORD1	204h
EICHD5_WORD1	244h
EICHD6_WORD1	284h
EICHD7_WORD1	2C4h

Table continues on the next page...

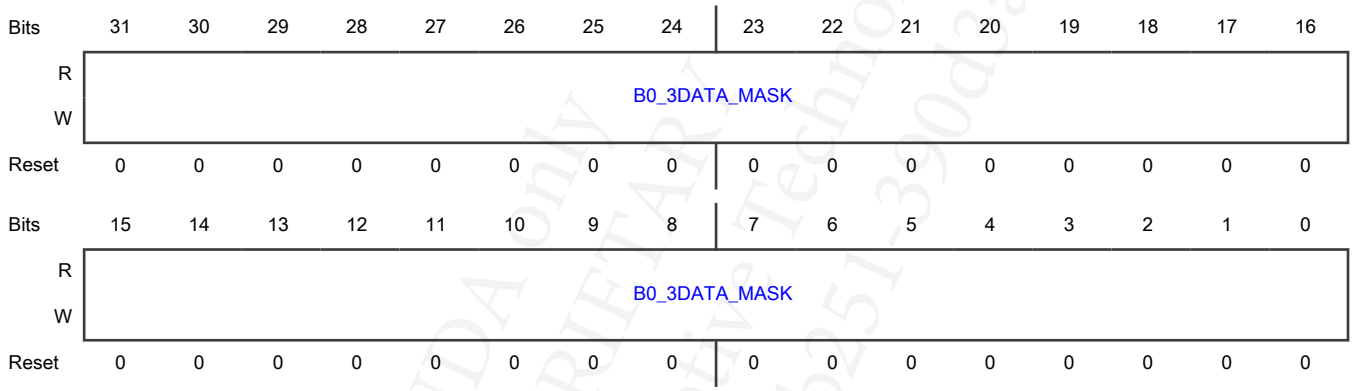
Table continued from the previous page...

Register	Offset
EICHD8_WORD1	304h
EICHD9_WORD1	344h
EICHD10_WORD1	384h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing ECU Automotive Technology Co., Ltd.
c8c462e-1482-48a8-b251-390da3ab6fc1

73.4.6 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD28_WORD2)

Offset

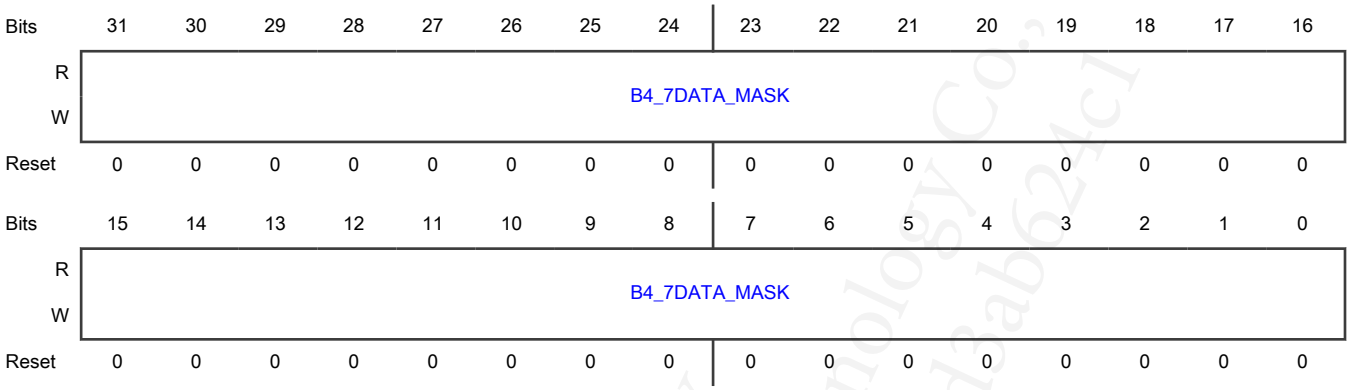
Register	Offset
EICHD0_WORD2	108h
EICHD1_WORD2	148h
EICHD2_WORD2	188h
EICHD3_WORD2	1C8h
EICHD4_WORD2	208h
EICHD5_WORD2	248h
EICHD6_WORD2	288h
EICHD7_WORD2	2C8h
EICHD8_WORD2	308h
EICHD9_WORD2	348h
EICHD10_WORD2	388h
EICHD11_WORD2	3C8h
EICHD12_WORD2	408h
EICHD13_WORD2	448h
EICHD14_WORD2	488h
EICHD15_WORD2	4C8h
EICHD16_WORD2	508h
EICHD17_WORD2	548h
EICHD18_WORD2	588h
EICHD19_WORD2	5C8h
EICHD20_WORD2	608h
EICHD21_WORD2	648h
EICHD22_WORD2	688h
EICHD23_WORD2	6C8h
EICHD24_WORD2	708h
EICHD25_WORD2	748h
EICHD26_WORD2	788h
EICHD27_WORD2	7C8h
EICHD28_WORD2	808h

Confidential under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Bytes 4-7 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

73.4.7 Error Injection Channel Descriptor n, Word3 (EICHD0_WORD3 - EICHD28_WORD3)

Offset

Register	Offset
EICHD0_WORD3	10Ch
EICHD1_WORD3	14Ch
EICHD2_WORD3	18Ch
EICHD3_WORD3	1CCh
EICHD4_WORD3	20Ch
EICHD5_WORD3	24Ch

Table continues on the next page...

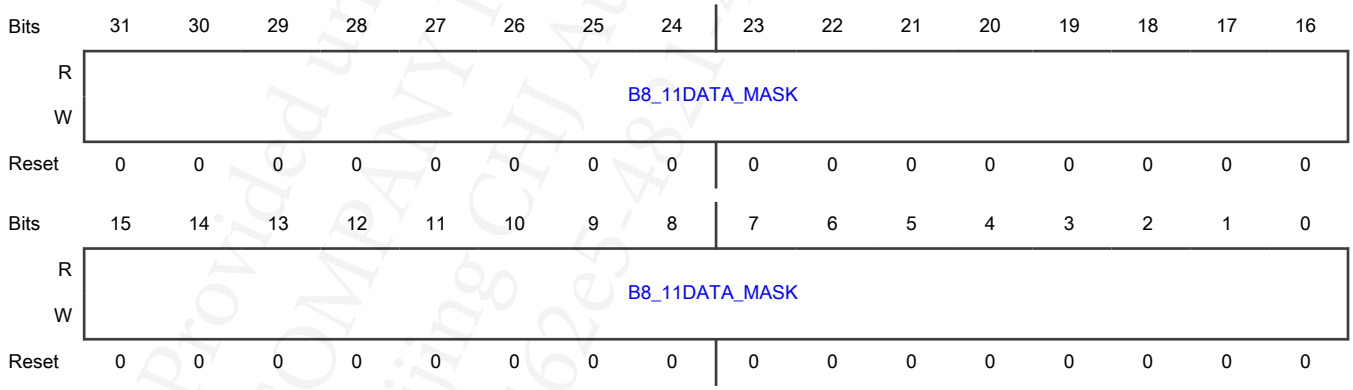
Table continued from the previous page...

Register	Offset
EICHD6_WORD3	28Ch
EICHD7_WORD3	2CCh
EICHD12_WORD3	40Ch
EICHD13_WORD3	44Ch
EICHD14_WORD3	48Ch
EICHD15_WORD3	4CCh
EICHD18_WORD3	58Ch
EICHD19_WORD3	5CCh
EICHD20_WORD3	60Ch
EICHD21_WORD3	64Ch
EICHD24_WORD3	70Ch
EICHD25_WORD3	74Ch
EICHD26_WORD3	78Ch
EICHD27_WORD3	7CCh
EICHD28_WORD3	80Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to bytes 8–11 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482f-4825-1-390d3ab624c1

Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Bytes 8-11 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 8-11 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 8-11 on the read data bus is inverted.

73.4.8 Error Injection Channel Descriptor n, Word4 (EICHD0_WORD4 - EICHD27_WORD4)

Offset

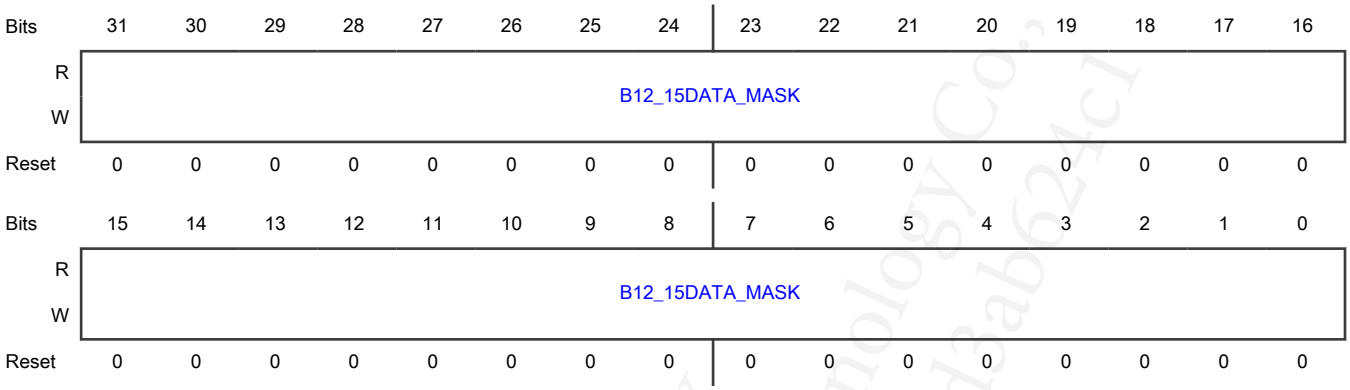
Register	Offset
EICHD0_WORD4	110h
EICHD1_WORD4	150h
EICHD2_WORD4	190h
EICHD3_WORD4	1D0h
EICHD4_WORD4	210h
EICHD5_WORD4	250h
EICHD6_WORD4	290h
EICHD7_WORD4	2D0h
EICHD12_WORD4	410h
EICHD13_WORD4	450h
EICHD14_WORD4	490h
EICHD15_WORD4	4D0h
EICHD18_WORD4	590h
EICHD19_WORD4	5D0h
EICHD20_WORD4	610h
EICHD21_WORD4	650h
EICHD24_WORD4	710h
EICHD25_WORD4	750h
EICHD26_WORD4	790h
EICHD27_WORD4	7D0h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3b624c1

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to bytes 12–15 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Bytes 12-15 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 12-15 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 12-15 on the read data bus is inverted.

73.4.9 Error Injection Channel Descriptor n, Word5 (EICHD0_WORD5 - EICHD7_WORD5)

Offset

Register	Offset
EICHD0_WORD5	114h
EICHD1_WORD5	154h
EICHD2_WORD5	194h
EICHD3_WORD5	1D4h
EICHD4_WORD5	214h
EICHD5_WORD5	254h

Table continues on the next page...

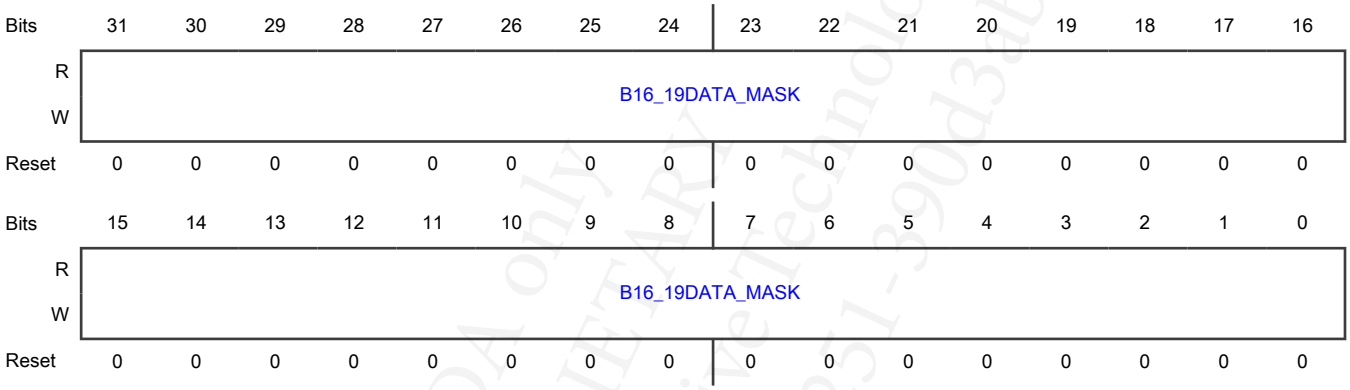
Table continued from the previous page...

Register	Offset
EICHD6_WORD5	294h
EICHD7_WORD5	2D4h

Function

The sixth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B16_19DATA_MASK correspond to bytes 16–19 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B16_19DATA_MASK	Data Mask Bytes 16-19 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B16_19DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 16-19 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 16-19 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e2-4e01-4848-b251-390d3ab62

73.4.10 Error Injection Channel Descriptor n, Word6 (EICHD0_WORD6 - EICHD7_WORD6)

Offset

Register	Offset
EICHD0_WORD6	118h
EICHD1_WORD6	158h
EICHD2_WORD6	198h
EICHD3_WORD6	1D8h
EICHD4_WORD6	218h
EICHD5_WORD6	258h
EICHD6_WORD6	298h
EICHD7_WORD6	2D8h

Function

The seventh word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B20_23DATA_MASK correspond to bytes 20–23 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B20_23DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B20_23DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B20_23DATA_MASK	Data Mask Bytes 20-23 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B20_23DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details.

Table continues on the next page...

Field	Function
	0b - The corresponding bit of bytes 20-23 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 20-23 on the read data bus is inverted.

73.4.11 Error Injection Channel Descriptor n, Word7 (EICHD0_WORD7 - EICHD7_WORD7)

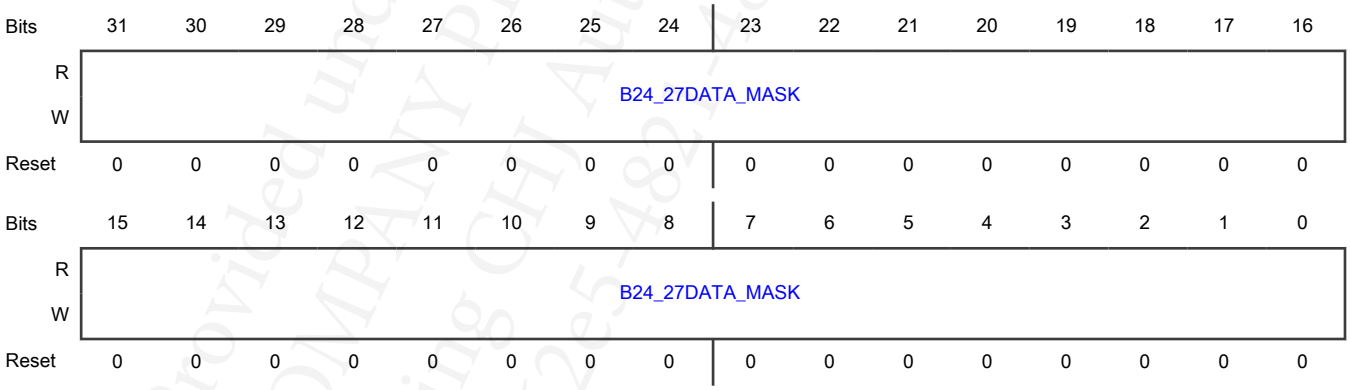
Offset

Register	Offset
EICHD0_WORD7	11Ch
EICHD1_WORD7	15Ch
EICHD2_WORD7	19Ch
EICHD3_WORD7	1DCh
EICHD4_WORD7	21Ch
EICHD5_WORD7	25Ch
EICHD6_WORD7	29Ch
EICHD7_WORD7	2DCh

Function

The eighth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B24_27DATA_MASK correspond to bytes 24–27 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482f-48a8-9251-390d3ab624c1

Fields

Field	Function
31-0 B24_27DATA_MASK	Data Mask Bytes 24-27 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B24_27DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 24-27 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 24-27 on the read data bus is inverted.

73.4.12 Error Injection Channel Descriptor n, Word8 (EICHD0_WORD8 - EICHD7_WORD8)

Offset

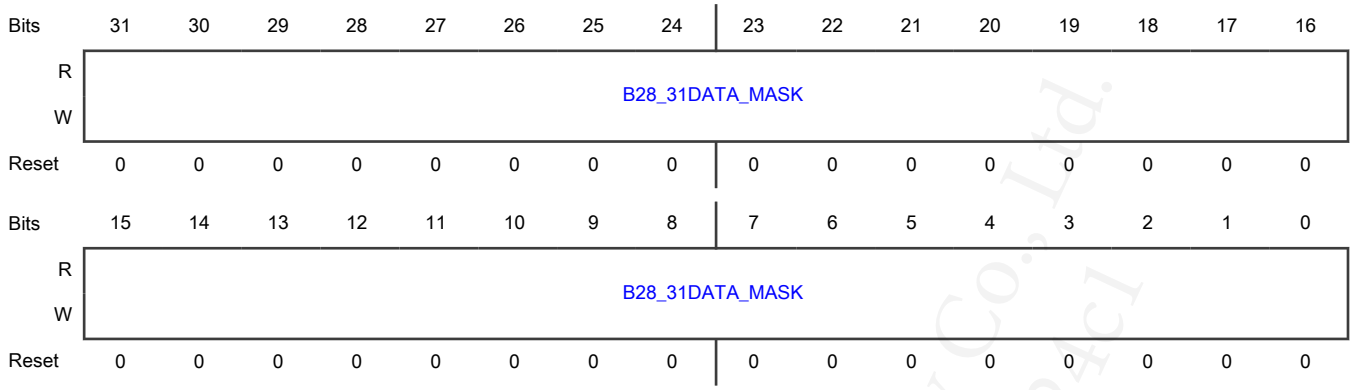
Register	Offset
EICHD0_WORD8	120h
EICHD1_WORD8	160h
EICHD2_WORD8	1A0h
EICHD3_WORD8	1E0h
EICHD4_WORD8	220h
EICHD5_WORD8	260h
EICHD6_WORD8	2A0h
EICHD7_WORD8	2E0h

Function

The ninth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B28_31DATA_MASK correspond to bytes 28–31 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing C... Automotive Technology Co., Ltd.
c8c462e5-4... -48a8-b251-390d3...

Diagram



Fields

Field	Function
31-0 B28_31DATA_MASK	Data Mask Bytes 28-31 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B28_31DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 28-31 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 28-31 on the read data bus is inverted.

73.4.13 Error Injection Channel Descriptor n, Word0 (EICHD8_WORD0 - EICHD9_WORD0)

Offset

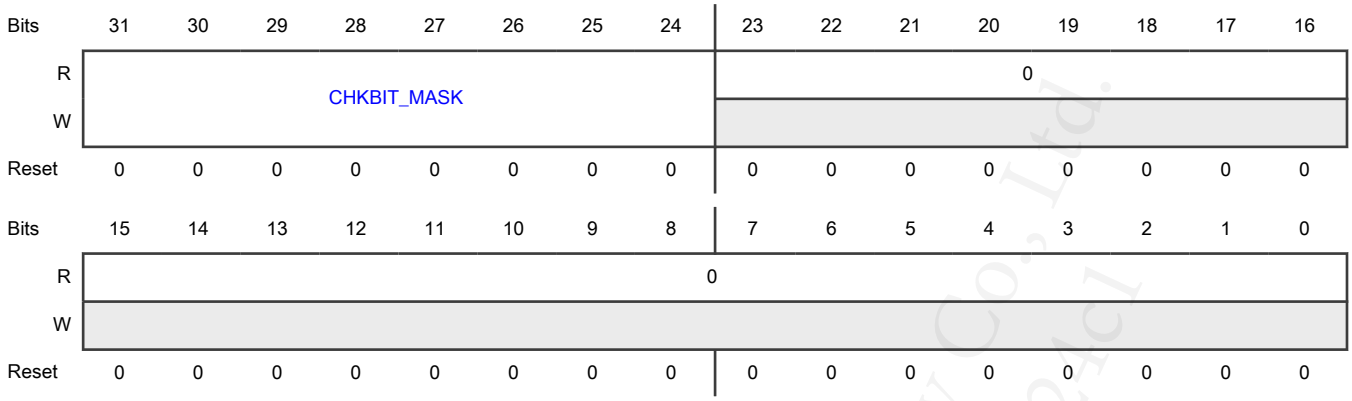
Register	Offset
EICHD8_WORD0	300h
EICHD9_WORD0	340h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e-4821-48a8-b2c1-390d3ab624c1

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.4.14 Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0)

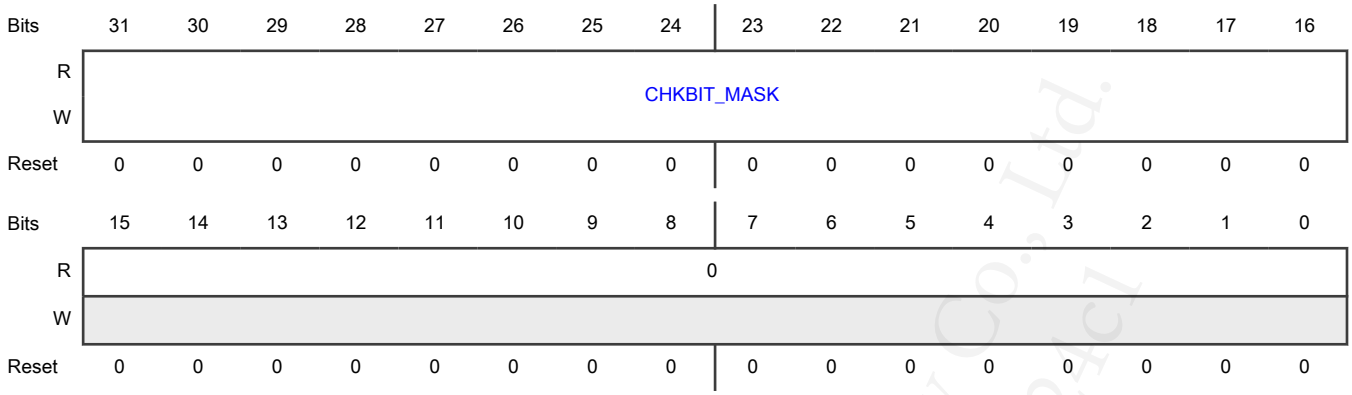
Offset

Register	Offset
EICHD10_WORD0	380h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

73.4.15 Error Injection Channel Descriptor 11, Word0 (EICHD11_WORD0)

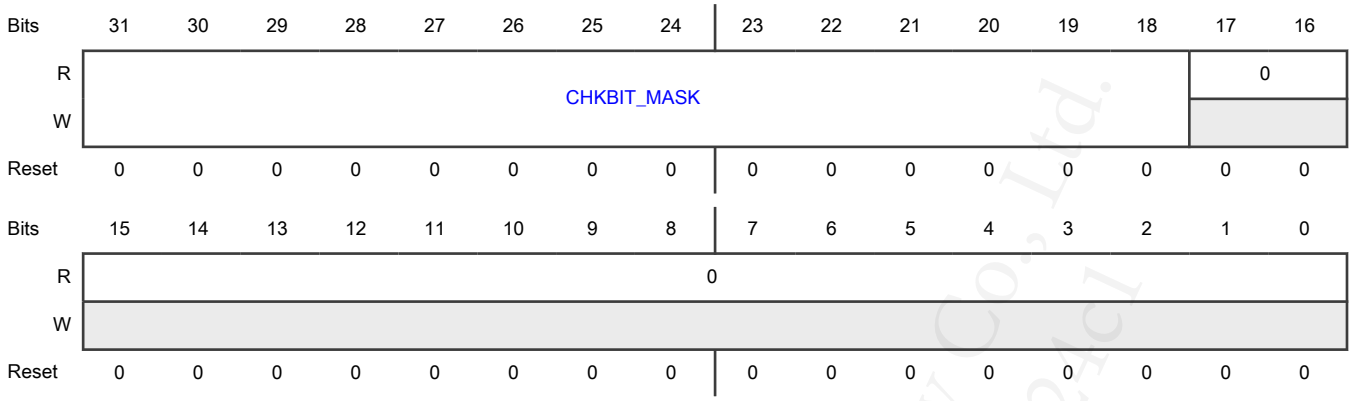
Offset

Register	Offset
EICHD11_WORD0	3C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-18 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[13:0] (14 bits wide), CHKBIT_MASK[13] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
17-0 —	Reserved

73.4.16 Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)

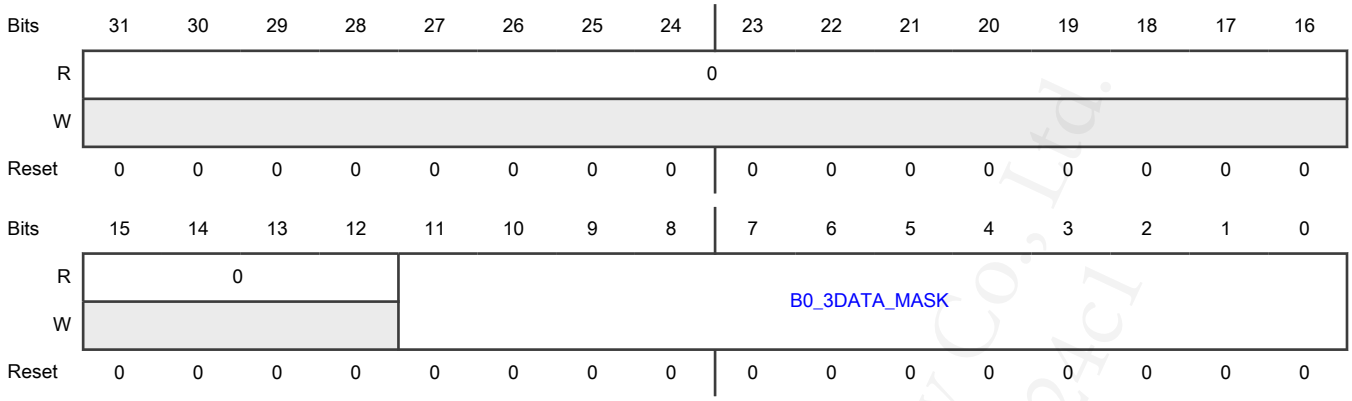
Offset

Register	Offset
EICHD11_WORD1	3C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.17 Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0)

Offset

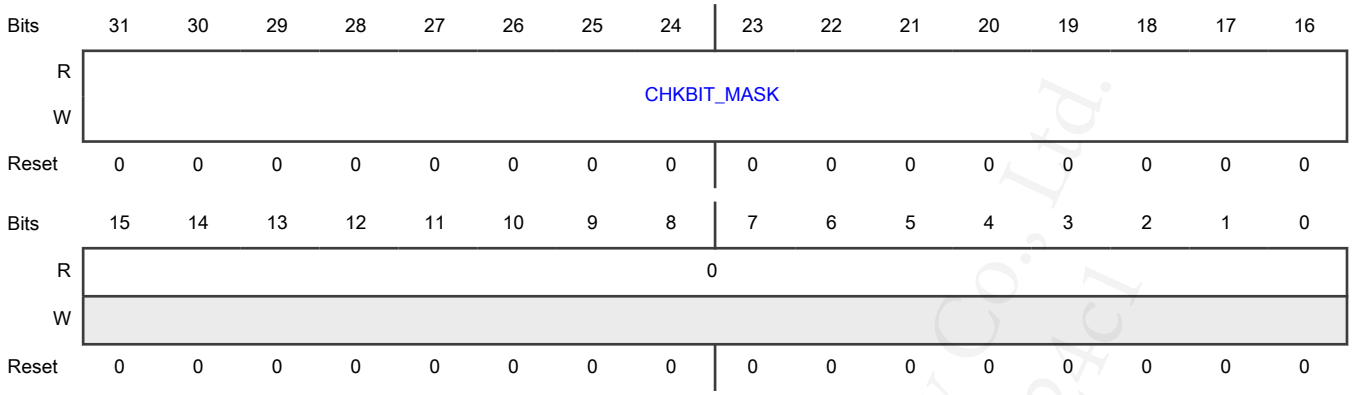
Register	Offset
EICHD12_WORD0	400h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b513-90d8ab624c1

Diagram



Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

73.4.18 Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)

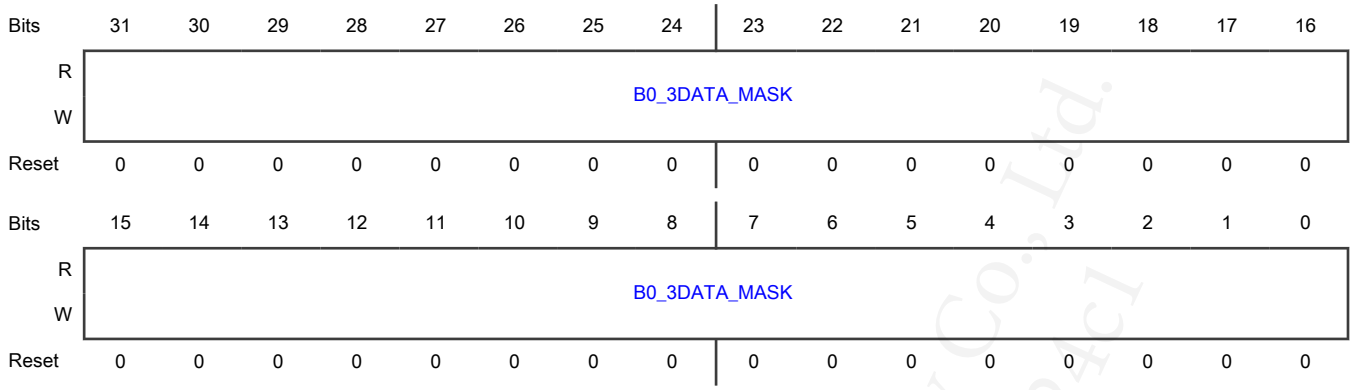
Offset

Register	Offset
EICHD12_WORD1	404h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.19 Error Injection Channel Descriptor n, Word0 (EICHD13_WORD0 - EICHD15_WORD0)

Offset

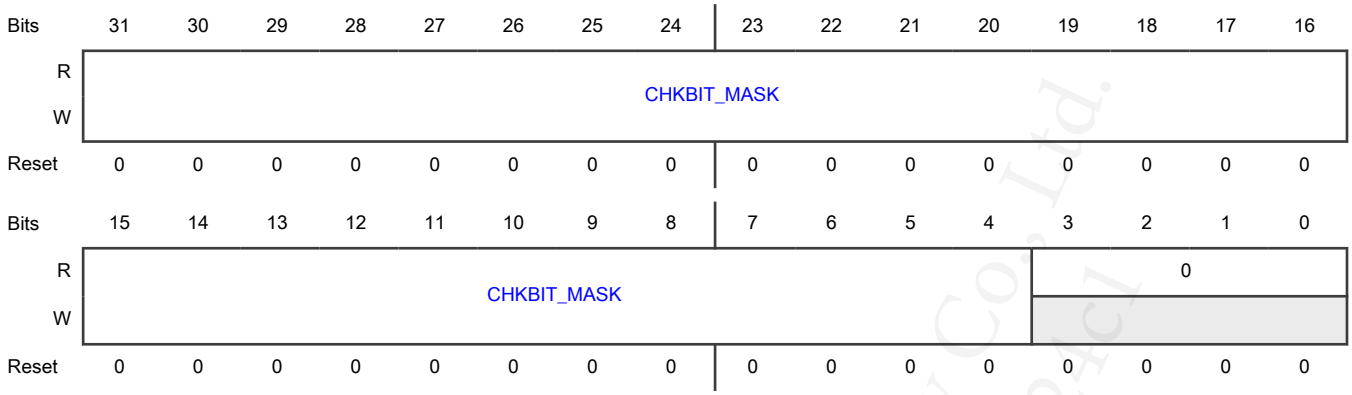
Register	Offset
EICHD13_WORD0	440h
EICHD14_WORD0	480h
EICHD15_WORD0	4C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c4025-4821-48a6-b251-39d3ab624c1

Diagram



Fields

Field	Function
31-4 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[27:0] (28 bits wide), CHKBIT_MASK[27] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
3-0 —	Reserved

73.4.20 Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)

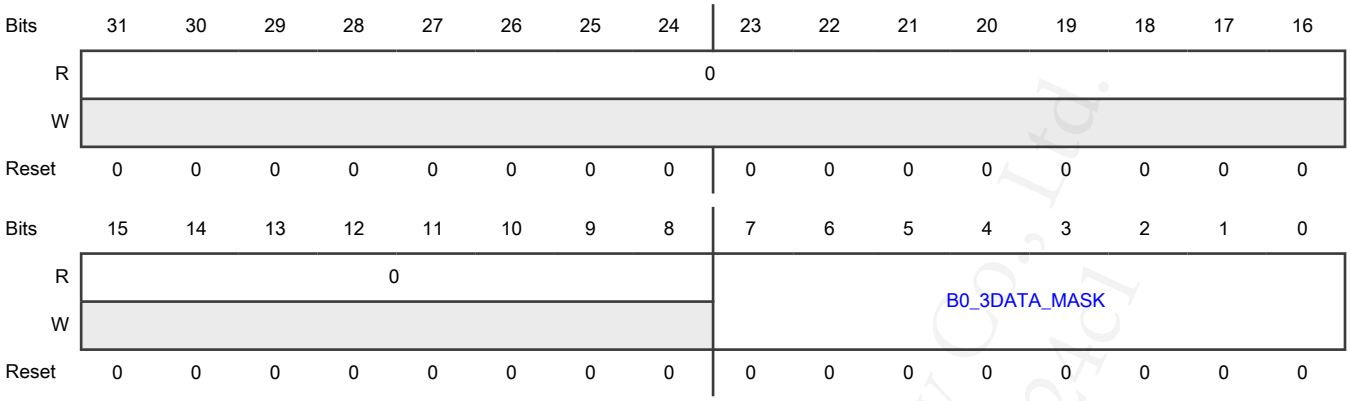
Offset

Register	Offset
EICHD13_WORD1	444h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.21 Error Injection Channel Descriptor n, Word1 (EICHD14_WORD1 - EICHD16_WORD1)

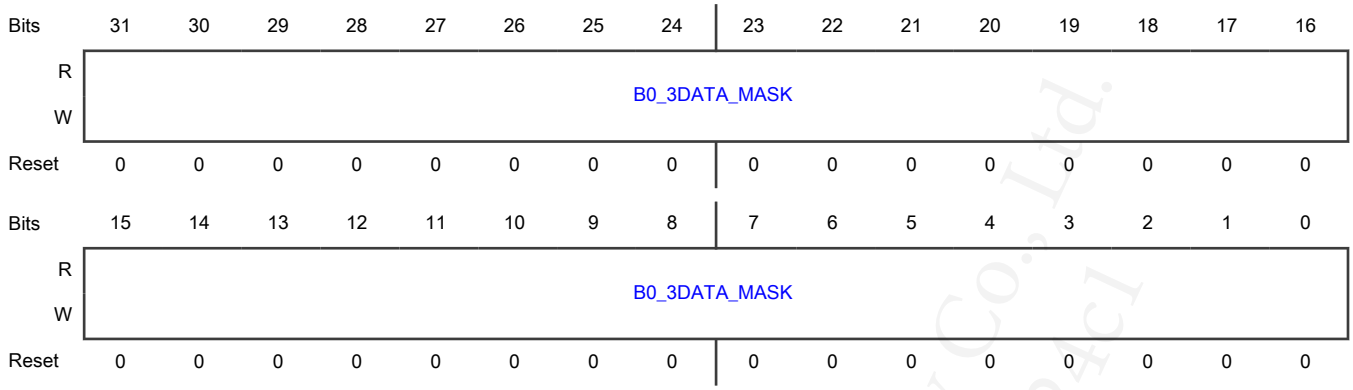
Offset

Register	Offset
EICHD14_WORD1	484h
EICHD15_WORD1	4C4h
EICHD16_WORD1	504h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.22 Error Injection Channel Descriptor 16, Word0 (EICHD16_WORD0)

Offset

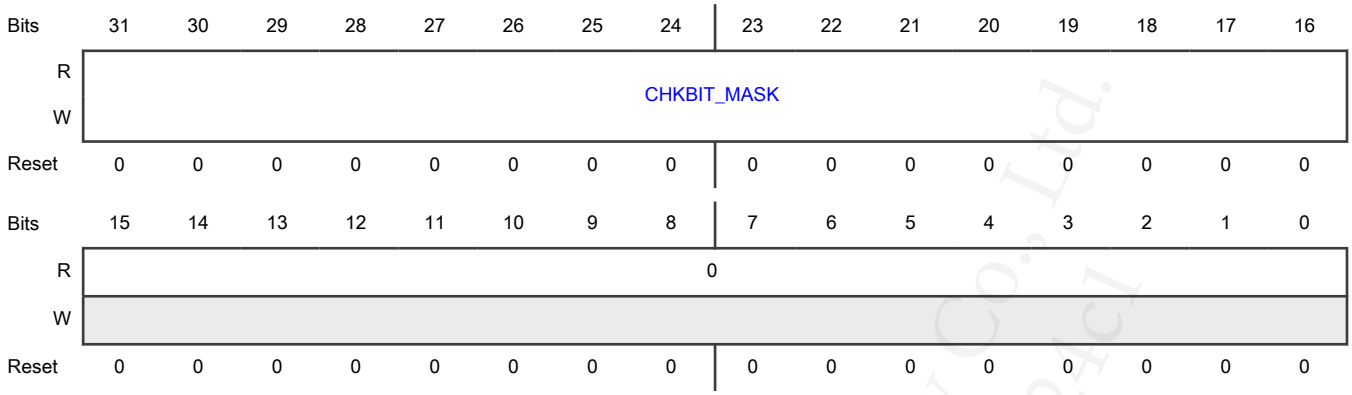
Register	Offset
EICHD16_WORD0	500h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing HHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39d3ab624c1

Diagram



Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

73.4.23 Error Injection Channel Descriptor 17, Word0 (EICHD17_WORD0)

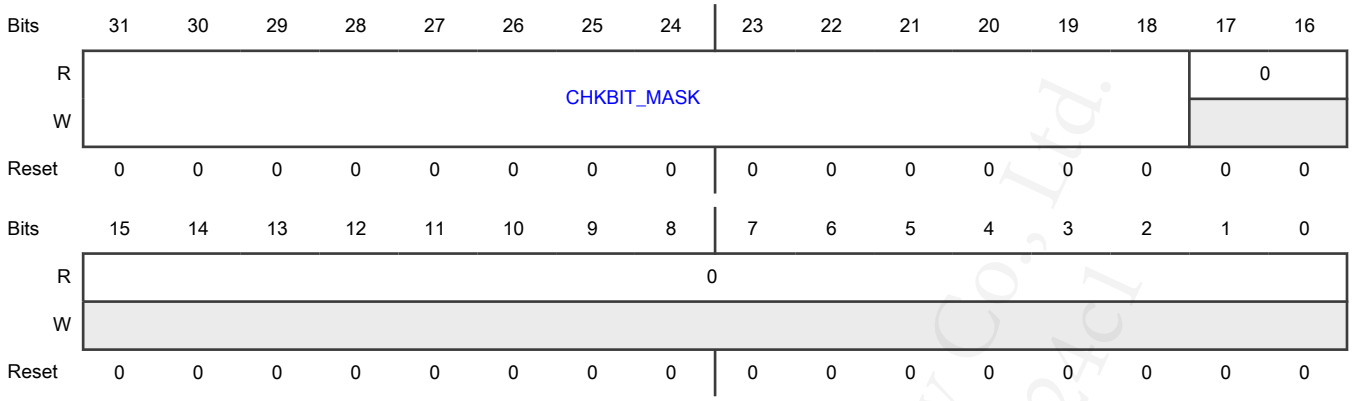
Offset

Register	Offset
EICHD17_WORD0	540h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-18 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[13:0] (14 bits wide), CHKBIT_MASK[13] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
17-0 —	Reserved

73.4.24 Error Injection Channel Descriptor 17, Word1 (EICHD17_WORD1)

Offset

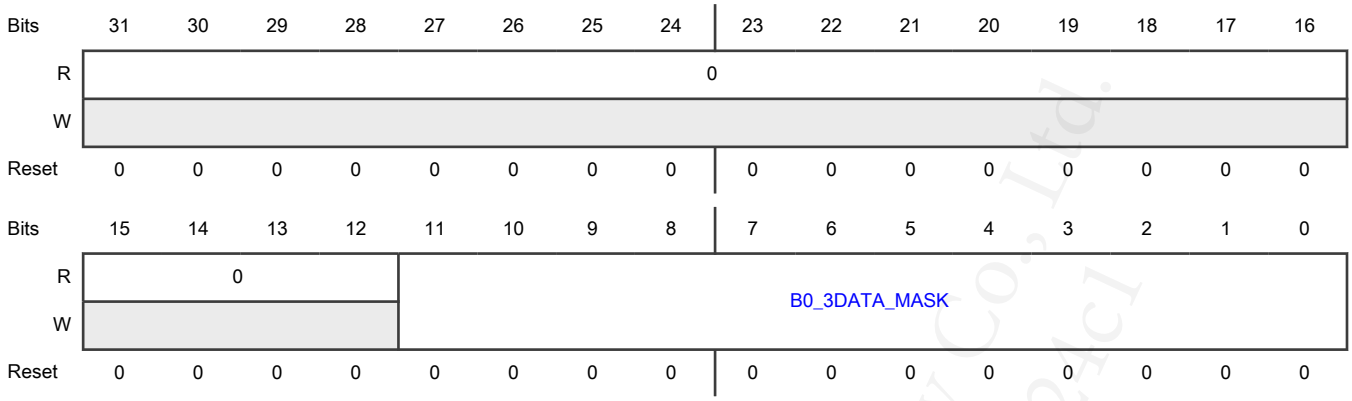
Register	Offset
EICHD17_WORD1	544h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NXP SEMICONDUCTOR CONFIDENTIAL PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. 0462e5482148485b51-59d3ab624c1

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.25 Error Injection Channel Descriptor 18, Word0 (EICHD18_WORD0)

Offset

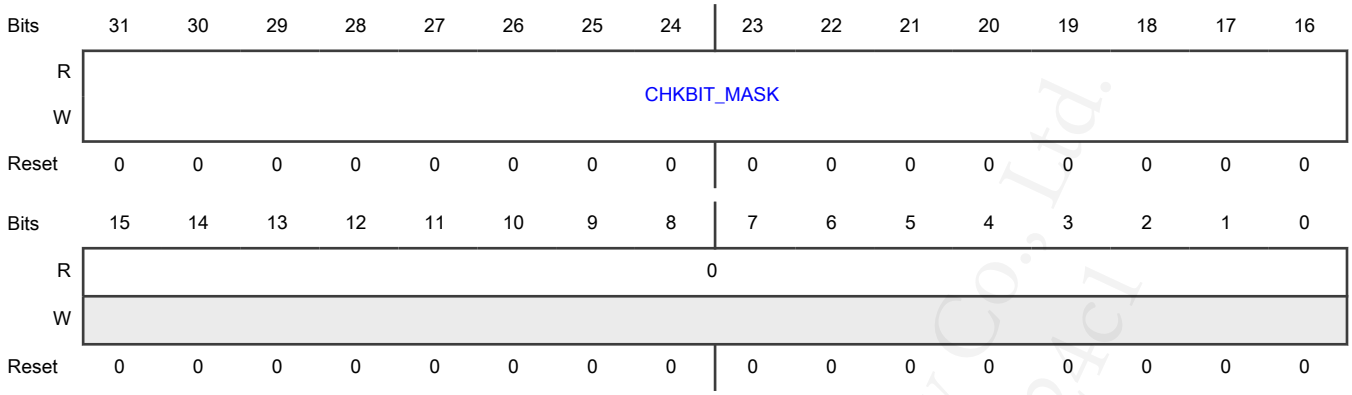
Register	Offset
EICHD18_WORD0	580h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b513-90d8ab624c1

Diagram



Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

73.4.26 Error Injection Channel Descriptor 18, Word1 (EICHD18_WORD1)

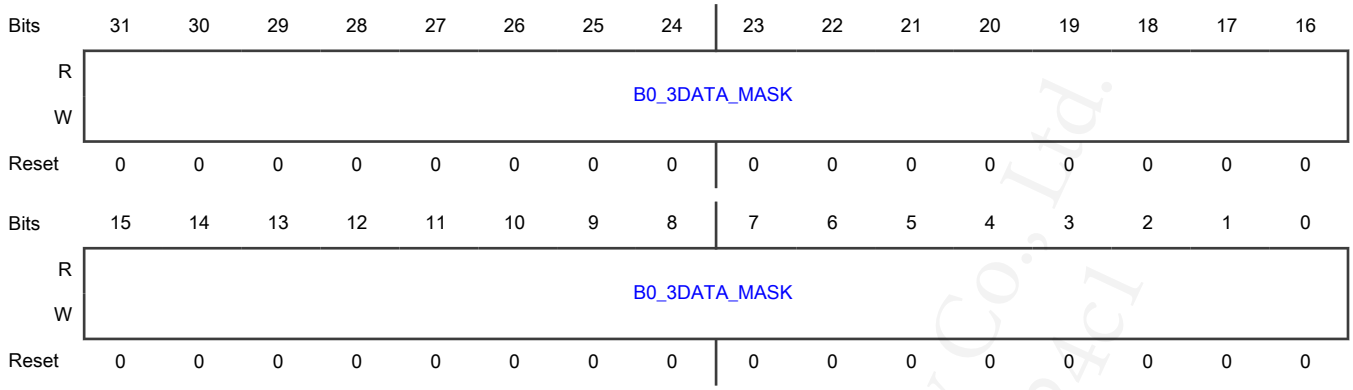
Offset

Register	Offset
EICHD18_WORD1	584h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.27 Error Injection Channel Descriptor n, Word0 (EICHD19_WORD0 - EICHD21_WORD0)

Offset

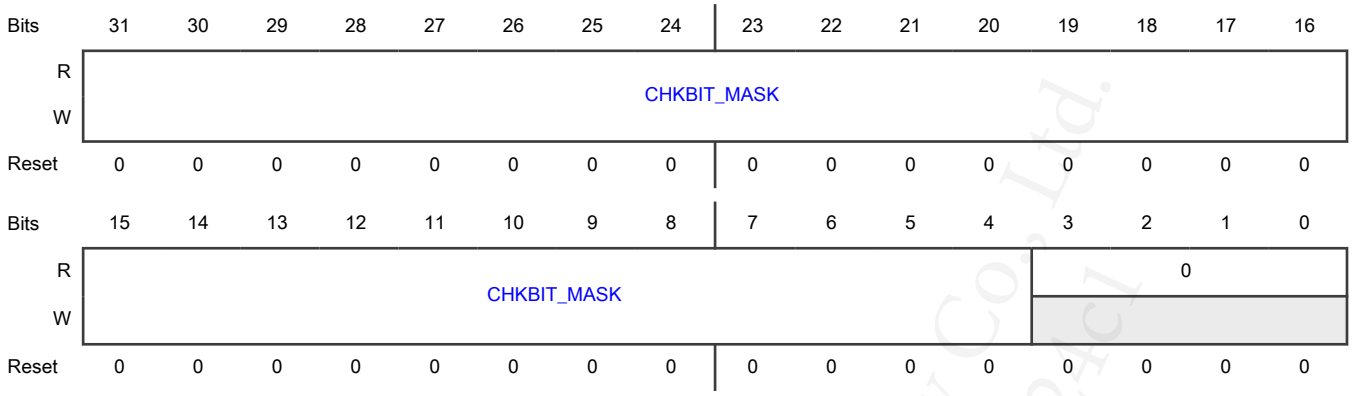
Register	Offset
EICHD19_WORD0	5C0h
EICHD20_WORD0	600h
EICHD21_WORD0	640h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co.,Ltd. c8c4025-4821-48a6-b251-39d3ab624c1

Diagram



Fields

Field	Function
31-4 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[27:0] (28 bits wide), CHKBIT_MASK[27] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
3-0 —	Reserved

73.4.28 Error Injection Channel Descriptor 19, Word1 (EICHD19_WORD1)

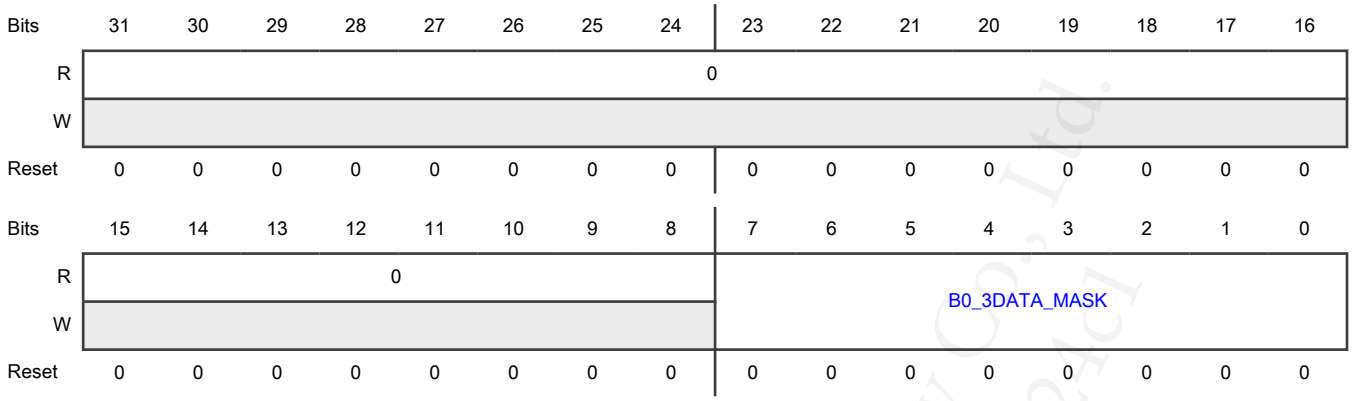
Offset

Register	Offset
EICHD19_WORD1	5C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.29 Error Injection Channel Descriptor n, Word1 (EICHD20_WORD1 - EICHD22_WORD1)

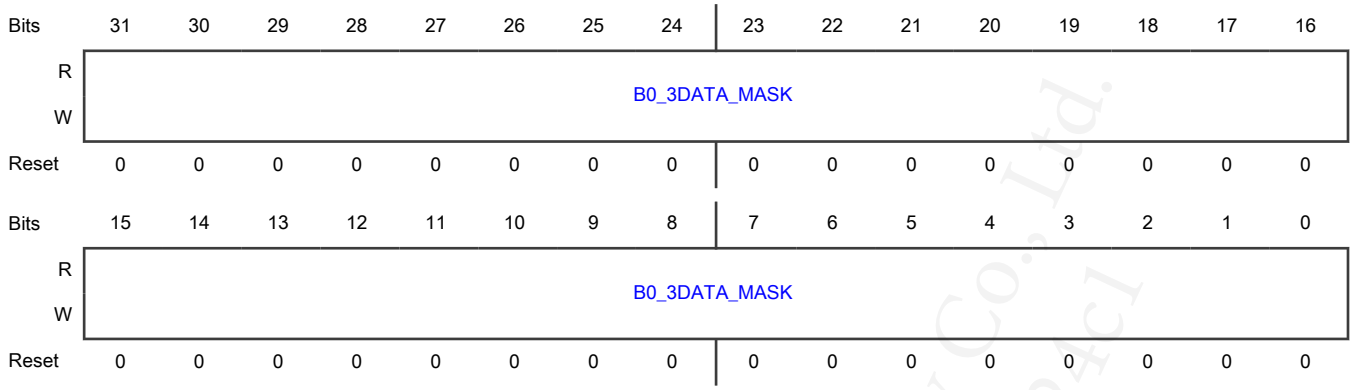
Offset

Register	Offset
EICHD20_WORD1	604h
EICHD21_WORD1	644h
EICHD22_WORD1	684h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.30 Error Injection Channel Descriptor 22, Word0 (EICHD22_WORD0)

Offset

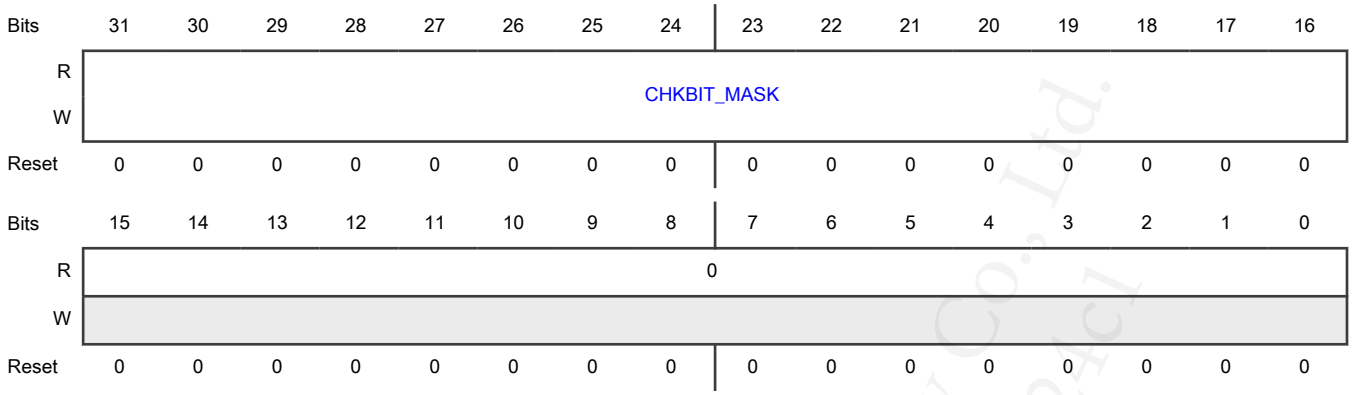
Register	Offset
EICHD22_WORD0	680h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39d3ab624c1

Diagram



Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

73.4.31 Error Injection Channel Descriptor 23, Word0 (EICHD23_WORD0)

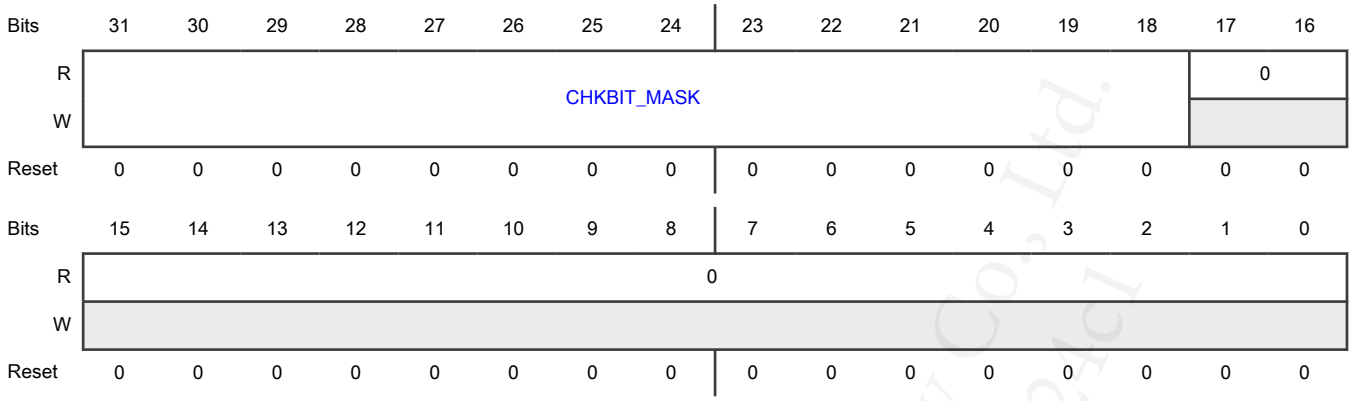
Offset

Register	Offset
EICHD23_WORD0	6C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-18 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[13:0] (14 bits wide), CHKBIT_MASK[13] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
17-0 —	Reserved

73.4.32 Error Injection Channel Descriptor 23, Word1 (EICHD23_WORD1)

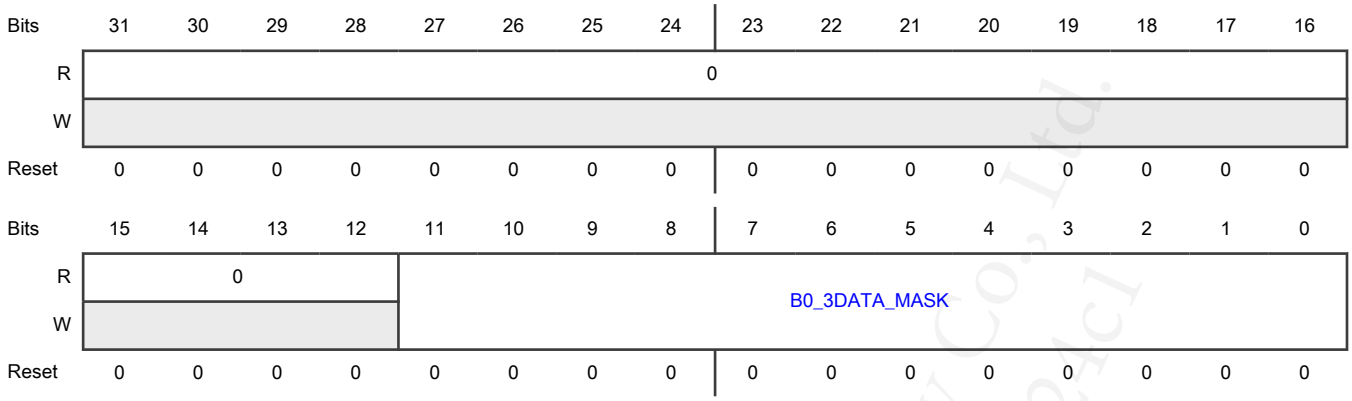
Offset

Register	Offset
EICHD23_WORD1	6C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 —	Reserved
11-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.33 Error Injection Channel Descriptor 24, Word0 (EICHD24_WORD0)

Offset

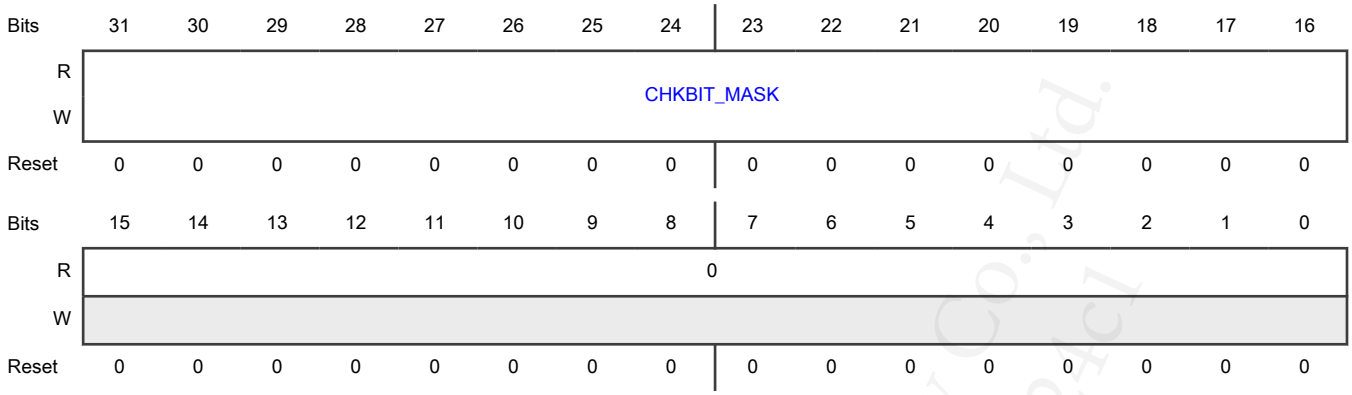
Register	Offset
EICHD24_WORD0	700h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b513-90d8ab624c1

Diagram



Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

73.4.34 Error Injection Channel Descriptor 24, Word1 (EICHD24_WORD1)

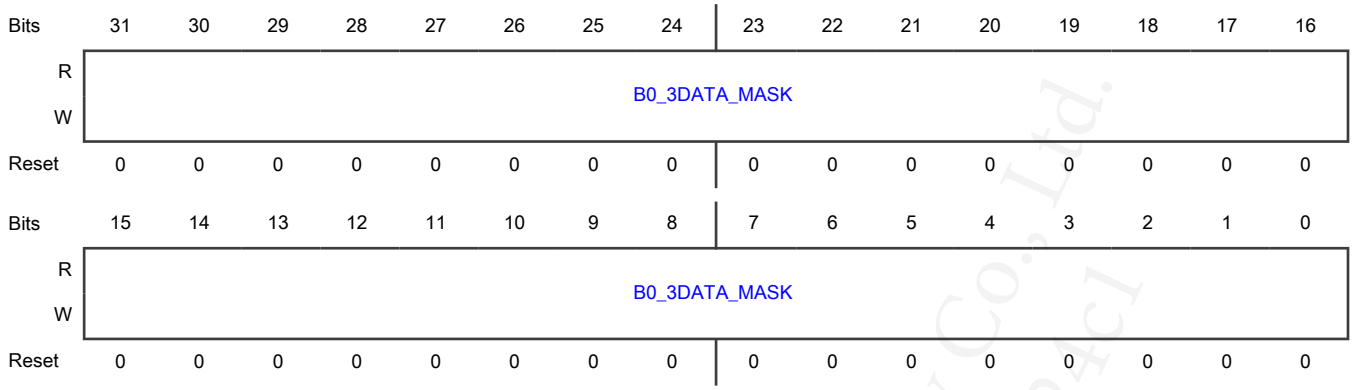
Offset

Register	Offset
EICHD24_WORD1	704h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.35 Error Injection Channel Descriptor n, Word0 (EICHD25_WORD0 - EICHD27_WORD0)

Offset

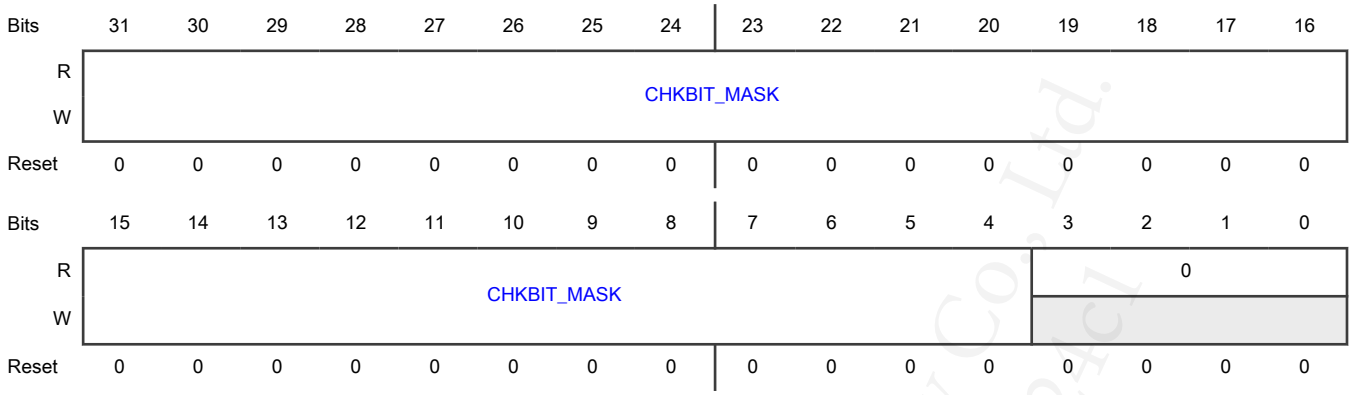
Register	Offset
EICHD25_WORD0	740h
EICHD26_WORD0	780h
EICHD27_WORD0	7C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co.,Ltd. c8c4025-4821-48a6-b251-39d3ab624c1

Diagram



Fields

Field	Function
31-4 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[27:0] (28 bits wide), CHKBIT_MASK[27] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
3-0 —	Reserved

73.4.36 Error Injection Channel Descriptor 25, Word1 (EICHD25_WORD1)

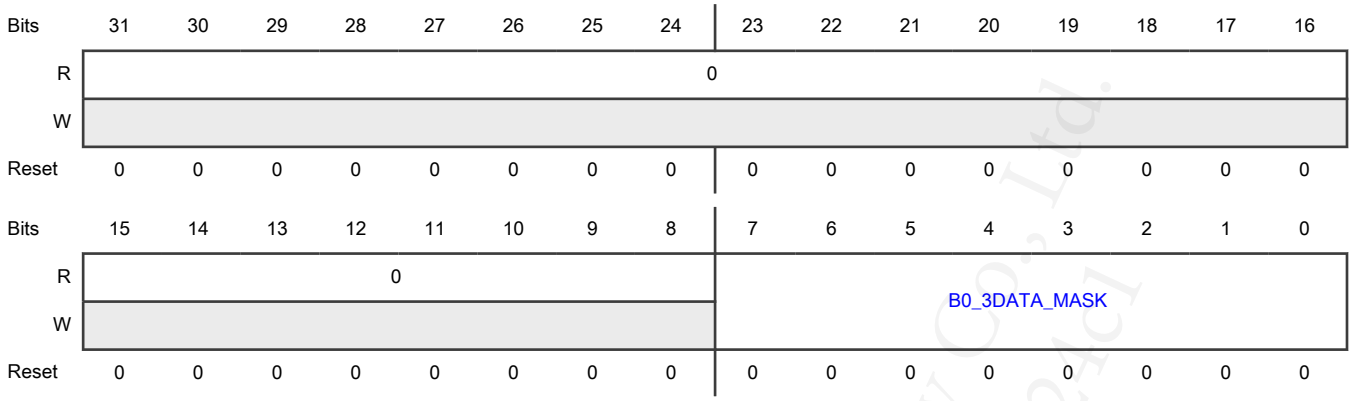
Offset

Register	Offset
EICHD25_WORD1	744h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.37 Error Injection Channel Descriptor n, Word1 (EICHD26_WORD1 - EICHD27_WORD1)

Offset

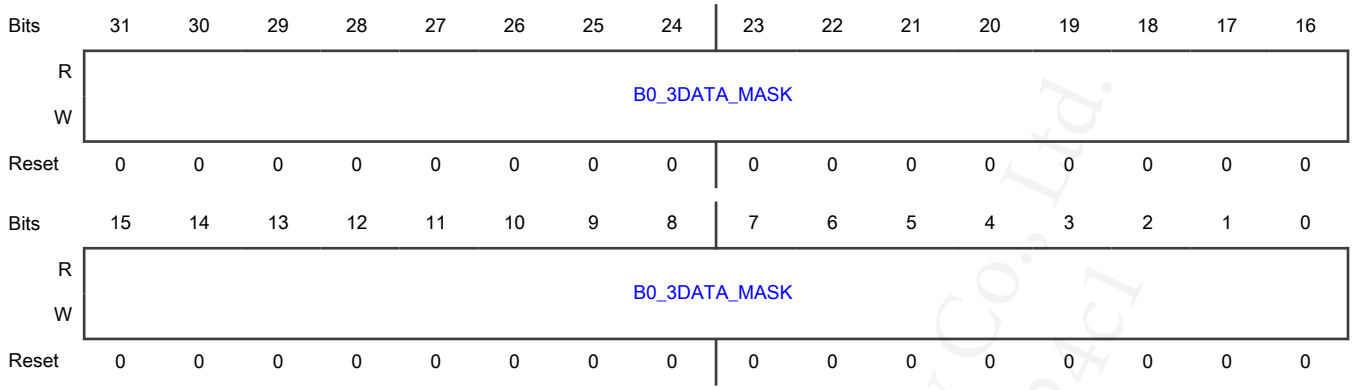
Register	Offset
EICHD26_WORD1	784h
EICHD27_WORD1	7C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8a2e5-4821-45a8-b513-90d3ab621c1

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.38 Error Injection Channel Descriptor 28, Word1 (EICHD28_WORD1)

Offset

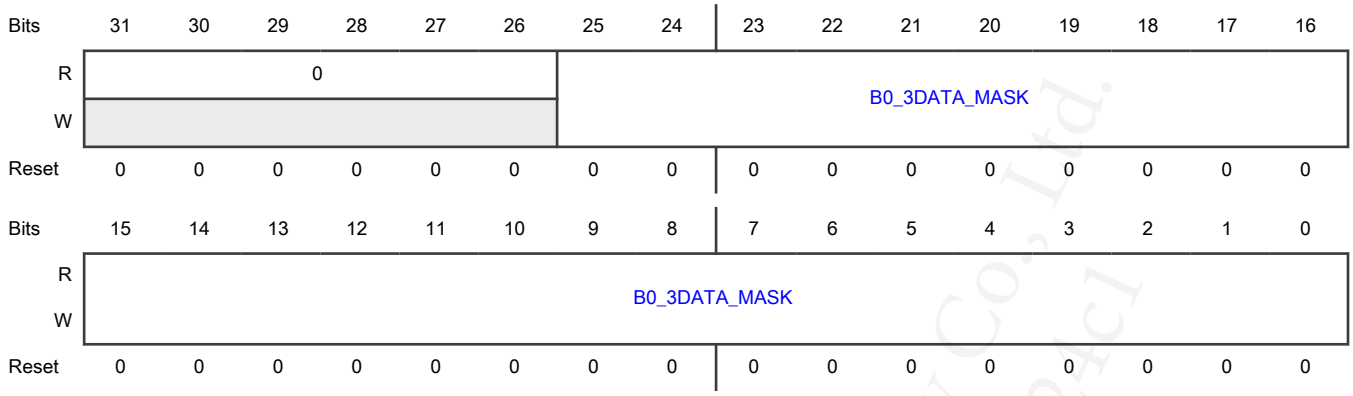
Register	Offset
EICHD28_WORD1	804h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e4-821-48a6-b251-39d3ab624c1

Diagram



Fields

Field	Function
31-26 —	Reserved
25-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.39 Error Injection Channel Descriptor 29, Word1 (EICHD29_WORD1)

Offset

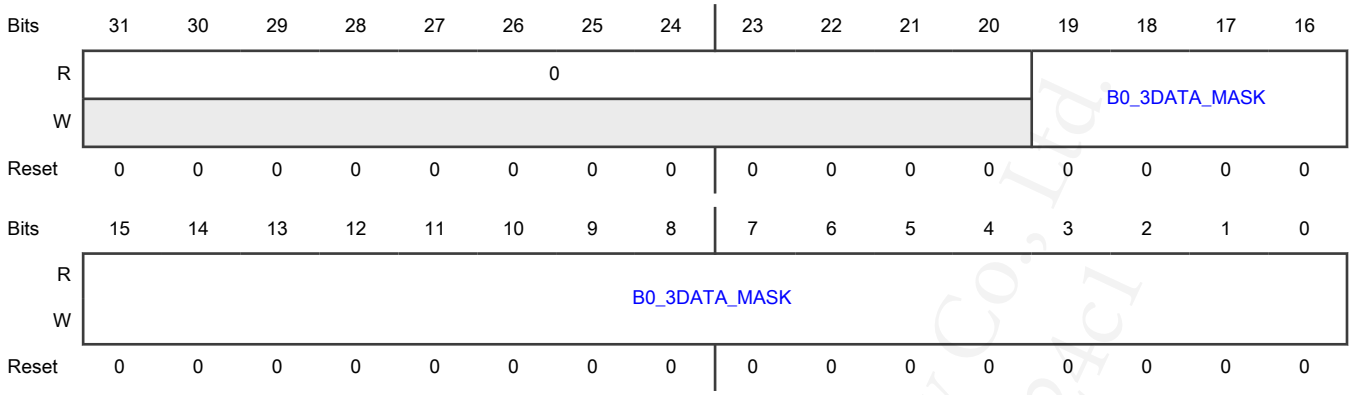
Register	Offset
EICHD29_WORD1	844h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c45e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-20 —	Reserved
19-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.4.40 Error Injection Channel Descriptor 30, Word1 (EICHD30_WORD1)

Offset

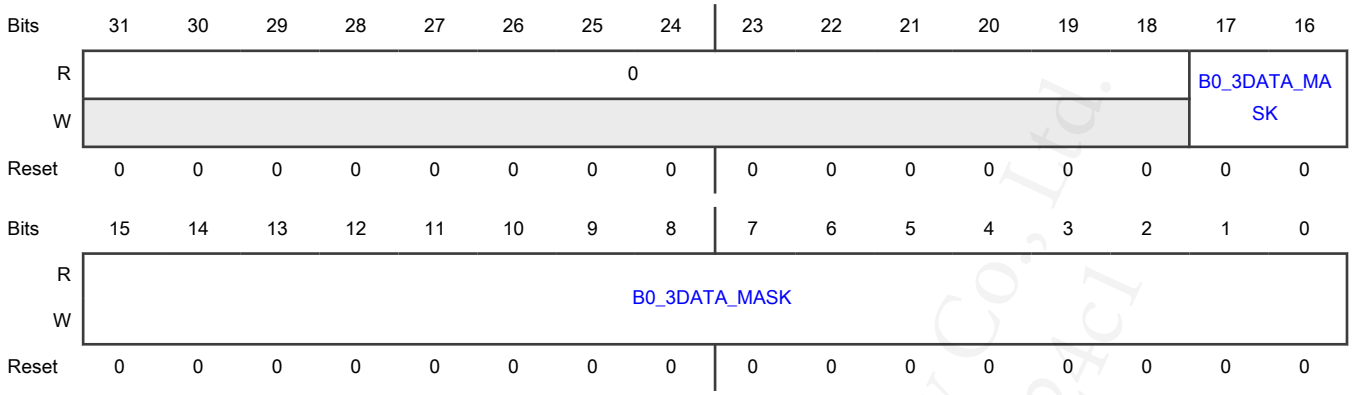
Register	Offset
EICHD30_WORD1	884h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c45e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-18 —	Reserved
17-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.5 EIM_0 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.
  - Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

Error injection channel descriptor: DATA_MASK details

The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 578. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	256	255-224	223-192	191-160

Table 579. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
0	159-128	127-96	95-64	63-32	31-0

73.5.1 EIM_0 memory map

EIM_0 base address: 4033_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
10Ch	Error Injection Channel Descriptor 0, Word3 (EICHD0_WORD3)	32	RW	0000_0000h
110h	Error Injection Channel Descriptor 0, Word4 (EICHD0_WORD4)	32	RW	0000_0000h
114h	Error Injection Channel Descriptor 0, Word5 (EICHD0_WORD5)	32	RW	0000_0000h
118h	Error Injection Channel Descriptor 0, Word6 (EICHD0_WORD6)	32	RW	0000_0000h
11Ch	Error Injection Channel Descriptor 0, Word7 (EICHD0_WORD7)	32	RW	0000_0000h
120h	Error Injection Channel Descriptor 0, Word8 (EICHD0_WORD8)	32	RW	0000_0000h

73.5.2 Error Injection Module Configuration Register (EIMCR)

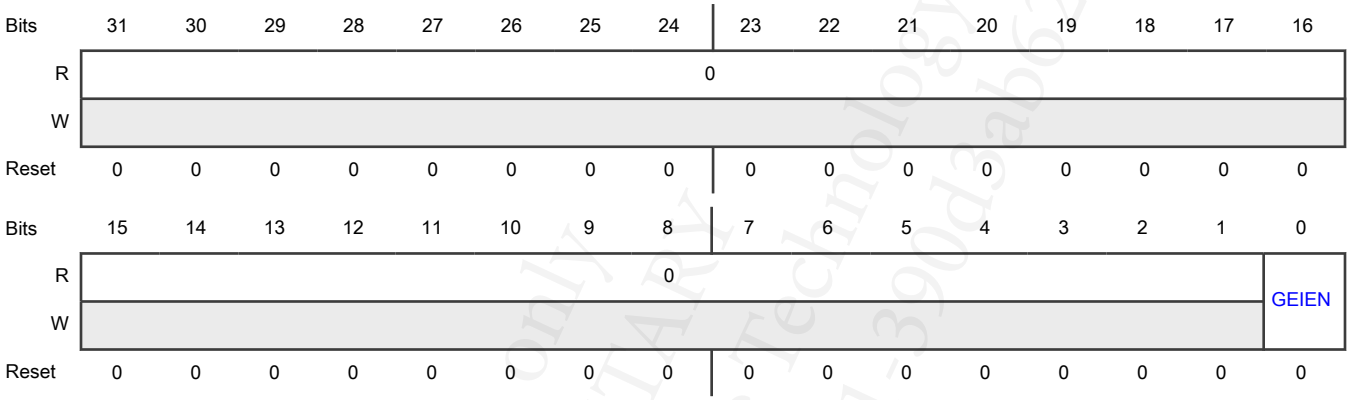
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

73.5.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-b251-390da6b624c1

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 —	Reserved
29 —	Reserved
28 —	Reserved
27	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

73.5.4 Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)

Offset

Register	Offset
EICHD0_WORD1	104h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.5.5 Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)

Offset

Register	Offset
EICHD0_WORD2	108h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Bytes 4-7 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

73.5.6 Error Injection Channel Descriptor 0, Word3 (EICHD0_WORD3)

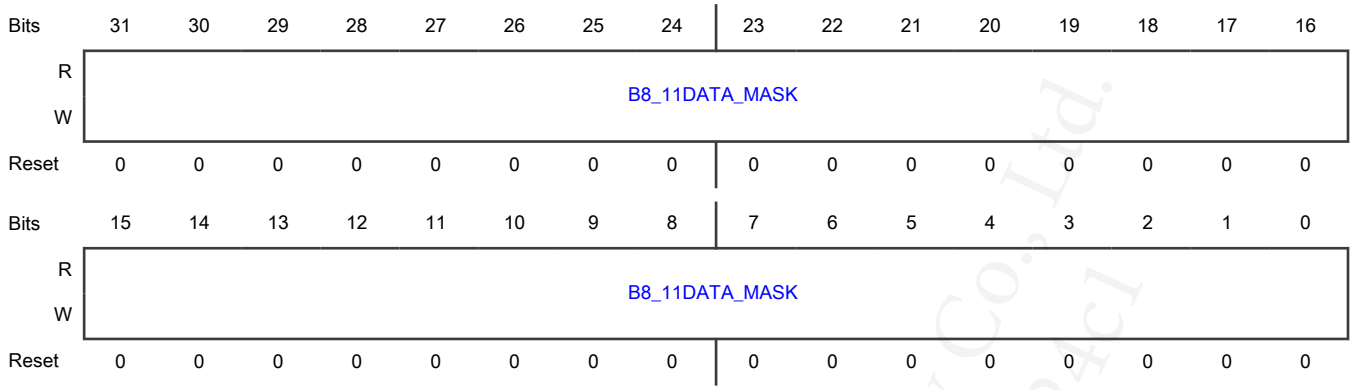
Offset

Register	Offset
EICHD0_WORD3	10Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to bytes 8–11 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Bytes 8-11 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 8-11 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 8-11 on the read data bus is inverted.

73.5.7 Error Injection Channel Descriptor 0, Word4 (EICHD0_WORD4)

Offset

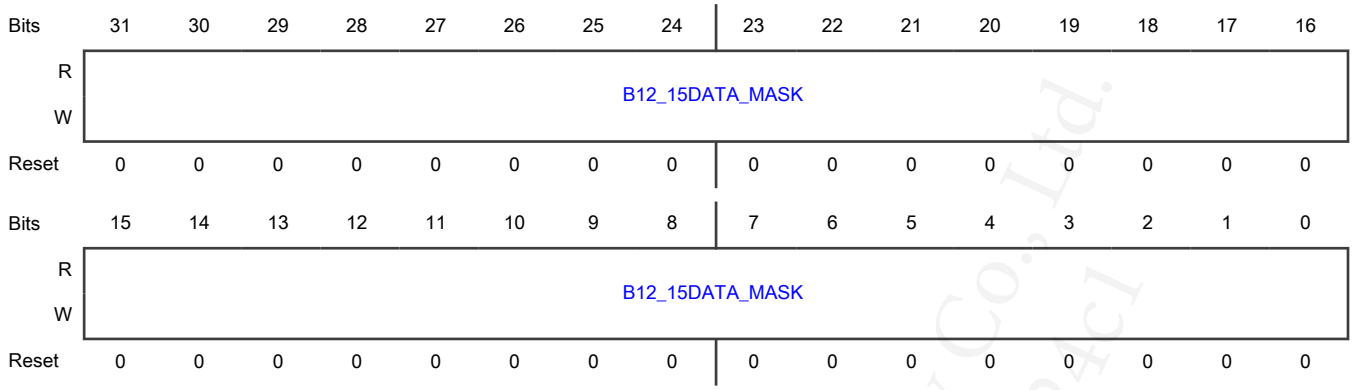
Register	Offset
EICHD0_WORD4	110h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to bytes 12–15 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46220-4821-48a6-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Bytes 12-15 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 12-15 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 12-15 on the read data bus is inverted.

73.5.8 Error Injection Channel Descriptor 0, Word5 (EICHD0_WORD5)

Offset

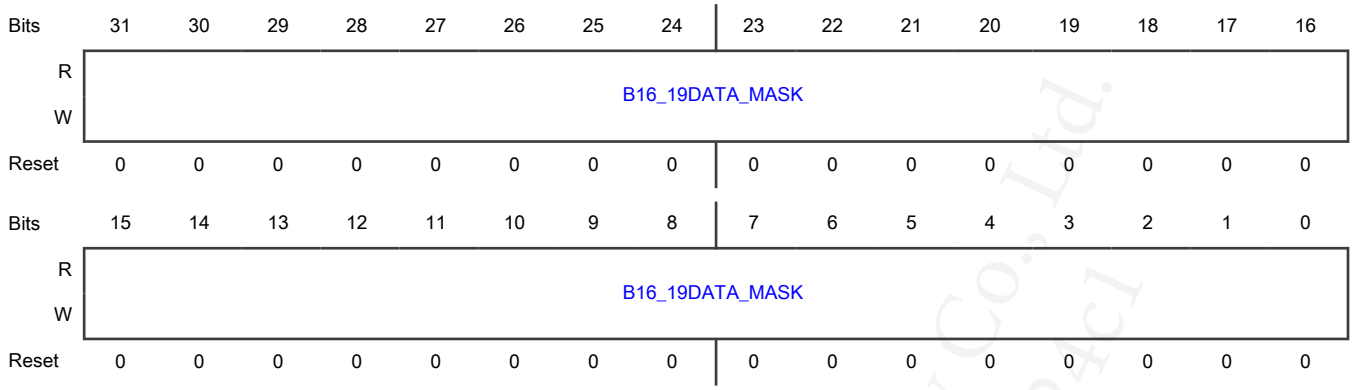
Register	Offset
EICHD0_WORD5	114h

Function

The sixth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B16_19DATA_MASK correspond to bytes 16–19 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA
COMPANY PROPRIETARY
Beijing HHJ Automotive Technology Co., Ltd.
c8c46250-4821-48a6-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B16_19DATA_MASK	Data Mask Bytes 16-19 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B16_19DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 16-19 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 16-19 on the read data bus is inverted.

73.5.9 Error Injection Channel Descriptor 0, Word6 (EICHD0_WORD6)

Offset

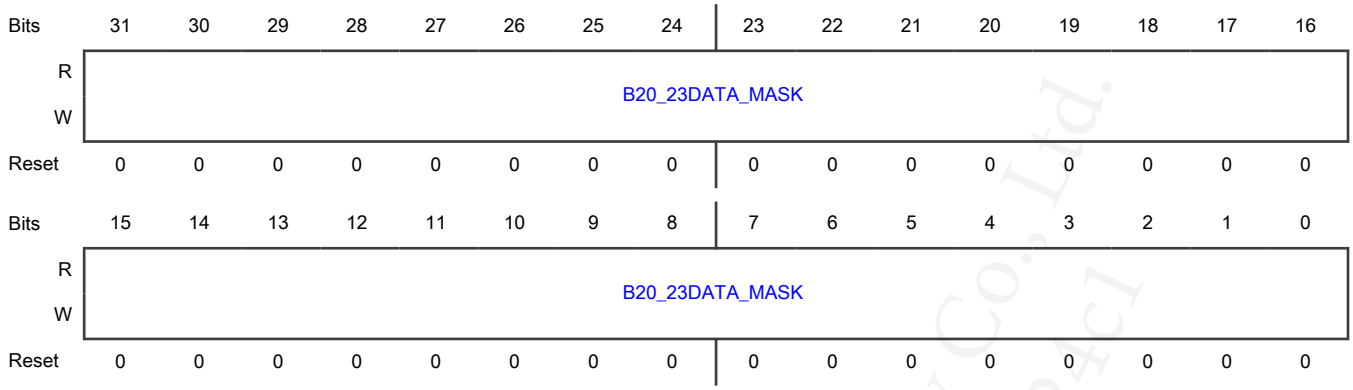
Register	Offset
EICHD0_WORD6	118h

Function

The seventh word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B20_23DATA_MASK correspond to bytes 20–23 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA
COMPANY PROPRIETARY
Beijing HHJ Automotive Technology Co., Ltd.
c8c46205-4821-48a6-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B20_23DATA_MASK	Data Mask Bytes 20-23 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B20_23DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 20-23 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 20-23 on the read data bus is inverted.

73.5.10 Error Injection Channel Descriptor 0, Word7 (EICHD0_WORD7)

Offset

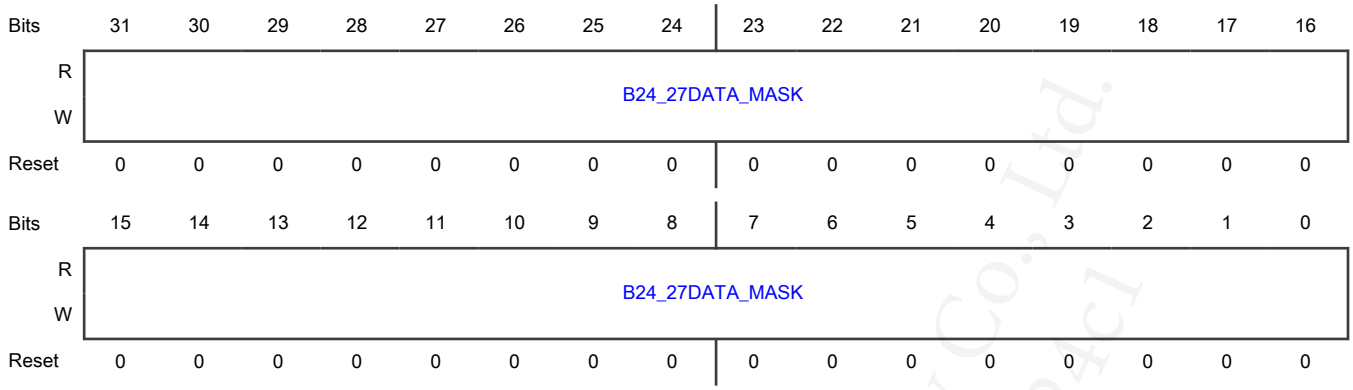
Register	Offset
EICHD0_WORD7	11Ch

Function

The eighth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B24_27DATA_MASK correspond to bytes 24–27 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing HHJ Automotive Technology Co., Ltd.
c8c46205-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B24_27DATA_MASK	Data Mask Bytes 24-27 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B24_27DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 24-27 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 24-27 on the read data bus is inverted.

73.5.11 Error Injection Channel Descriptor 0, Word8 (EICHD0_WORD8)

Offset

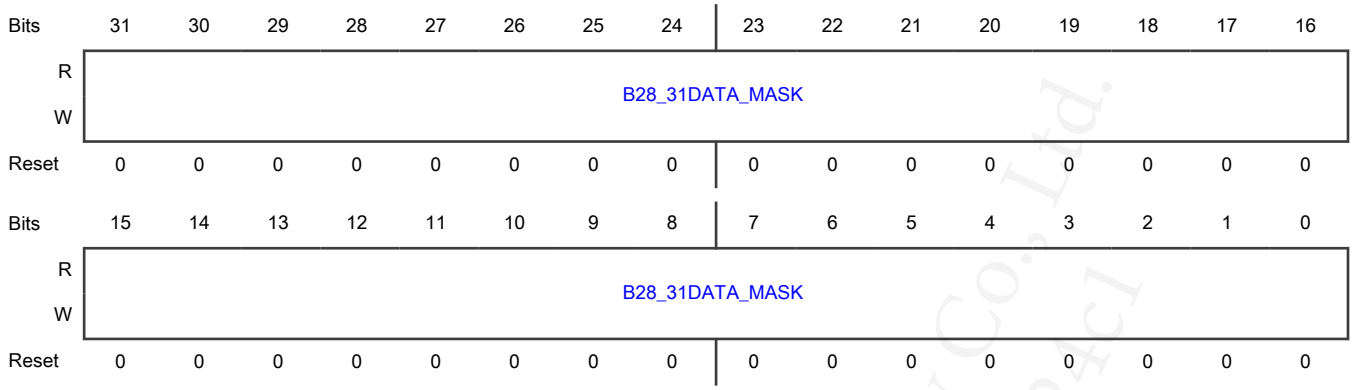
Register	Offset
EICHD0_WORD8	120h

Function

The ninth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B28_31DATA_MASK correspond to bytes 28–31 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing HHJ Automotive Technology Co., Ltd.
c8c46250-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B28_31DATA_MASK	Data Mask Bytes 28-31 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B28_31DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 28-31 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 28-31 on the read data bus is inverted.

73.6 EIM_1 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.

— Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.

— Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 580. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	256	255-224	223-192	191-160
1	256	255-224	223-192	191-160
2	256	255-224	223-192	191-160
3	256	255-224	223-192	191-160
4	256	255-224	223-192	191-160
5	256	255-224	223-192	191-160
6	256	255-224	223-192	191-160
7	256	255-224	223-192	191-160
8	256	255-224	223-192	191-160
9	256	255-224	223-192	191-160
10	256	255-224	223-192	191-160
11	256	255-224	223-192	191-160
12	256	255-224	223-192	191-160
13	256	255-224	223-192	191-160
14	256	255-224	223-192	191-160

Table 581. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
0	159-128	127-96	95-64	63-32	31-0

Table continues on the next page...

Table 581. DATA_MASK bit: Channel-word mapping (continued)

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
1	159-128	127-96	95-64	63-32	31-0
2	159-128	127-96	95-64	63-32	31-0
3	159-128	127-96	95-64	63-32	31-0
4	159-128	127-96	95-64	63-32	31-0
5	159-128	127-96	95-64	63-32	31-0
6	159-128	127-96	95-64	63-32	31-0
7	159-128	127-96	95-64	63-32	31-0
8	159-128	127-96	95-64	63-32	31-0
9	159-128	127-96	95-64	63-32	31-0
10	159-128	127-96	95-64	63-32	31-0
11	159-128	127-96	95-64	63-32	31-0
12	159-128	127-96	95-64	63-32	31-0
13	159-128	127-96	95-64	63-32	31-0
14	159-128	127-96	95-64	63-32	31-0

73.6.1 EIM_1 memory map

EIM_1 base address: 4033_1000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
10Ch	Error Injection Channel Descriptor 0, Word3 (EICHD0_WORD3)	32	RW	0000_0000h
110h	Error Injection Channel Descriptor 0, Word4 (EICHD0_WORD4)	32	RW	0000_0000h
114h	Error Injection Channel Descriptor 0, Word5 (EICHD0_WORD5)	32	RW	0000_0000h
118h	Error Injection Channel Descriptor 0, Word6 (EICHD0_WORD6)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
11Ch	Error Injection Channel Descriptor 0, Word7 (EICHD0_WORD7)	32	RW	0000_0000h
120h	Error Injection Channel Descriptor 0, Word8 (EICHD0_WORD8)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h
14Ch	Error Injection Channel Descriptor 1, Word3 (EICHD1_WORD3)	32	RW	0000_0000h
150h	Error Injection Channel Descriptor 1, Word4 (EICHD1_WORD4)	32	RW	0000_0000h
154h	Error Injection Channel Descriptor 1, Word5 (EICHD1_WORD5)	32	RW	0000_0000h
158h	Error Injection Channel Descriptor 1, Word6 (EICHD1_WORD6)	32	RW	0000_0000h
15Ch	Error Injection Channel Descriptor 1, Word7 (EICHD1_WORD7)	32	RW	0000_0000h
160h	Error Injection Channel Descriptor 1, Word8 (EICHD1_WORD8)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
18Ch	Error Injection Channel Descriptor 2, Word3 (EICHD2_WORD3)	32	RW	0000_0000h
190h	Error Injection Channel Descriptor 2, Word4 (EICHD2_WORD4)	32	RW	0000_0000h
194h	Error Injection Channel Descriptor 2, Word5 (EICHD2_WORD5)	32	RW	0000_0000h
198h	Error Injection Channel Descriptor 2, Word6 (EICHD2_WORD6)	32	RW	0000_0000h
19Ch	Error Injection Channel Descriptor 2, Word7 (EICHD2_WORD7)	32	RW	0000_0000h
1A0h	Error Injection Channel Descriptor 2, Word8 (EICHD2_WORD8)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
1CCh	Error Injection Channel Descriptor 3, Word3 (EICHD3_WORD3)	32	RW	0000_0000h
1D0h	Error Injection Channel Descriptor 3, Word4 (EICHD3_WORD4)	32	RW	0000_0000h
1D4h	Error Injection Channel Descriptor 3, Word5 (EICHD3_WORD5)	32	RW	0000_0000h
1D8h	Error Injection Channel Descriptor 3, Word6 (EICHD3_WORD6)	32	RW	0000_0000h
1DCh	Error Injection Channel Descriptor 3, Word7 (EICHD3_WORD7)	32	RW	0000_0000h
1E0h	Error Injection Channel Descriptor 3, Word8 (EICHD3_WORD8)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
208h	Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)	32	RW	0000_0000h
20Ch	Error Injection Channel Descriptor 4, Word3 (EICHD4_WORD3)	32	RW	0000_0000h
210h	Error Injection Channel Descriptor 4, Word4 (EICHD4_WORD4)	32	RW	0000_0000h
214h	Error Injection Channel Descriptor 4, Word5 (EICHD4_WORD5)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
218h	Error Injection Channel Descriptor 4, Word6 (EICHD4_WORD6)	32	RW	0000_0000h
21Ch	Error Injection Channel Descriptor 4, Word7 (EICHD4_WORD7)	32	RW	0000_0000h
220h	Error Injection Channel Descriptor 4, Word8 (EICHD4_WORD8)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
248h	Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)	32	RW	0000_0000h
24Ch	Error Injection Channel Descriptor 5, Word3 (EICHD5_WORD3)	32	RW	0000_0000h
250h	Error Injection Channel Descriptor 5, Word4 (EICHD5_WORD4)	32	RW	0000_0000h
254h	Error Injection Channel Descriptor 5, Word5 (EICHD5_WORD5)	32	RW	0000_0000h
258h	Error Injection Channel Descriptor 5, Word6 (EICHD5_WORD6)	32	RW	0000_0000h
25Ch	Error Injection Channel Descriptor 5, Word7 (EICHD5_WORD7)	32	RW	0000_0000h
260h	Error Injection Channel Descriptor 5, Word8 (EICHD5_WORD8)	32	RW	0000_0000h
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h
288h	Error Injection Channel Descriptor 6, Word2 (EICHD6_WORD2)	32	RW	0000_0000h
28Ch	Error Injection Channel Descriptor 6, Word3 (EICHD6_WORD3)	32	RW	0000_0000h
290h	Error Injection Channel Descriptor 6, Word4 (EICHD6_WORD4)	32	RW	0000_0000h
294h	Error Injection Channel Descriptor 6, Word5 (EICHD6_WORD5)	32	RW	0000_0000h
298h	Error Injection Channel Descriptor 6, Word6 (EICHD6_WORD6)	32	RW	0000_0000h
29Ch	Error Injection Channel Descriptor 6, Word7 (EICHD6_WORD7)	32	RW	0000_0000h
2A0h	Error Injection Channel Descriptor 6, Word8 (EICHD6_WORD8)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
2C8h	Error Injection Channel Descriptor 7, Word2 (EICHD7_WORD2)	32	RW	0000_0000h
2CCh	Error Injection Channel Descriptor 7, Word3 (EICHD7_WORD3)	32	RW	0000_0000h
2D0h	Error Injection Channel Descriptor 7, Word4 (EICHD7_WORD4)	32	RW	0000_0000h
2D4h	Error Injection Channel Descriptor 7, Word5 (EICHD7_WORD5)	32	RW	0000_0000h
2D8h	Error Injection Channel Descriptor 7, Word6 (EICHD7_WORD6)	32	RW	0000_0000h
2DCh	Error Injection Channel Descriptor 7, Word7 (EICHD7_WORD7)	32	RW	0000_0000h
2E0h	Error Injection Channel Descriptor 7, Word8 (EICHD7_WORD8)	32	RW	0000_0000h
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
308h	Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)	32	RW	0000_0000h
30Ch	Error Injection Channel Descriptor 8, Word3 (EICHD8_WORD3)	32	RW	0000_0000h
310h	Error Injection Channel Descriptor 8, Word4 (EICHD8_WORD4)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
314h	Error Injection Channel Descriptor 8, Word5 (EICHD8_WORD5)	32	RW	0000_0000h
318h	Error Injection Channel Descriptor 8, Word6 (EICHD8_WORD6)	32	RW	0000_0000h
31Ch	Error Injection Channel Descriptor 8, Word7 (EICHD8_WORD7)	32	RW	0000_0000h
320h	Error Injection Channel Descriptor 8, Word8 (EICHD8_WORD8)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
348h	Error Injection Channel Descriptor 9, Word2 (EICHD9_WORD2)	32	RW	0000_0000h
34Ch	Error Injection Channel Descriptor 9, Word3 (EICHD9_WORD3)	32	RW	0000_0000h
350h	Error Injection Channel Descriptor 9, Word4 (EICHD9_WORD4)	32	RW	0000_0000h
354h	Error Injection Channel Descriptor 9, Word5 (EICHD9_WORD5)	32	RW	0000_0000h
358h	Error Injection Channel Descriptor 9, Word6 (EICHD9_WORD6)	32	RW	0000_0000h
35Ch	Error Injection Channel Descriptor 9, Word7 (EICHD9_WORD7)	32	RW	0000_0000h
360h	Error Injection Channel Descriptor 9, Word8 (EICHD9_WORD8)	32	RW	0000_0000h
384h	Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)	32	RW	0000_0000h
388h	Error Injection Channel Descriptor 10, Word2 (EICHD10_WORD2)	32	RW	0000_0000h
38Ch	Error Injection Channel Descriptor 10, Word3 (EICHD10_WORD3)	32	RW	0000_0000h
390h	Error Injection Channel Descriptor 10, Word4 (EICHD10_WORD4)	32	RW	0000_0000h
394h	Error Injection Channel Descriptor 10, Word5 (EICHD10_WORD5)	32	RW	0000_0000h
398h	Error Injection Channel Descriptor 10, Word6 (EICHD10_WORD6)	32	RW	0000_0000h
39Ch	Error Injection Channel Descriptor 10, Word7 (EICHD10_WORD7)	32	RW	0000_0000h
3A0h	Error Injection Channel Descriptor 10, Word8 (EICHD10_WORD8)	32	RW	0000_0000h
3C4h	Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)	32	RW	0000_0000h
3C8h	Error Injection Channel Descriptor 11, Word2 (EICHD11_WORD2)	32	RW	0000_0000h
3CCh	Error Injection Channel Descriptor 11, Word3 (EICHD11_WORD3)	32	RW	0000_0000h
3D0h	Error Injection Channel Descriptor 11, Word4 (EICHD11_WORD4)	32	RW	0000_0000h
3D4h	Error Injection Channel Descriptor 11, Word5 (EICHD11_WORD5)	32	RW	0000_0000h
3D8h	Error Injection Channel Descriptor 11, Word6 (EICHD11_WORD6)	32	RW	0000_0000h
3DCh	Error Injection Channel Descriptor 11, Word7 (EICHD11_WORD7)	32	RW	0000_0000h
3E0h	Error Injection Channel Descriptor 11, Word8 (EICHD11_WORD8)	32	RW	0000_0000h
404h	Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)	32	RW	0000_0000h
408h	Error Injection Channel Descriptor 12, Word2 (EICHD12_WORD2)	32	RW	0000_0000h
40Ch	Error Injection Channel Descriptor 12, Word3 (EICHD12_WORD3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
410h	Error Injection Channel Descriptor 12, Word4 (EICHD12_WORD4)	32	RW	0000_0000h
414h	Error Injection Channel Descriptor 12, Word5 (EICHD12_WORD5)	32	RW	0000_0000h
418h	Error Injection Channel Descriptor 12, Word6 (EICHD12_WORD6)	32	RW	0000_0000h
41Ch	Error Injection Channel Descriptor 12, Word7 (EICHD12_WORD7)	32	RW	0000_0000h
420h	Error Injection Channel Descriptor 12, Word8 (EICHD12_WORD8)	32	RW	0000_0000h
444h	Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)	32	RW	0000_0000h
448h	Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)	32	RW	0000_0000h
44Ch	Error Injection Channel Descriptor 13, Word3 (EICHD13_WORD3)	32	RW	0000_0000h
450h	Error Injection Channel Descriptor 13, Word4 (EICHD13_WORD4)	32	RW	0000_0000h
454h	Error Injection Channel Descriptor 13, Word5 (EICHD13_WORD5)	32	RW	0000_0000h
458h	Error Injection Channel Descriptor 13, Word6 (EICHD13_WORD6)	32	RW	0000_0000h
45Ch	Error Injection Channel Descriptor 13, Word7 (EICHD13_WORD7)	32	RW	0000_0000h
460h	Error Injection Channel Descriptor 13, Word8 (EICHD13_WORD8)	32	RW	0000_0000h
484h	Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)	32	RW	0000_0000h
488h	Error Injection Channel Descriptor 14, Word2 (EICHD14_WORD2)	32	RW	0000_0000h
48Ch	Error Injection Channel Descriptor 14, Word3 (EICHD14_WORD3)	32	RW	0000_0000h
490h	Error Injection Channel Descriptor 14, Word4 (EICHD14_WORD4)	32	RW	0000_0000h
494h	Error Injection Channel Descriptor 14, Word5 (EICHD14_WORD5)	32	RW	0000_0000h
498h	Error Injection Channel Descriptor 14, Word6 (EICHD14_WORD6)	32	RW	0000_0000h
49Ch	Error Injection Channel Descriptor 14, Word7 (EICHD14_WORD7)	32	RW	0000_0000h
4A0h	Error Injection Channel Descriptor 14, Word8 (EICHD14_WORD8)	32	RW	0000_0000h

73.6.2 Error Injection Module Configuration Register (EIMCR)

Offset

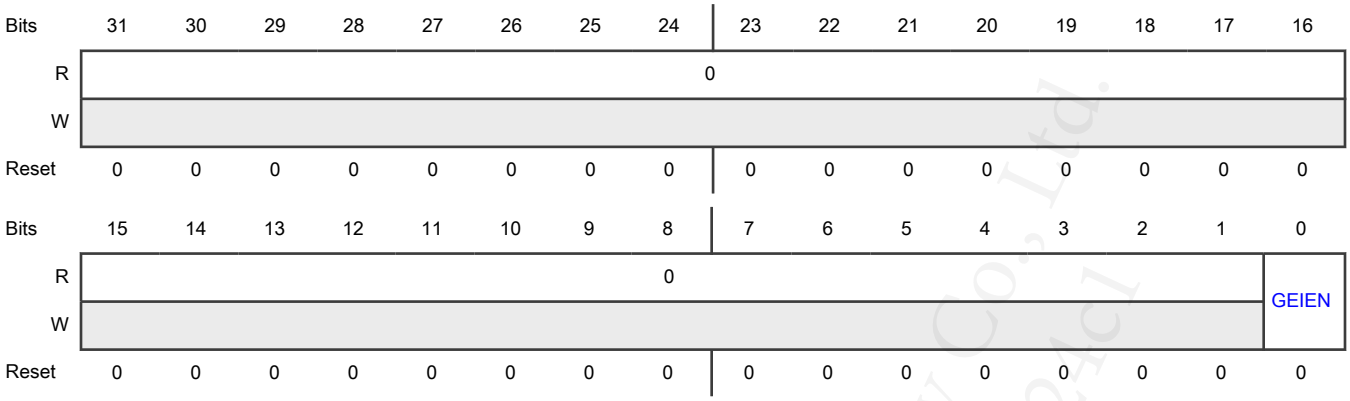
Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Provided by IHS Markit
COMPANY CONFIDENTIAL
Beijing CHINA
c84625-481 2020-5-9 09:06:38

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

73.6.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-821-48a8-b251-390d8ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	EICH1EN	EICH2EN	EICH3EN	EICH4EN	EICH5EN	EICH6EN	EICH7EN	EICH8EN	EICH9EN	EICH10EN	EICH11EN	EICH12EN	EICH13EN	EICH14EN	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 EICH10EN	Error Injection Channel 10 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 10 1b - Error injection is enabled on Error Injection Channel 10
20 EICH11EN	Error Injection Channel 11 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 11 1b - Error injection is enabled on Error Injection Channel 11
19 EICH12EN	Error Injection Channel 12 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 12 1b - Error injection is enabled on Error Injection Channel 12
18 EICH13EN	Error Injection Channel 13 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 13 1b - Error injection is enabled on Error Injection Channel 13
17 EICH14EN	Error Injection Channel 14 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 14 1b - Error injection is enabled on Error Injection Channel 14
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

73.6.4 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD14_WORD1)

Offset

Register	Offset
EICHD0_WORD1	104h
EICHD1_WORD1	144h
EICHD2_WORD1	184h
EICHD3_WORD1	1C4h
EICHD4_WORD1	204h
EICHD5_WORD1	244h
EICHD6_WORD1	284h
EICHD7_WORD1	2C4h
EICHD8_WORD1	304h

Table continues on the next page...

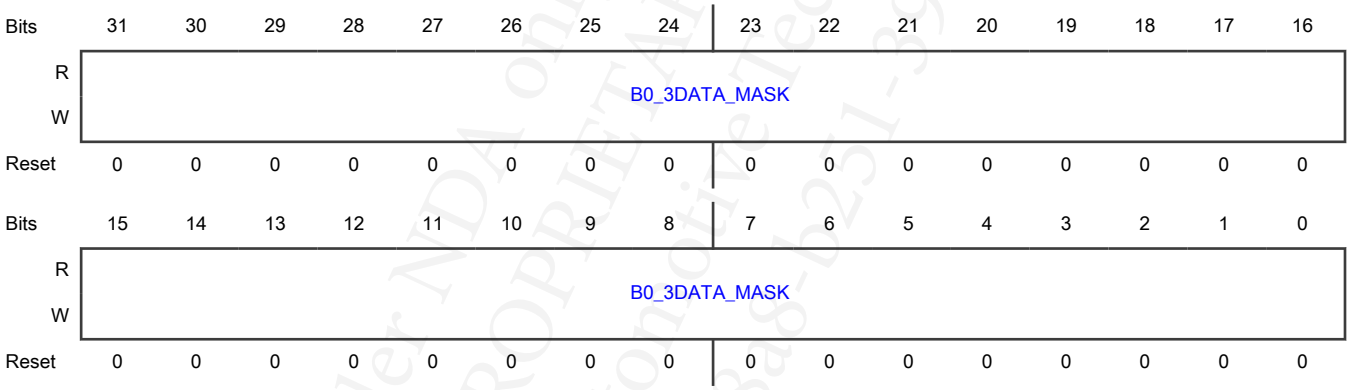
Table continued from the previous page...

Register	Offset
EICHD9_WORD1	344h
EICHD10_WORD1	384h
EICHD11_WORD1	3C4h
EICHD12_WORD1	404h
EICHD13_WORD1	444h
EICHD14_WORD1	484h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing HJ Automotive Technology Co., Ltd.
ca6625f4-8218-4848-b251-590d3e624c1

73.6.5 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD14_WORD2)

Offset

Register	Offset
EICHD0_WORD2	108h
EICHD1_WORD2	148h
EICHD2_WORD2	188h
EICHD3_WORD2	1C8h
EICHD4_WORD2	208h
EICHD5_WORD2	248h
EICHD6_WORD2	288h
EICHD7_WORD2	2C8h
EICHD8_WORD2	308h
EICHD9_WORD2	348h
EICHD10_WORD2	388h
EICHD11_WORD2	3C8h
EICHD12_WORD2	408h
EICHD13_WORD2	448h
EICHD14_WORD2	488h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e54-4921-4e8-b251-390d3ab624c1

Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Bytes 4-7 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

73.6.6 Error Injection Channel Descriptor n, Word3 (EICHD0_WORD3 - EICHD14_WORD3)

Offset

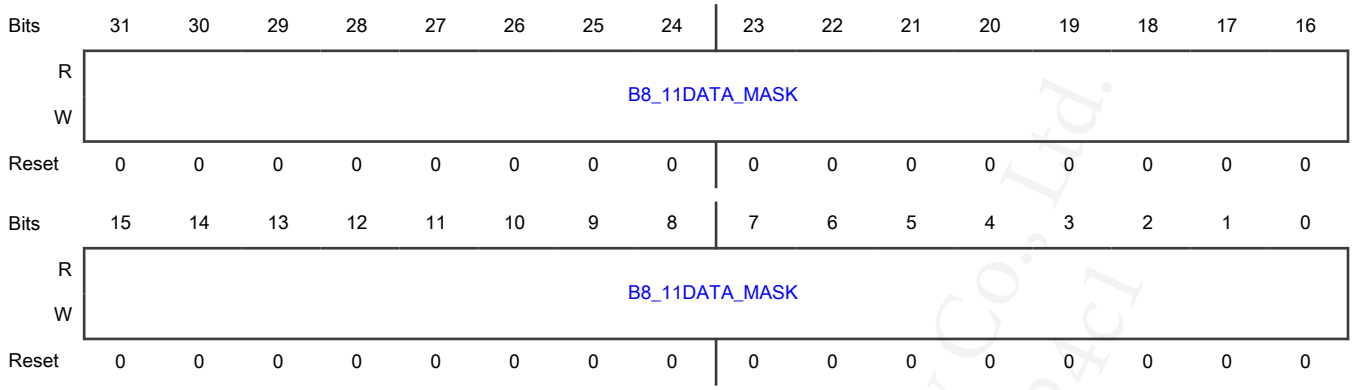
Register	Offset
EICHD0_WORD3	10Ch
EICHD1_WORD3	14Ch
EICHD2_WORD3	18Ch
EICHD3_WORD3	1CCh
EICHD4_WORD3	20Ch
EICHD5_WORD3	24Ch
EICHD6_WORD3	28Ch
EICHD7_WORD3	2CCh
EICHD8_WORD3	30Ch
EICHD9_WORD3	34Ch
EICHD10_WORD3	38Ch
EICHD11_WORD3	3CCh
EICHD12_WORD3	40Ch
EICHD13_WORD3	44Ch
EICHD14_WORD3	48Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to bytes 8–11 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
e462e5-4821-48a8-b251-390d3b62ac1

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Bytes 8-11 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 8-11 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 8-11 on the read data bus is inverted.

73.6.7 Error Injection Channel Descriptor n, Word4 (EICHD0_WORD4 - EICHD14_WORD4)

Offset

Register	Offset
EICHD0_WORD4	110h
EICHD1_WORD4	150h
EICHD2_WORD4	190h
EICHD3_WORD4	1D0h
EICHD4_WORD4	210h
EICHD5_WORD4	250h
EICHD6_WORD4	290h
EICHD7_WORD4	2D0h
EICHD8_WORD4	310h
EICHD9_WORD4	350h

Table continues on the next page...

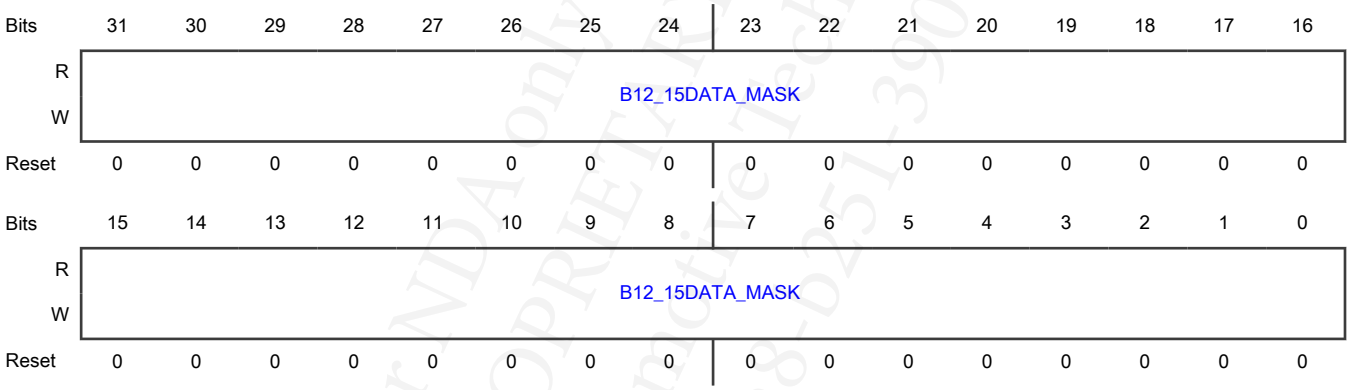
Table continued from the previous page...

Register	Offset
EICHD10_WORD4	390h
EICHD11_WORD4	3D0h
EICHD12_WORD4	410h
EICHD13_WORD4	450h
EICHD14_WORD4	490h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to bytes 12–15 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Bytes 12-15 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 12-15 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 12-15 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJ Automotive Technology Co., Ltd.
c8c8e6e6-421-48a8-9251-390d3a1c24c1

73.6.8 Error Injection Channel Descriptor n, Word5 (EICHD0_WORD5 - EICHD14_WORD5)

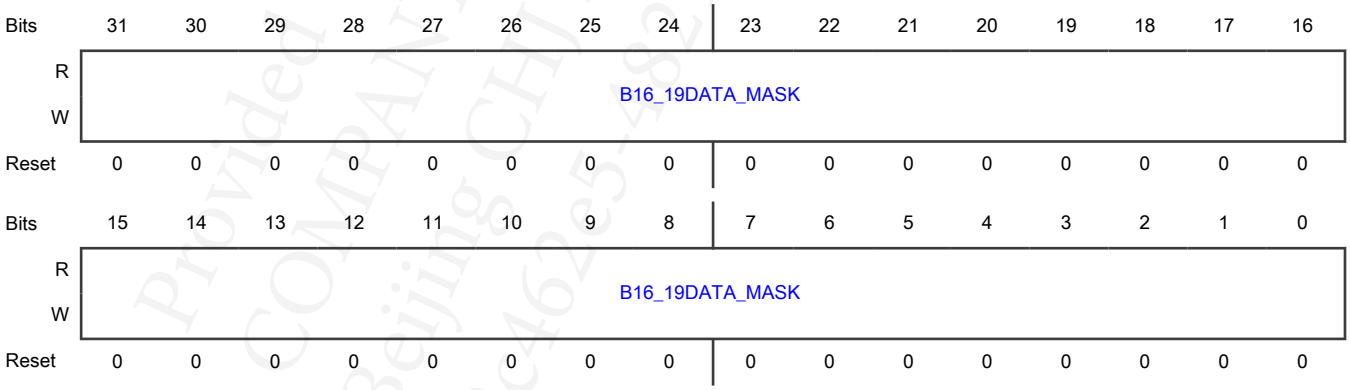
Offset

Register	Offset
EICHD0_WORD5	114h
EICHD1_WORD5	154h
EICHD2_WORD5	194h
EICHD3_WORD5	1D4h
EICHD4_WORD5	214h
EICHD5_WORD5	254h
EICHD6_WORD5	294h
EICHD7_WORD5	2D4h
EICHD8_WORD5	314h
EICHD9_WORD5	354h
EICHD10_WORD5	394h
EICHD11_WORD5	3D4h
EICHD12_WORD5	414h
EICHD13_WORD5	454h
EICHD14_WORD5	494h

Function

The sixth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B16_19DATA_MASK correspond to bytes 16–19 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e54-8921-4e8-b251-390d3ab624c1

Fields

Field	Function
31-0 B16_19DATA_MASK	Data Mask Bytes 16-19 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B16_19DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 16-19 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 16-19 on the read data bus is inverted.

73.6.9 Error Injection Channel Descriptor n, Word6 (EICHD0_WORD6 - EICHD14_WORD6)

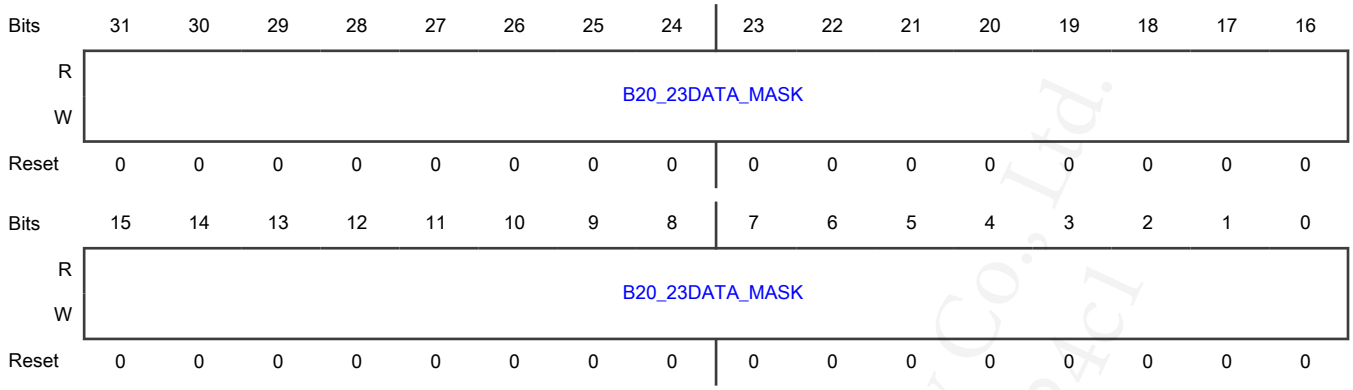
Offset

Register	Offset
EICHD0_WORD6	118h
EICHD1_WORD6	158h
EICHD2_WORD6	198h
EICHD3_WORD6	1D8h
EICHD4_WORD6	218h
EICHD5_WORD6	258h
EICHD6_WORD6	298h
EICHD7_WORD6	2D8h
EICHD8_WORD6	318h
EICHD9_WORD6	358h
EICHD10_WORD6	398h
EICHD11_WORD6	3D8h
EICHD12_WORD6	418h
EICHD13_WORD6	458h
EICHD14_WORD6	498h

Function

The seventh word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B20_23DATA_MASK correspond to bytes 20–23 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B20_23DATA_MASK	Data Mask Bytes 20-23 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B20_23DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 20-23 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 20-23 on the read data bus is inverted.

73.6.10 Error Injection Channel Descriptor n, Word7 (EICHD0_WORD7 - EICHD14_WORD7)

Offset

Register	Offset
EICHD0_WORD7	11Ch
EICHD1_WORD7	15Ch
EICHD2_WORD7	19Ch
EICHD3_WORD7	1DCh
EICHD4_WORD7	21Ch
EICHD5_WORD7	25Ch
EICHD6_WORD7	29Ch
EICHD7_WORD7	2DCh
EICHD8_WORD7	31Ch
EICHD9_WORD7	35Ch

Table continues on the next page...

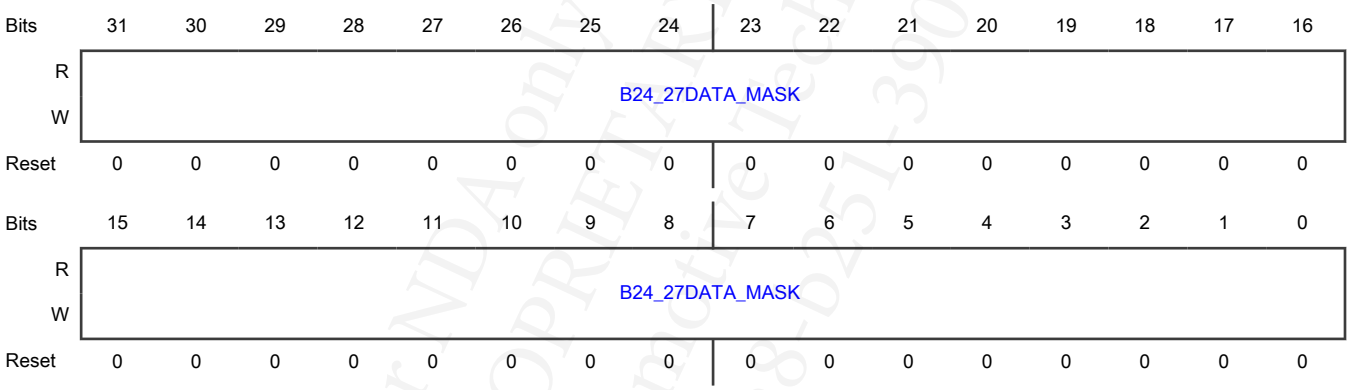
Table continued from the previous page...

Register	Offset
EICHD10_WORD7	39Ch
EICHD11_WORD7	3DCh
EICHD12_WORD7	41Ch
EICHD13_WORD7	45Ch
EICHD14_WORD7	49Ch

Function

The eighth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B24_27DATA_MASK correspond to bytes 24–27 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B24_27DATA_MASK	Data Mask Bytes 24-27 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B24_27DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 24-27 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 24-27 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJ Automotive Technology Co., Ltd.
c8c8e6e6-421-48a8-9251-390d3a824c1

73.6.11 Error Injection Channel Descriptor n, Word8 (EICHD0_WORD8 - EICHD14_WORD8)

Offset

Register	Offset
EICHD0_WORD8	120h
EICHD1_WORD8	160h
EICHD2_WORD8	1A0h
EICHD3_WORD8	1E0h
EICHD4_WORD8	220h
EICHD5_WORD8	260h
EICHD6_WORD8	2A0h
EICHD7_WORD8	2E0h
EICHD8_WORD8	320h
EICHD9_WORD8	360h
EICHD10_WORD8	3A0h
EICHD11_WORD8	3E0h
EICHD12_WORD8	420h
EICHD13_WORD8	460h
EICHD14_WORD8	4A0h

Function

The ninth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B28_31DATA_MASK correspond to bytes 28–31 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B28_31DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B28_31DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e54-4921-4e3-b251-390d3ab624c1

Fields

Field	Function
31-0 B28_31DATA_MASK	Data Mask Bytes 28-31 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B28_31DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 28-31 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 28-31 on the read data bus is inverted.

73.7 EIM_2 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.
  - Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 582. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	256	255-224	223-192	191-160
1	256	255-224	223-192	191-160
2	256	255-224	223-192	191-160
3	256	255-224	223-192	191-160
4	256	255-224	223-192	191-160
5	256	255-224	223-192	191-160

Table 583. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
0	159-128	127-96	95-64	63-32	31-0
1	159-128	127-96	95-64	63-32	31-0
2	159-128	127-96	95-64	63-32	31-0
3	159-128	127-96	95-64	63-32	31-0
4	159-128	127-96	95-64	63-32	31-0
5	159-128	127-96	95-64	63-32	31-0

73.7.1 EIM_2 memory map

EIM_2 base address: 4033_2000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
10Ch	Error Injection Channel Descriptor 0, Word3 (EICHD0_WORD3)	32	RW	0000_0000h
110h	Error Injection Channel Descriptor 0, Word4 (EICHD0_WORD4)	32	RW	0000_0000h
114h	Error Injection Channel Descriptor 0, Word5 (EICHD0_WORD5)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
118h	Error Injection Channel Descriptor 0, Word6 (EICHD0_WORD6)	32	RW	0000_0000h
11Ch	Error Injection Channel Descriptor 0, Word7 (EICHD0_WORD7)	32	RW	0000_0000h
120h	Error Injection Channel Descriptor 0, Word8 (EICHD0_WORD8)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h
14Ch	Error Injection Channel Descriptor 1, Word3 (EICHD1_WORD3)	32	RW	0000_0000h
150h	Error Injection Channel Descriptor 1, Word4 (EICHD1_WORD4)	32	RW	0000_0000h
154h	Error Injection Channel Descriptor 1, Word5 (EICHD1_WORD5)	32	RW	0000_0000h
158h	Error Injection Channel Descriptor 1, Word6 (EICHD1_WORD6)	32	RW	0000_0000h
15Ch	Error Injection Channel Descriptor 1, Word7 (EICHD1_WORD7)	32	RW	0000_0000h
160h	Error Injection Channel Descriptor 1, Word8 (EICHD1_WORD8)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
18Ch	Error Injection Channel Descriptor 2, Word3 (EICHD2_WORD3)	32	RW	0000_0000h
190h	Error Injection Channel Descriptor 2, Word4 (EICHD2_WORD4)	32	RW	0000_0000h
194h	Error Injection Channel Descriptor 2, Word5 (EICHD2_WORD5)	32	RW	0000_0000h
198h	Error Injection Channel Descriptor 2, Word6 (EICHD2_WORD6)	32	RW	0000_0000h
19Ch	Error Injection Channel Descriptor 2, Word7 (EICHD2_WORD7)	32	RW	0000_0000h
1A0h	Error Injection Channel Descriptor 2, Word8 (EICHD2_WORD8)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
1CCh	Error Injection Channel Descriptor 3, Word3 (EICHD3_WORD3)	32	RW	0000_0000h
1D0h	Error Injection Channel Descriptor 3, Word4 (EICHD3_WORD4)	32	RW	0000_0000h
1D4h	Error Injection Channel Descriptor 3, Word5 (EICHD3_WORD5)	32	RW	0000_0000h
1D8h	Error Injection Channel Descriptor 3, Word6 (EICHD3_WORD6)	32	RW	0000_0000h
1DCh	Error Injection Channel Descriptor 3, Word7 (EICHD3_WORD7)	32	RW	0000_0000h
1E0h	Error Injection Channel Descriptor 3, Word8 (EICHD3_WORD8)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
208h	Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)	32	RW	0000_0000h
20Ch	Error Injection Channel Descriptor 4, Word3 (EICHD4_WORD3)	32	RW	0000_0000h
210h	Error Injection Channel Descriptor 4, Word4 (EICHD4_WORD4)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
214h	Error Injection Channel Descriptor 4, Word5 (EICHD4_WORD5)	32	RW	0000_0000h
218h	Error Injection Channel Descriptor 4, Word6 (EICHD4_WORD6)	32	RW	0000_0000h
21Ch	Error Injection Channel Descriptor 4, Word7 (EICHD4_WORD7)	32	RW	0000_0000h
220h	Error Injection Channel Descriptor 4, Word8 (EICHD4_WORD8)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
248h	Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)	32	RW	0000_0000h
24Ch	Error Injection Channel Descriptor 5, Word3 (EICHD5_WORD3)	32	RW	0000_0000h
250h	Error Injection Channel Descriptor 5, Word4 (EICHD5_WORD4)	32	RW	0000_0000h
254h	Error Injection Channel Descriptor 5, Word5 (EICHD5_WORD5)	32	RW	0000_0000h
258h	Error Injection Channel Descriptor 5, Word6 (EICHD5_WORD6)	32	RW	0000_0000h
25Ch	Error Injection Channel Descriptor 5, Word7 (EICHD5_WORD7)	32	RW	0000_0000h
260h	Error Injection Channel Descriptor 5, Word8 (EICHD5_WORD8)	32	RW	0000_0000h

73.7.2 Error Injection Module Configuration Register (EIMCR)

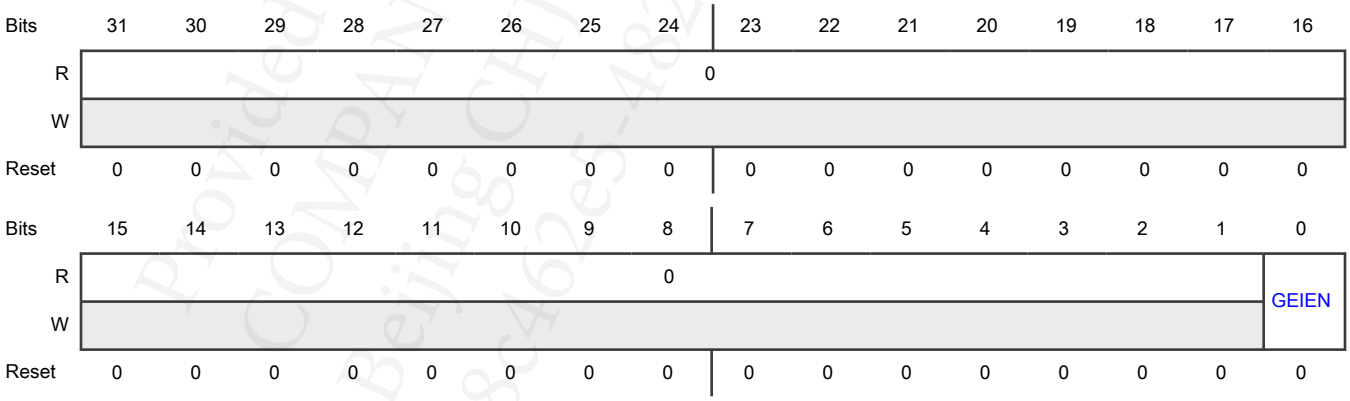
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Provided under NDA
COMPANY PROPRIETARY
Beijing CHI Automotive Technology Co.,Ltd.
c8462e5-4821-48a8-b251-39004bb6a1a1

Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

73.7.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	EICH1EN	EICH2EN	EICH3EN	EICH4EN	EICH5EN	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Beijing CHT Automotive Technology Co., Ltd.
COMPANY PROPRIETARY
Product under NDA only
c8c467e54821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

73.7.4 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD5_WORD1)

Offset

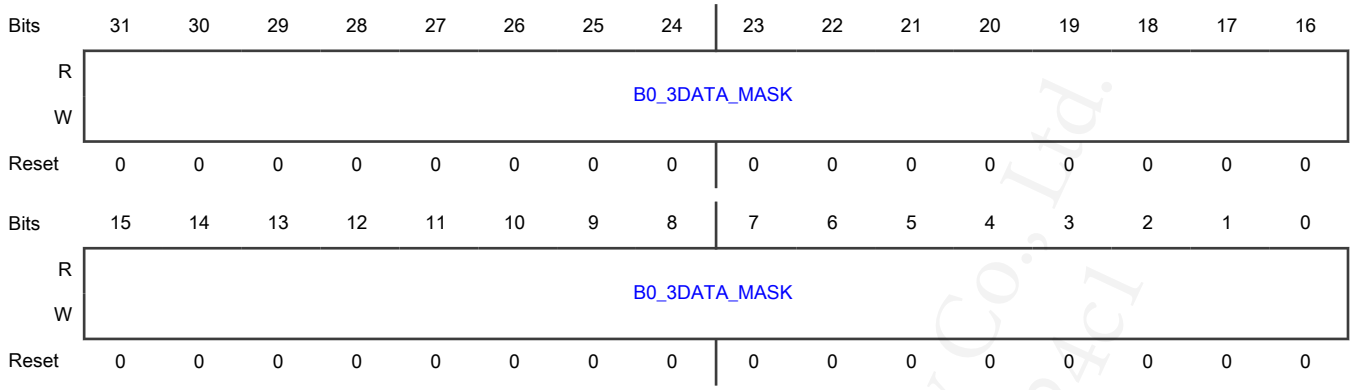
Register	Offset
EICHD0_WORD1	104h
EICHD1_WORD1	144h
EICHD2_WORD1	184h
EICHD3_WORD1	1C4h
EICHD4_WORD1	204h
EICHD5_WORD1	244h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b251-390a3ab624c1

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.7.5 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD5_WORD2)

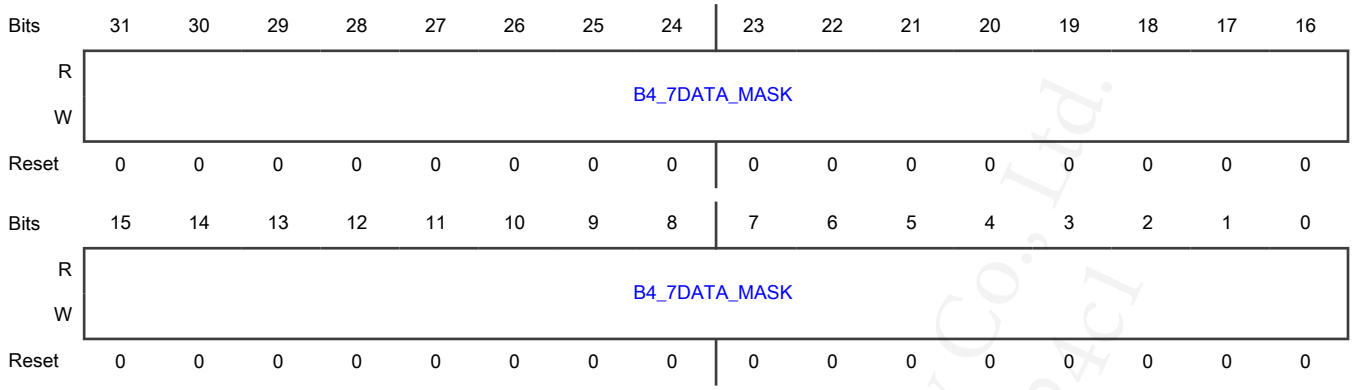
Offset

Register	Offset
EICHD0_WORD2	108h
EICHD1_WORD2	148h
EICHD2_WORD2	188h
EICHD3_WORD2	1C8h
EICHD4_WORD2	208h
EICHD5_WORD2	248h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Bytes 4-7 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

73.7.6 Error Injection Channel Descriptor n, Word3 (EICHD0_WORD3 - EICHD5_WORD3)

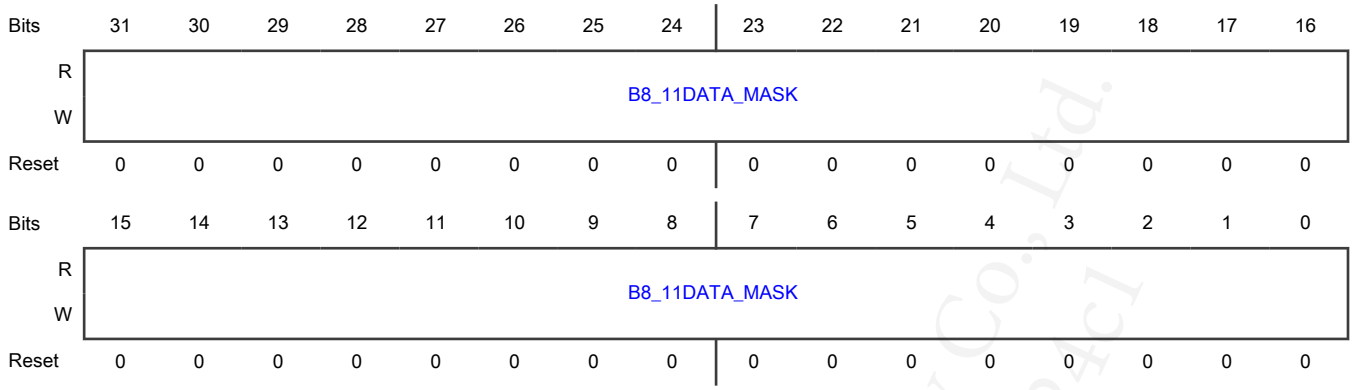
Offset

Register	Offset
EICHD0_WORD3	10Ch
EICHD1_WORD3	14Ch
EICHD2_WORD3	18Ch
EICHD3_WORD3	1CCh
EICHD4_WORD3	20Ch
EICHD5_WORD3	24Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to bytes 8–11 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Bytes 8-11 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 8-11 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 8-11 on the read data bus is inverted.

73.7.7 Error Injection Channel Descriptor n, Word4 (EICHD0_WORD4 - EICHD5_WORD4)

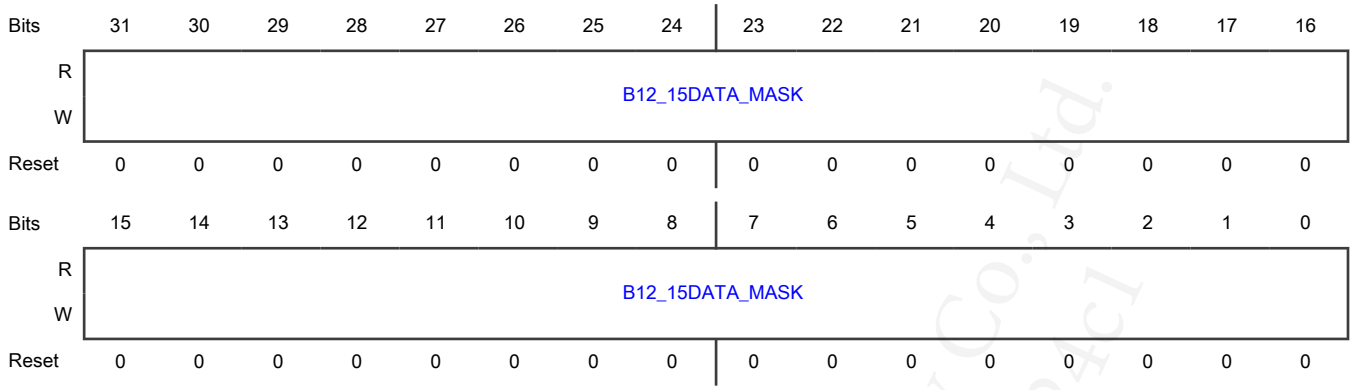
Offset

Register	Offset
EICHD0_WORD4	110h
EICHD1_WORD4	150h
EICHD2_WORD4	190h
EICHD3_WORD4	1D0h
EICHD4_WORD4	210h
EICHD5_WORD4	250h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to bytes 12–15 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Bytes 12-15 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 12-15 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 12-15 on the read data bus is inverted.

73.7.8 Error Injection Channel Descriptor n, Word5 (EICHD0_WORD5 - EICHD5_WORD5)

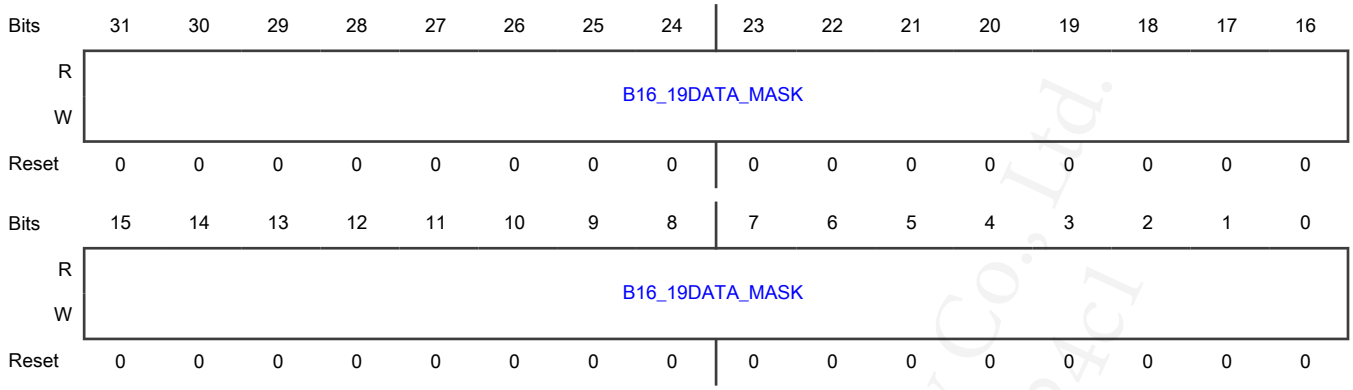
Offset

Register	Offset
EICHD0_WORD5	114h
EICHD1_WORD5	154h
EICHD2_WORD5	194h
EICHD3_WORD5	1D4h
EICHD4_WORD5	214h
EICHD5_WORD5	254h

Function

The sixth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B16_19DATA_MASK correspond to bytes 16–19 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B16_19DATA_MASK	Data Mask Bytes 16-19 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B16_19DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 16-19 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 16-19 on the read data bus is inverted.

73.7.9 Error Injection Channel Descriptor n, Word6 (EICHD0_WORD6 - EICHD5_WORD6)

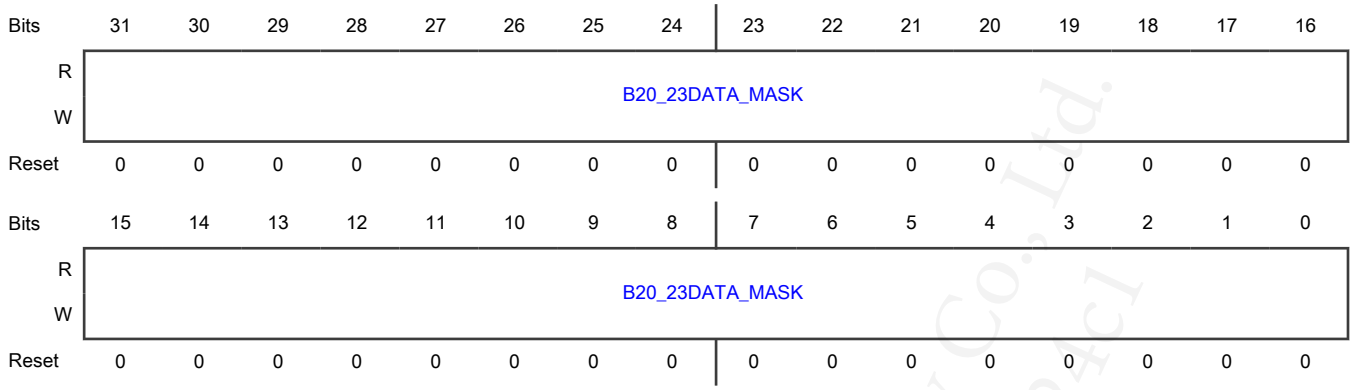
Offset

Register	Offset
EICHD0_WORD6	118h
EICHD1_WORD6	158h
EICHD2_WORD6	198h
EICHD3_WORD6	1D8h
EICHD4_WORD6	218h
EICHD5_WORD6	258h

Function

The seventh word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B20_23DATA_MASK correspond to bytes 20–23 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B20_23DATA_MASK	Data Mask Bytes 20-23 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B20_23DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 20-23 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 20-23 on the read data bus is inverted.

73.7.10 Error Injection Channel Descriptor n, Word7 (EICHD0_WORD7 - EICHD5_WORD7)

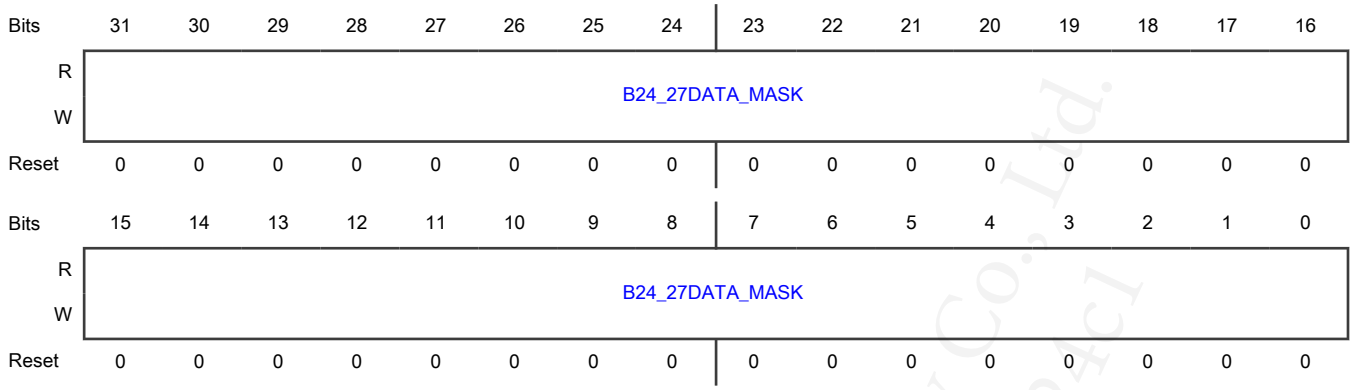
Offset

Register	Offset
EICHD0_WORD7	11Ch
EICHD1_WORD7	15Ch
EICHD2_WORD7	19Ch
EICHD3_WORD7	1DCh
EICHD4_WORD7	21Ch
EICHD5_WORD7	25Ch

Function

The eighth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B24_27DATA_MASK correspond to bytes 24–27 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B24_27DATA_MASK	Data Mask Bytes 24-27 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B24_27DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 24-27 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 24-27 on the read data bus is inverted.

73.7.11 Error Injection Channel Descriptor n, Word8 (EICHD0_WORD8 - EICHD5_WORD8)

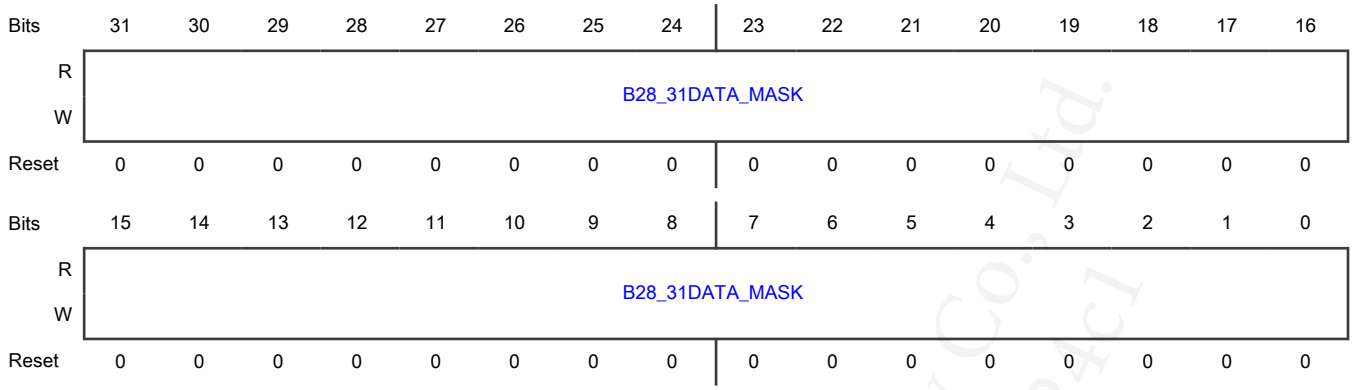
Offset

Register	Offset
EICHD0_WORD8	120h
EICHD1_WORD8	160h
EICHD2_WORD8	1A0h
EICHD3_WORD8	1E0h
EICHD4_WORD8	220h
EICHD5_WORD8	260h

Function

The ninth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B28_31DATA_MASK correspond to bytes 28–31 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B28_31DATA_MASK	Data Mask Bytes 28-31 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B28_31DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 28-31 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 28-31 on the read data bus is inverted.

73.8 EIM_3 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.

— Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.

— Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 584. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	256	255-224	223-192	191-160
1	256	255-224	223-192	191-160
2	256	255-224	223-192	191-160
3	256	255-224	223-192	191-160

Table 585. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
0	159-128	127-96	95-64	63-32	31-0
1	159-128	127-96	95-64	63-32	31-0
2	159-128	127-96	95-64	63-32	31-0
3	159-128	127-96	95-64	63-32	31-0

73.8.1 EIM_3 memory map

EIM_3 base address: 4033_3000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
10Ch	Error Injection Channel Descriptor 0, Word3 (EICHD0_WORD3)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
110h	Error Injection Channel Descriptor 0, Word4 (EICHD0_WORD4)	32	RW	0000_0000h
114h	Error Injection Channel Descriptor 0, Word5 (EICHD0_WORD5)	32	RW	0000_0000h
118h	Error Injection Channel Descriptor 0, Word6 (EICHD0_WORD6)	32	RW	0000_0000h
11Ch	Error Injection Channel Descriptor 0, Word7 (EICHD0_WORD7)	32	RW	0000_0000h
120h	Error Injection Channel Descriptor 0, Word8 (EICHD0_WORD8)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h
14Ch	Error Injection Channel Descriptor 1, Word3 (EICHD1_WORD3)	32	RW	0000_0000h
150h	Error Injection Channel Descriptor 1, Word4 (EICHD1_WORD4)	32	RW	0000_0000h
154h	Error Injection Channel Descriptor 1, Word5 (EICHD1_WORD5)	32	RW	0000_0000h
158h	Error Injection Channel Descriptor 1, Word6 (EICHD1_WORD6)	32	RW	0000_0000h
15Ch	Error Injection Channel Descriptor 1, Word7 (EICHD1_WORD7)	32	RW	0000_0000h
160h	Error Injection Channel Descriptor 1, Word8 (EICHD1_WORD8)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
18Ch	Error Injection Channel Descriptor 2, Word3 (EICHD2_WORD3)	32	RW	0000_0000h
190h	Error Injection Channel Descriptor 2, Word4 (EICHD2_WORD4)	32	RW	0000_0000h
194h	Error Injection Channel Descriptor 2, Word5 (EICHD2_WORD5)	32	RW	0000_0000h
198h	Error Injection Channel Descriptor 2, Word6 (EICHD2_WORD6)	32	RW	0000_0000h
19Ch	Error Injection Channel Descriptor 2, Word7 (EICHD2_WORD7)	32	RW	0000_0000h
1A0h	Error Injection Channel Descriptor 2, Word8 (EICHD2_WORD8)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
1CCh	Error Injection Channel Descriptor 3, Word3 (EICHD3_WORD3)	32	RW	0000_0000h
1D0h	Error Injection Channel Descriptor 3, Word4 (EICHD3_WORD4)	32	RW	0000_0000h
1D4h	Error Injection Channel Descriptor 3, Word5 (EICHD3_WORD5)	32	RW	0000_0000h
1D8h	Error Injection Channel Descriptor 3, Word6 (EICHD3_WORD6)	32	RW	0000_0000h
1DCh	Error Injection Channel Descriptor 3, Word7 (EICHD3_WORD7)	32	RW	0000_0000h
1E0h	Error Injection Channel Descriptor 3, Word8 (EICHD3_WORD8)	32	RW	0000_0000h

73.8.2 Error Injection Module Configuration Register (EIMCR)

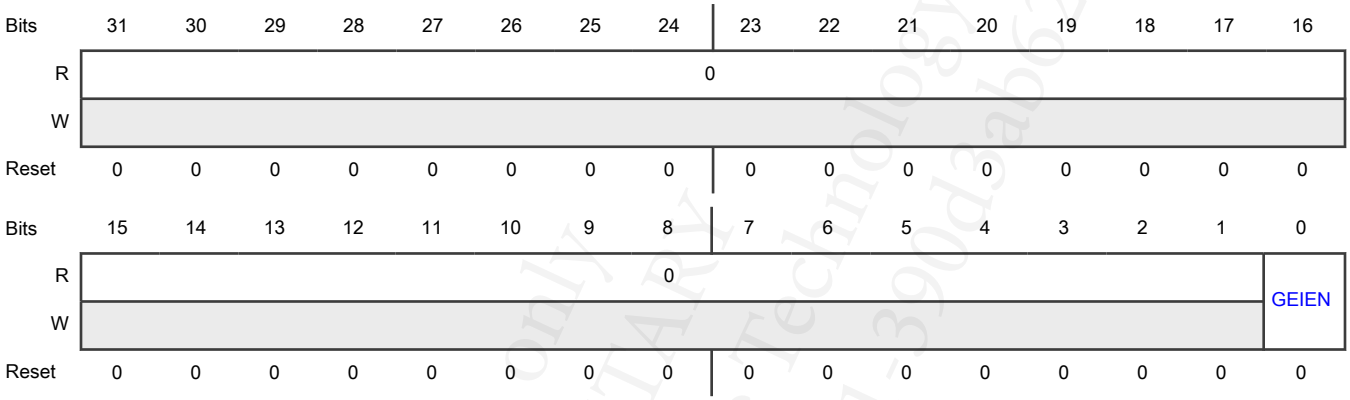
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

73.8.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-b251-390da6b624c1

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	EICH1EN	EICH2EN	EICH3EN	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

73.8.4 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD3_WORD1)

Offset

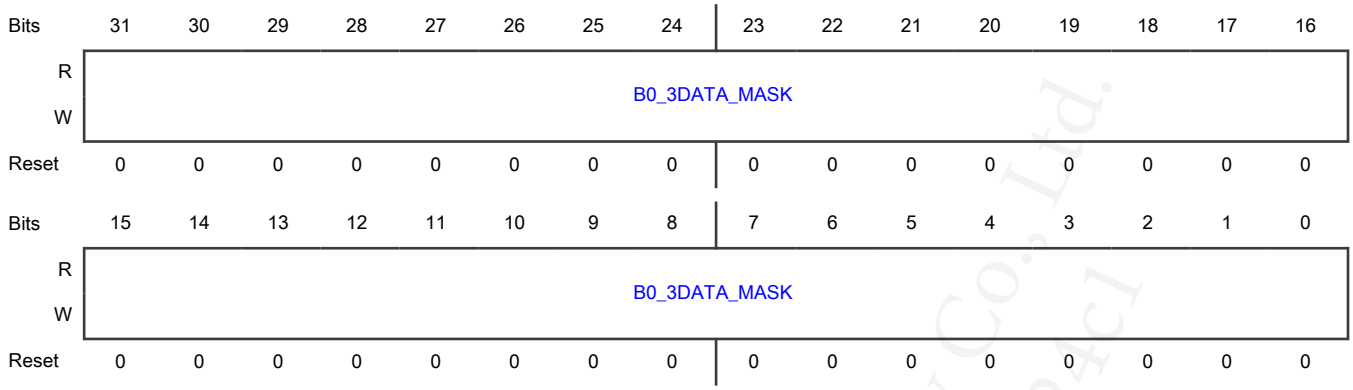
Register	Offset
EICHD0_WORD1	104h
EICHD1_WORD1	144h
EICHD2_WORD1	184h
EICHD3_WORD1	1C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Bosch Chassis Systems Control (CHJ) Automotive Technology Co., Ltd.
c8c2e54821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.8.5 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD3_WORD2)

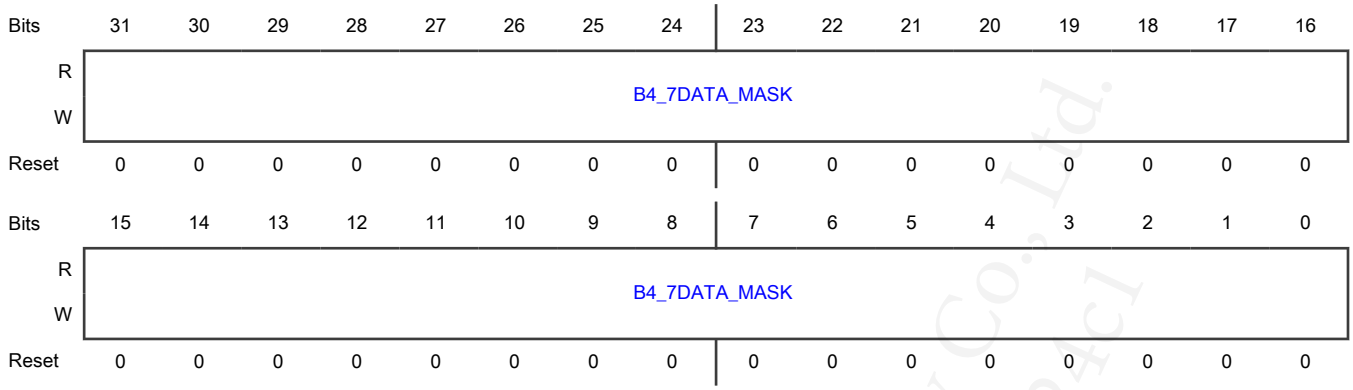
Offset

Register	Offset
EICHD0_WORD2	108h
EICHD1_WORD2	148h
EICHD2_WORD2	188h
EICHD3_WORD2	1C8h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Bytes 4-7 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

73.8.6 Error Injection Channel Descriptor n, Word3 (EICHD0_WORD3 - EICHD3_WORD3)

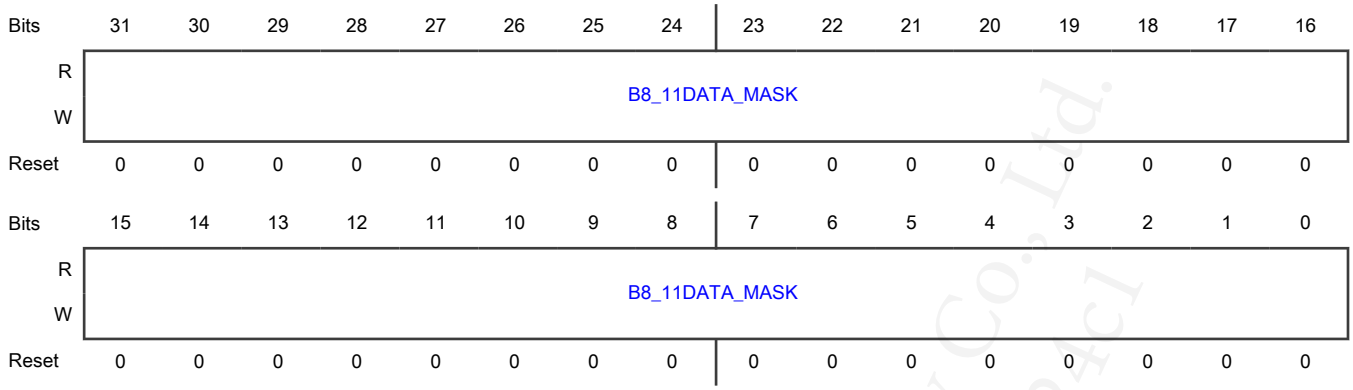
Offset

Register	Offset
EICHD0_WORD3	10Ch
EICHD1_WORD3	14Ch
EICHD2_WORD3	18Ch
EICHD3_WORD3	1CCh

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to bytes 8–11 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Bytes 8-11 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 8-11 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 8-11 on the read data bus is inverted.

73.8.7 Error Injection Channel Descriptor n, Word4 (EICHD0_WORD4 - EICHD3_WORD4)

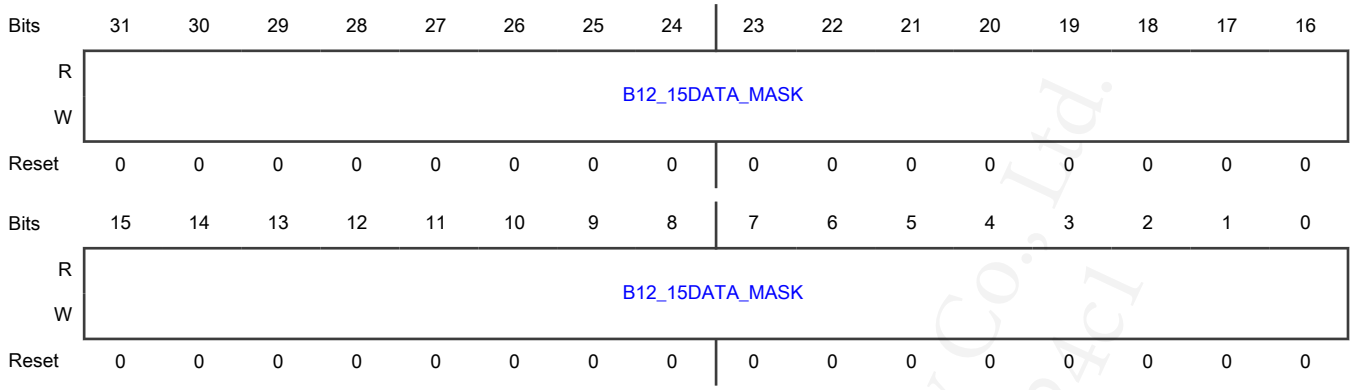
Offset

Register	Offset
EICHD0_WORD4	110h
EICHD1_WORD4	150h
EICHD2_WORD4	190h
EICHD3_WORD4	1D0h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to bytes 12–15 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Bytes 12-15 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 12-15 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 12-15 on the read data bus is inverted.

73.8.8 Error Injection Channel Descriptor n, Word5 (EICHD0_WORD5 - EICHD3_WORD5)

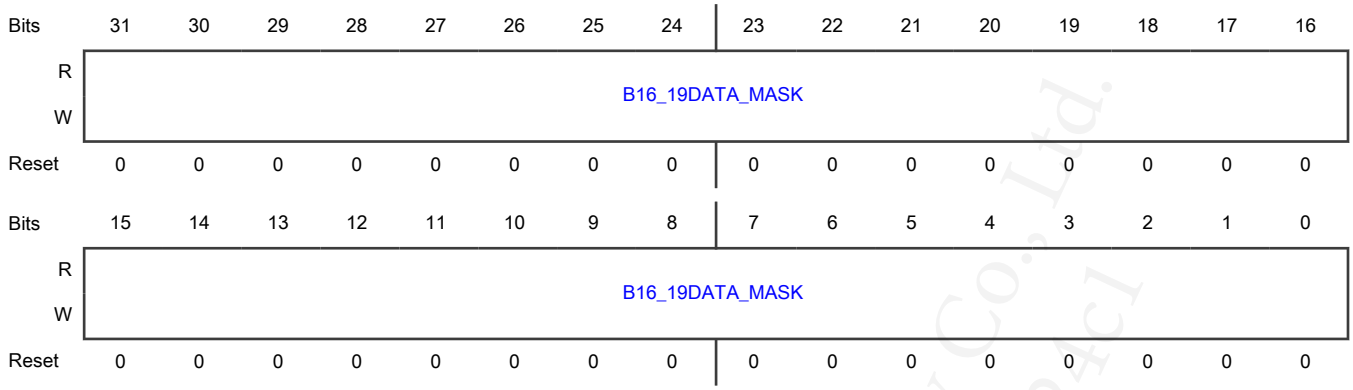
Offset

Register	Offset
EICHD0_WORD5	114h
EICHD1_WORD5	154h
EICHD2_WORD5	194h
EICHD3_WORD5	1D4h

Function

The sixth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B16_19DATA_MASK correspond to bytes 16–19 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B16_19DATA_MASK	Data Mask Bytes 16-19 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B16_19DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 16-19 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 16-19 on the read data bus is inverted.

73.8.9 Error Injection Channel Descriptor n, Word6 (EICHD0_WORD6 - EICHD3_WORD6)

Offset

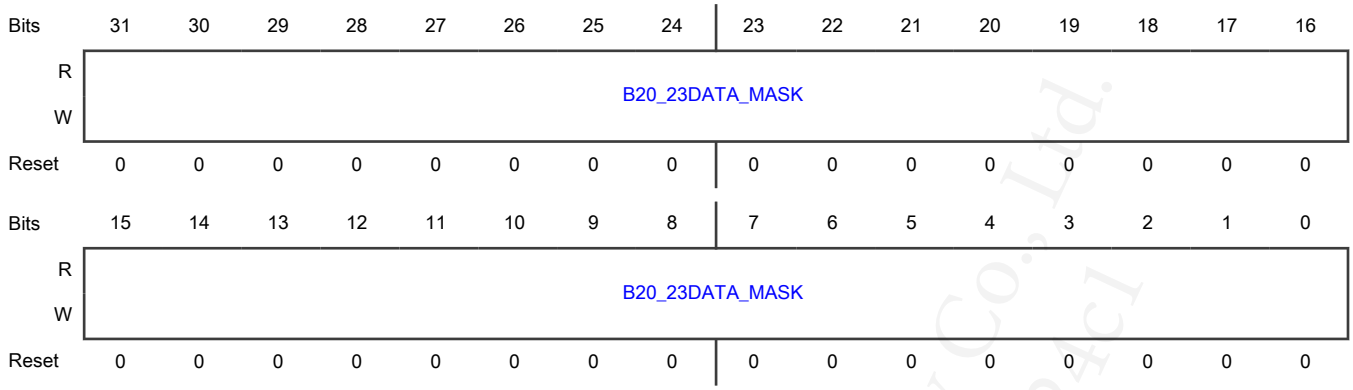
Register	Offset
EICHD0_WORD6	118h
EICHD1_WORD6	158h
EICHD2_WORD6	198h
EICHD3_WORD6	1D8h

Function

The seventh word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B20_23DATA_MASK correspond to bytes 20–23 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c80e2e5-4821-48a8-b2c1-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B20_23DATA_MASK	Data Mask Bytes 20-23 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B20_23DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 20-23 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 20-23 on the read data bus is inverted.

73.8.10 Error Injection Channel Descriptor n, Word7 (EICHD0_WORD7 - EICHD3_WORD7)

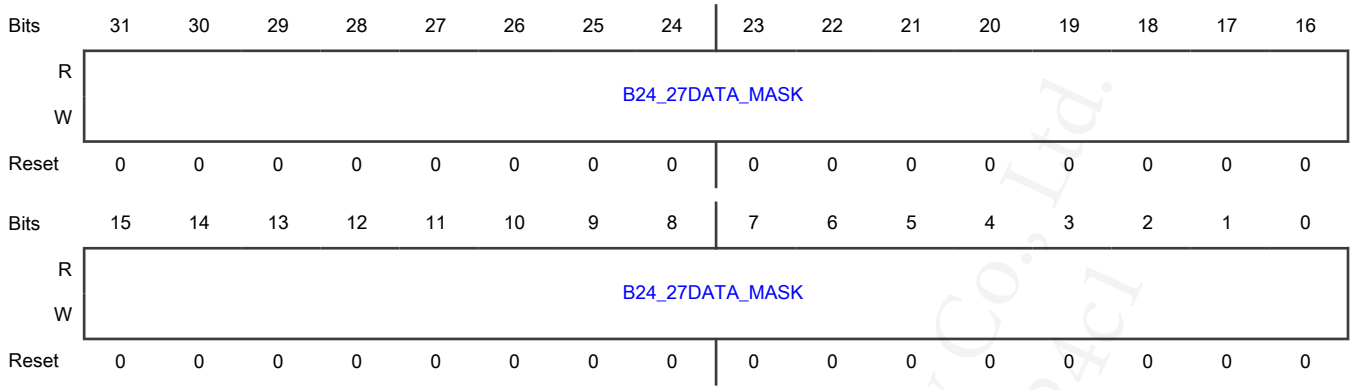
Offset

Register	Offset
EICHD0_WORD7	11Ch
EICHD1_WORD7	15Ch
EICHD2_WORD7	19Ch
EICHD3_WORD7	1DCh

Function

The eighth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B24_27DATA_MASK correspond to bytes 24–27 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B24_27DATA_MASK	Data Mask Bytes 24-27 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B24_27DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 24-27 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 24-27 on the read data bus is inverted.

73.8.11 Error Injection Channel Descriptor n, Word8 (EICHD0_WORD8 - EICHD3_WORD8)

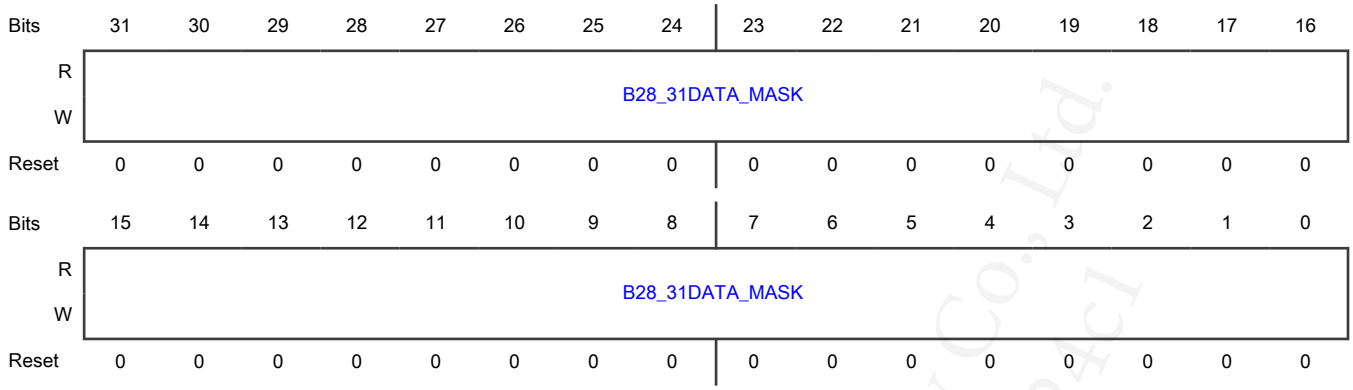
Offset

Register	Offset
EICHD0_WORD8	120h
EICHD1_WORD8	160h
EICHD2_WORD8	1A0h
EICHD3_WORD8	1E0h

Function

The ninth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B28_31DATA_MASK correspond to bytes 28–31 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B28_31DATA_MASK	Data Mask Bytes 28-31 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B28_31DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 28-31 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 28-31 on the read data bus is inverted.

73.9 EIM_LLCE register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 128-bit (16-byte) structure, composed of four 32-bit words, in the EIM programming model. Unused words are not documented.

— Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.

— Word1-3 (EICHD*n*_WORD1-3), if present, define the data mask. Word2 and Word3 are present only when required by the total width of the channel's data mask. See Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following table shows the total width of DATA_MASK and the distribution of its bits across the WORD registers.

Table 586. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	32	31-0	—	—
1	32	31-0	—	—
3	32	31-0	—	—
4	32	31-0	—	—
5	32	31-0	—	—
6	32	31-0	—	—

73.9.1 EIM_LLCE memory map

EIM_LLCE base address: 4405_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
100h	Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
140h	Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
1C0h	Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
200h	Error Injection Channel Descriptor 4, Word0 (EICHD4_WORD0)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
240h	Error Injection Channel Descriptor 5, Word0 (EICHD5_WORD0)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
280h	Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0)	32	RW	0000_0000h
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h

73.9.2 Error Injection Module Configuration Register (EIMCR)

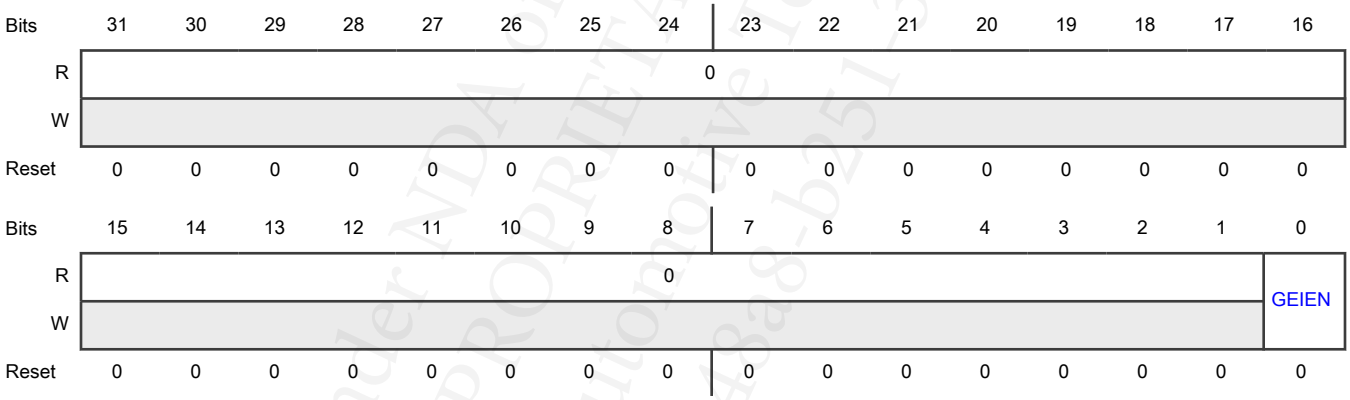
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8c462e5-4821-48a8-b251-390d3ab624c1

73.9.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	EICH1EN	0	EICH3EN	EICH4EN	EICH5EN	EICH6EN	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable

Table continues on the next page...

Table continued from the previous page...

Field	Function
	This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 —	Reserved
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
0 —	Reserved

73.9.4 Error Injection Channel Descriptor n, Word0 (EICHD0_WORD0 - EICHD6_WORD0)

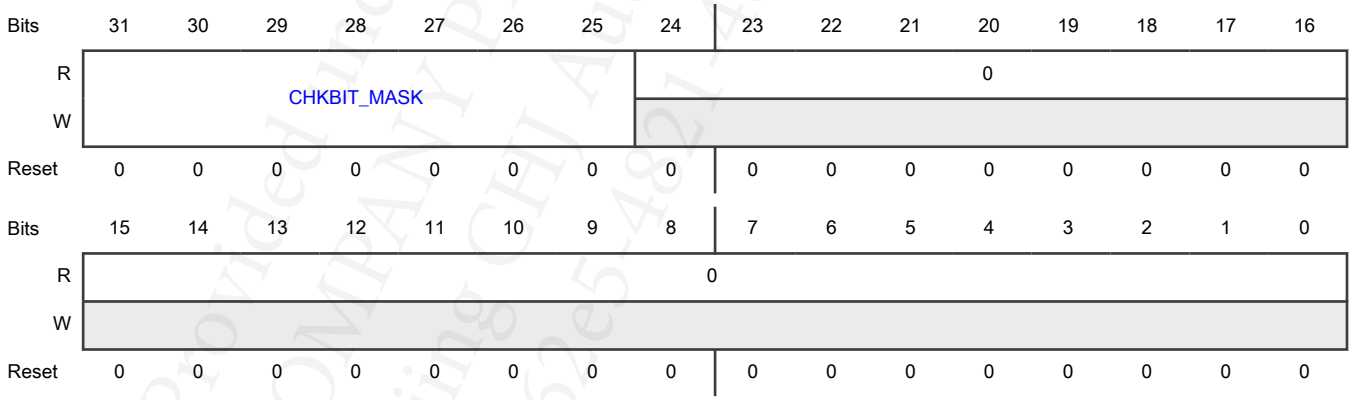
Offset

Register	Offset
EICHD0_WORD0	100h
EICHD1_WORD0	140h
EICHD3_WORD0	1C0h
EICHD4_WORD0	200h
EICHD5_WORD0	240h
EICHD6_WORD0	280h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c46e54821f48a8b0251-390d3ab6240

Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.9.5 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD6_WORD1)

Offset

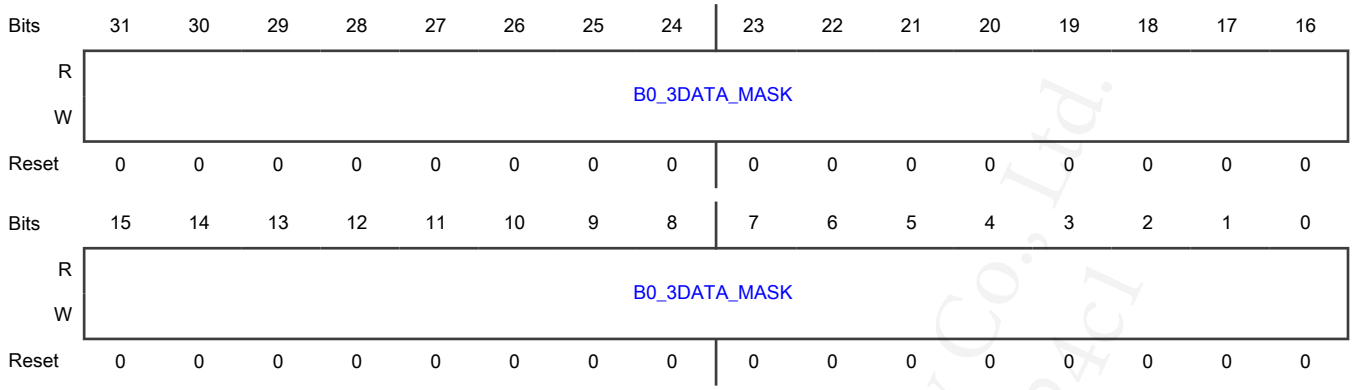
Register	Offset
EICHD0_WORD1	104h
EICHD1_WORD1	144h
EICHD3_WORD1	1C4h
EICHD4_WORD1	204h
EICHD5_WORD1	244h
EICHD6_WORD1	284h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-4821-48a8-b251-590d8ab624c1

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.10 EIM_MISC register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.

— Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.

— Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 587. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	72	71-64	63-32	31-0
1	72	71-64	63-32	31-0
2	104	103-96	95-64	63-32
3	104	103-96	95-64	63-32
4	104	103-96	95-64	63-32
5	39	38-32	31-0	—
6	87	86-64	63-32	31-0
7	96	95-64	63-32	31-0
8	96	95-64	63-32	31-0
9	46	45-32	31-0	—
10	86	85-64	63-32	31-0
11	86	85-64	63-32	31-0
12	88	87-64	63-32	31-0
13	86	85-64	63-32	31-0
14	86	85-64	63-32	31-0
15	47	46-32	31-0	—
16	16	15-0	—	—

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-481-48a8-b251-29013ab64c1

Table 588. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
2	31-0	—	—	—	—
3	31-0	—	—	—	—
4	31-0	—	—	—	—

73.10.1 EIM_MISC memory map

EIM_MISC base address: 4404_C000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
10Ch	Error Injection Channel Descriptor 0, Word3 (EICHD0_WORD3)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h
14Ch	Error Injection Channel Descriptor 1, Word3 (EICHD1_WORD3)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
18Ch	Error Injection Channel Descriptor 2, Word3 (EICHD2_WORD3)	32	RW	0000_0000h
190h	Error Injection Channel Descriptor 2, Word4 (EICHD2_WORD4)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
1CCh	Error Injection Channel Descriptor 3, Word3 (EICHD3_WORD3)	32	RW	0000_0000h
1D0h	Error Injection Channel Descriptor 3, Word4 (EICHD3_WORD4)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
208h	Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)	32	RW	0000_0000h
20Ch	Error Injection Channel Descriptor 4, Word3 (EICHD4_WORD3)	32	RW	0000_0000h
210h	Error Injection Channel Descriptor 4, Word4 (EICHD4_WORD4)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
248h	Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h
288h	Error Injection Channel Descriptor 6, Word2 (EICHD6_WORD2)	32	RW	0000_0000h
28Ch	Error Injection Channel Descriptor 6, Word3 (EICHD6_WORD3)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
2C8h	Error Injection Channel Descriptor 7, Word2 (EICHD7_WORD2)	32	RW	0000_0000h
2CCh	Error Injection Channel Descriptor 7, Word3 (EICHD7_WORD3)	32	RW	0000_0000h
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
308h	Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)	32	RW	0000_0000h
30Ch	Error Injection Channel Descriptor 8, Word3 (EICHD8_WORD3)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
348h	Error Injection Channel Descriptor 9, Word2 (EICHD9_WORD2)	32	RW	0000_0000h
384h	Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)	32	RW	0000_0000h
388h	Error Injection Channel Descriptor 10, Word2 (EICHD10_WORD2)	32	RW	0000_0000h
38Ch	Error Injection Channel Descriptor 10, Word3 (EICHD10_WORD3)	32	RW	0000_0000h
3C4h	Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)	32	RW	0000_0000h
3C8h	Error Injection Channel Descriptor 11, Word2 (EICHD11_WORD2)	32	RW	0000_0000h
3CCh	Error Injection Channel Descriptor 11, Word3 (EICHD11_WORD3)	32	RW	0000_0000h
404h	Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)	32	RW	0000_0000h
408h	Error Injection Channel Descriptor 12, Word2 (EICHD12_WORD2)	32	RW	0000_0000h
40Ch	Error Injection Channel Descriptor 12, Word3 (EICHD12_WORD3)	32	RW	0000_0000h
444h	Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)	32	RW	0000_0000h
448h	Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)	32	RW	0000_0000h
44Ch	Error Injection Channel Descriptor 13, Word3 (EICHD13_WORD3)	32	RW	0000_0000h
484h	Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)	32	RW	0000_0000h
488h	Error Injection Channel Descriptor 14, Word2 (EICHD14_WORD2)	32	RW	0000_0000h
48Ch	Error Injection Channel Descriptor 14, Word3 (EICHD14_WORD3)	32	RW	0000_0000h
4C4h	Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)	32	RW	0000_0000h
4C8h	Error Injection Channel Descriptor 15, Word2 (EICHD15_WORD2)	32	RW	0000_0000h
504h	Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)	32	RW	0000_0000h

73.10.2 Error Injection Module Configuration Register (EIMCR)

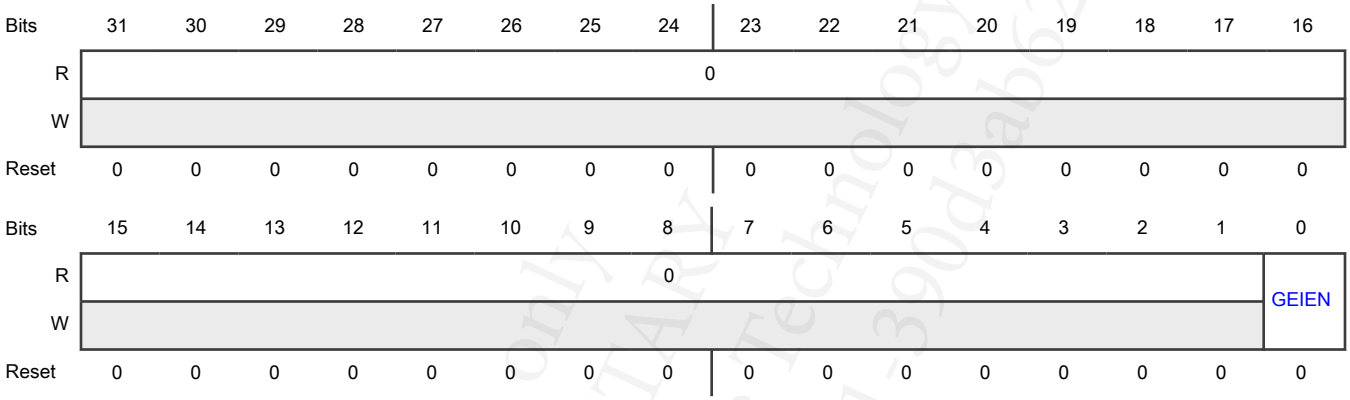
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

73.10.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390daab624c1

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	EICH1EN	EICH2EN	EICH3EN	EICH4EN	EICH5EN	EICH6EN	EICH7EN	EICH8EN	EICH9EN	EICH10EN	EICH11EN	EICH12EN	EICH13EN	EICH14EN	EICH15EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EICH16EN	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9
21 EICH10EN	Error Injection Channel 10 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 10 1b - Error injection is enabled on Error Injection Channel 10
20 EICH11EN	Error Injection Channel 11 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 11 1b - Error injection is enabled on Error Injection Channel 11
19 EICH12EN	Error Injection Channel 12 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 12 1b - Error injection is enabled on Error Injection Channel 12

Table continues on the next page...

Table continued from the previous page...

Field	Function
18 EICH13EN	Error Injection Channel 13 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 13 1b - Error injection is enabled on Error Injection Channel 13
17 EICH14EN	Error Injection Channel 14 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 14 1b - Error injection is enabled on Error Injection Channel 14
16 EICH15EN	Error Injection Channel 15 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 15 1b - Error injection is enabled on Error Injection Channel 15
15 EICH16EN	Error Injection Channel 16 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 16 1b - Error injection is enabled on Error Injection Channel 16
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
1 —	Reserved
0 —	Reserved

73.10.4 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD4_WORD1)

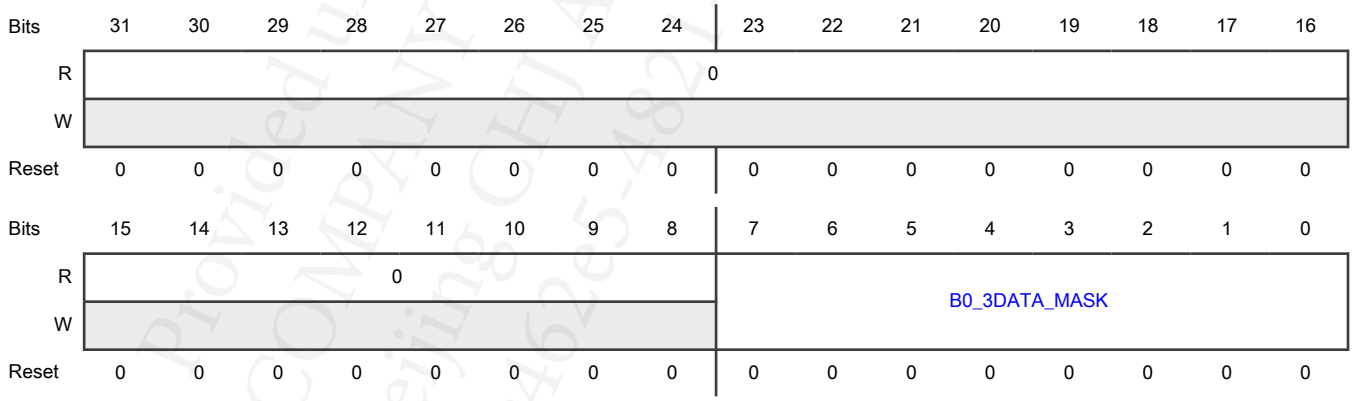
Offset

Register	Offset
EICHD0_WORD1	104h
EICHD1_WORD1	144h
EICHD2_WORD1	184h
EICHD3_WORD1	1C4h
EICHD4_WORD1	204h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482a-4b25-390d3ab624c1

Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.10.5 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD15_WORD2)

Offset

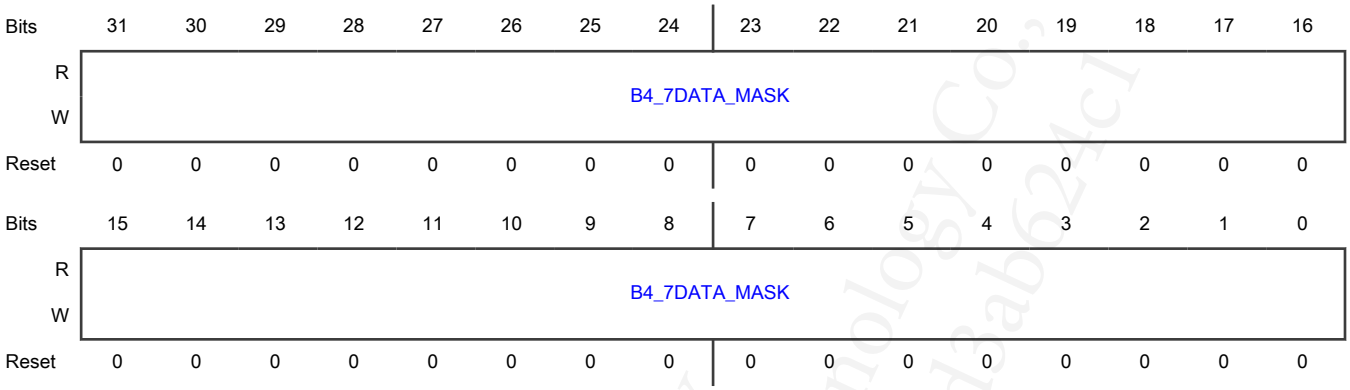
Register	Offset
EICHD0_WORD2	108h
EICHD1_WORD2	148h
EICHD2_WORD2	188h
EICHD3_WORD2	1C8h
EICHD4_WORD2	208h
EICHD5_WORD2	248h
EICHD6_WORD2	288h
EICHD7_WORD2	2C8h
EICHD8_WORD2	308h
EICHD9_WORD2	348h
EICHD10_WORD2	388h
EICHD11_WORD2	3C8h
EICHD12_WORD2	408h
EICHD13_WORD2	448h
EICHD14_WORD2	488h
EICHD15_WORD2	4C8h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-39018b6240

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Bytes 4-7 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

73.10.6 Error Injection Channel Descriptor n, Word3 (EICHD0_WORD3 - EICHD14_WORD3)

Offset

Register	Offset
EICHD0_WORD3	10Ch
EICHD1_WORD3	14Ch
EICHD2_WORD3	18Ch
EICHD3_WORD3	1CCh
EICHD4_WORD3	20Ch
EICHD6_WORD3	28Ch

Table continues on the next page...

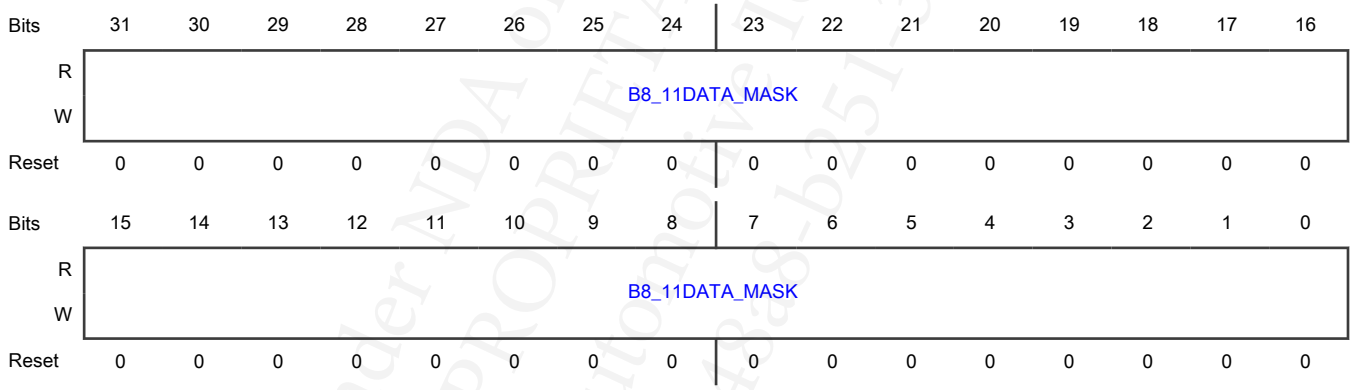
Table continued from the previous page...

Register	Offset
EICHD7_WORD3	2CCh
EICHD8_WORD3	30Ch
EICHD10_WORD3	38Ch
EICHD11_WORD3	3CCh
EICHD12_WORD3	40Ch
EICHD13_WORD3	44Ch
EICHD14_WORD3	48Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to bytes 8–11 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Bytes 8-11 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 8-11 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 8-11 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
e8462e-4821-48a8-b251-3900a5b624c1

73.10.7 Error Injection Channel Descriptor n, Word4 (EICHD2_WORD4 - EICHD4_WORD4)

Offset

Register	Offset
EICHD2_WORD4	190h
EICHD3_WORD4	1D0h
EICHD4_WORD4	210h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to bytes 12–15 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B12_15DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B12_15DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Bytes 12-15 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 12-15 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 12-15 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c42e-4c91-48a8-b251-390d3ab24c1

73.10.8 Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)

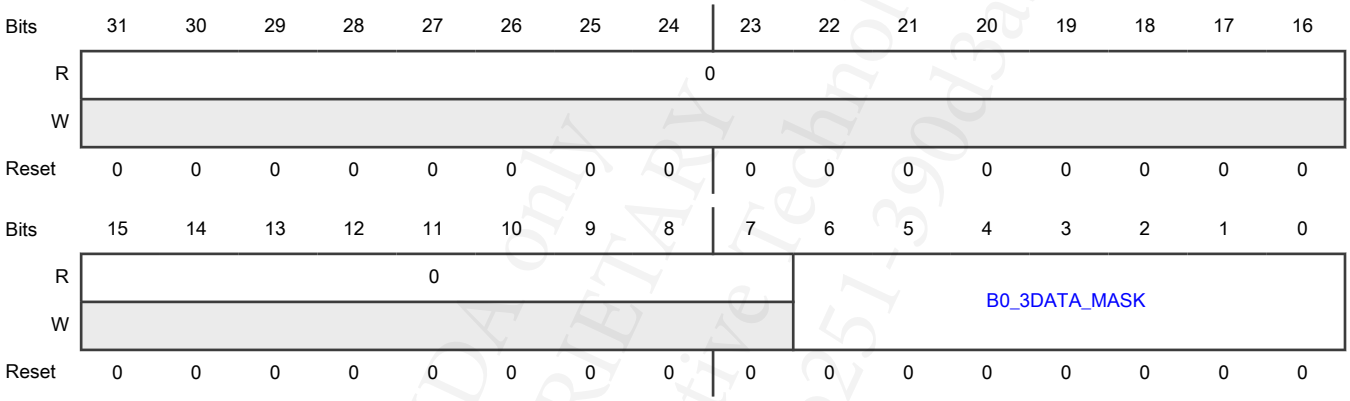
Offset

Register	Offset
EICHD5_WORD1	244h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-7 —	Reserved
6-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJ Automotive Technology Co., Ltd.
c8c8b6e-421-48a8-b251-390d3ab6201

73.10.9 Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)

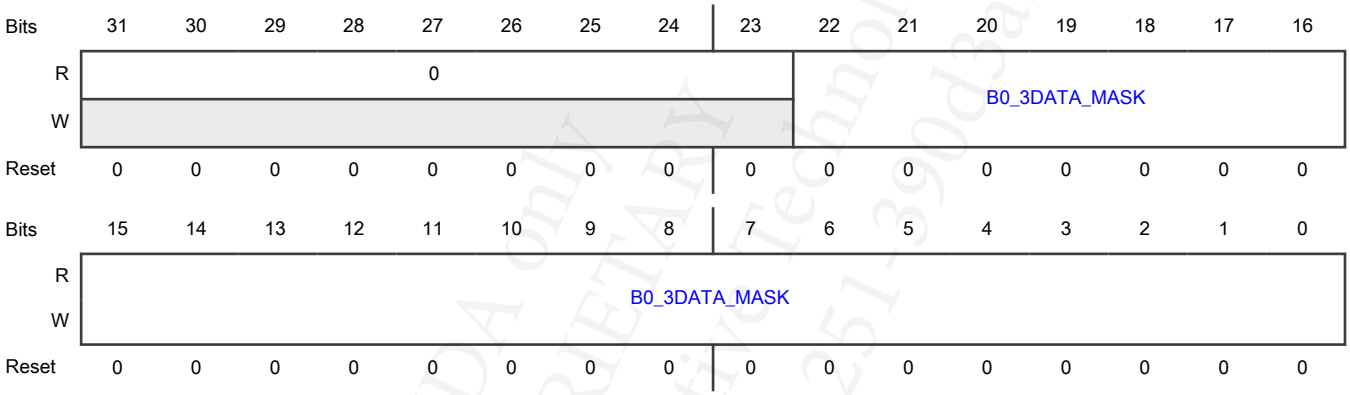
Offset

Register	Offset
EICHD6_WORD1	284h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-23 —	Reserved
22-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing C&I Automotive Technology Co., Ltd.
c8a6e421-48a8-b251-390dab6201

73.10.10 Error Injection Channel Descriptor n, Word1 (EICHD7_WORD1 - EICHD8_WORD1)

Offset

Register	Offset
EICHD7_WORD1	2C4h
EICHD8_WORD1	304h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-8187-48a8-b251-590d3ab64c1

73.10.11 Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)

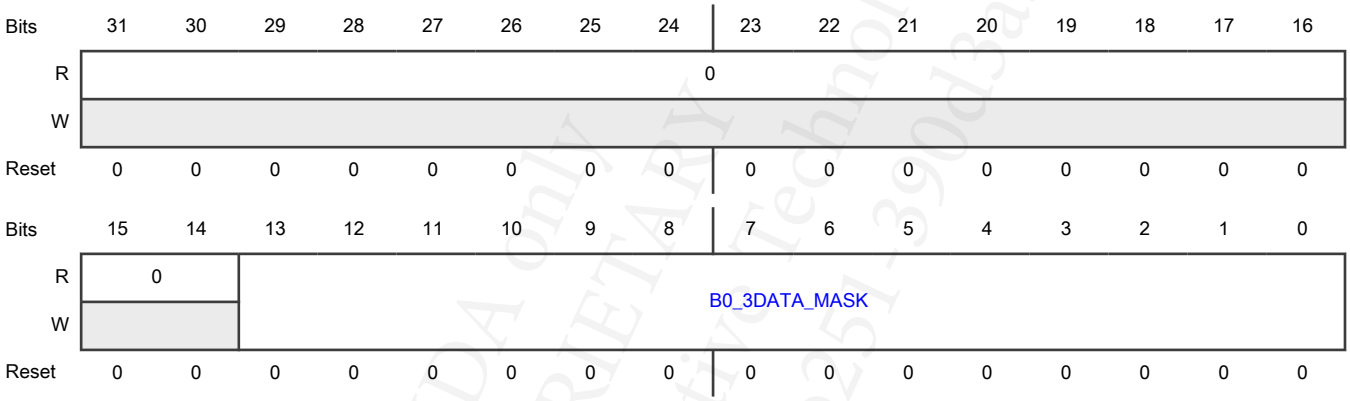
Offset

Register	Offset
EICHD9_WORD1	344h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-14 —	Reserved
13-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJ Automotive Technology Co., Ltd.
c8c8b6e-421-48a8-b251-390d3ab6201

73.10.12 Error Injection Channel Descriptor n, Word1 (EICHD10_WORD1 - EICHD11_WORD1)

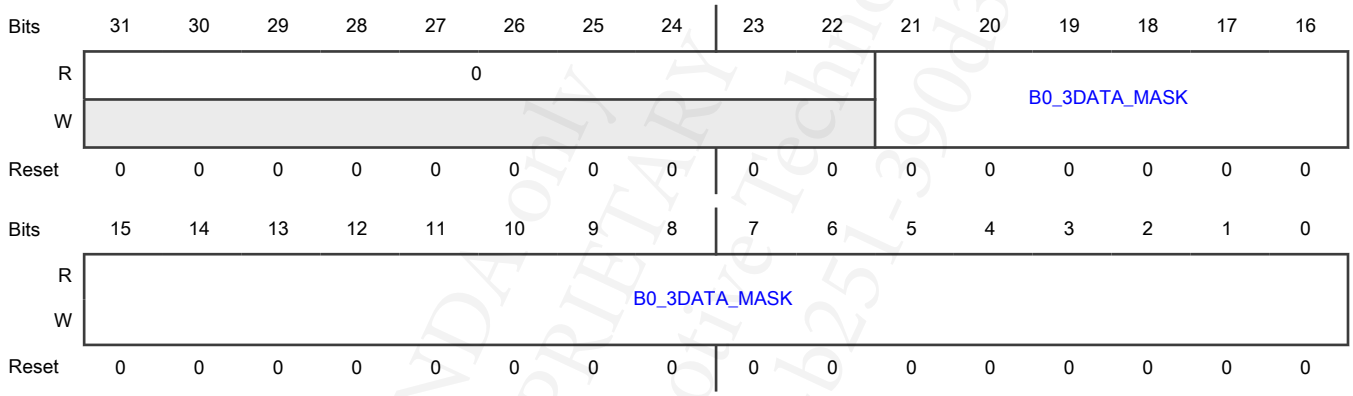
Offset

Register	Offset
EICHD10_WORD1	384h
EICHD11_WORD1	3C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-22 —	Reserved
21-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Bain, GHJ Automotive Technology Co., Ltd.
c8046c5-821-48a8-b251-290d3ab64c1

73.10.13 Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)

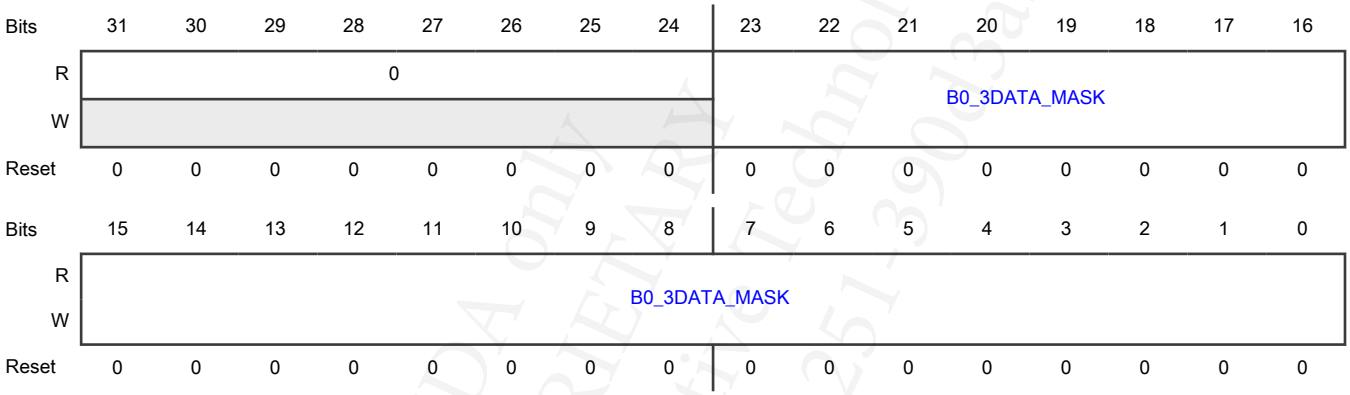
Offset

Register	Offset
EICHD12_WORD1	404h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 —	Reserved
23-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CCI Automotive Technology Co., Ltd.
c8a6e421-48a8-b251-390d3ab6201

73.10.14 Error Injection Channel Descriptor n, Word1 (EICHD13_WORD1 - EICHD14_WORD1)

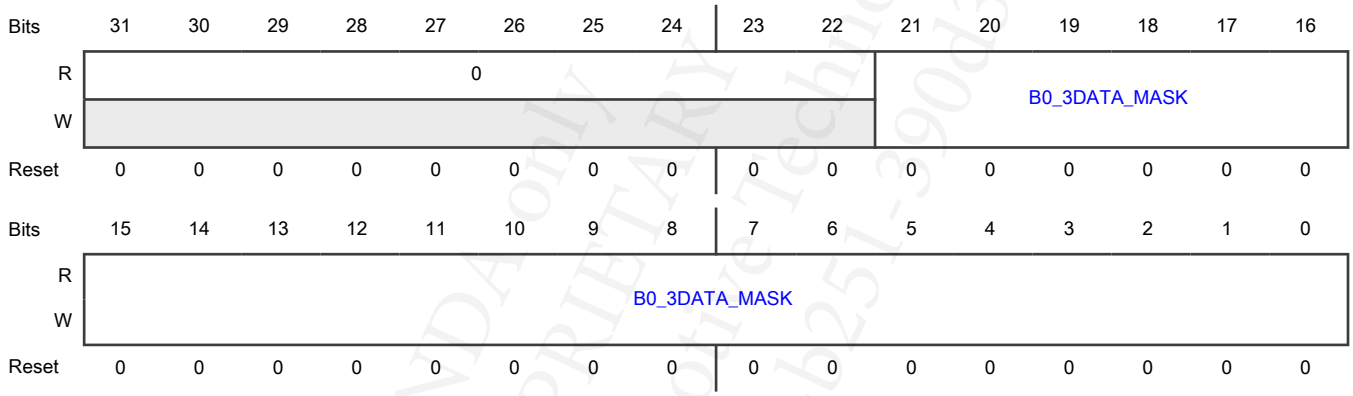
Offset

Register	Offset
EICHD13_WORD1	444h
EICHD14_WORD1	484h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-22 —	Reserved
21-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Bain, GHJ Automotive Technology Co., Ltd.
c8d6b5-821-48a8-b251-290d3ab64c1

73.10.15 Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)

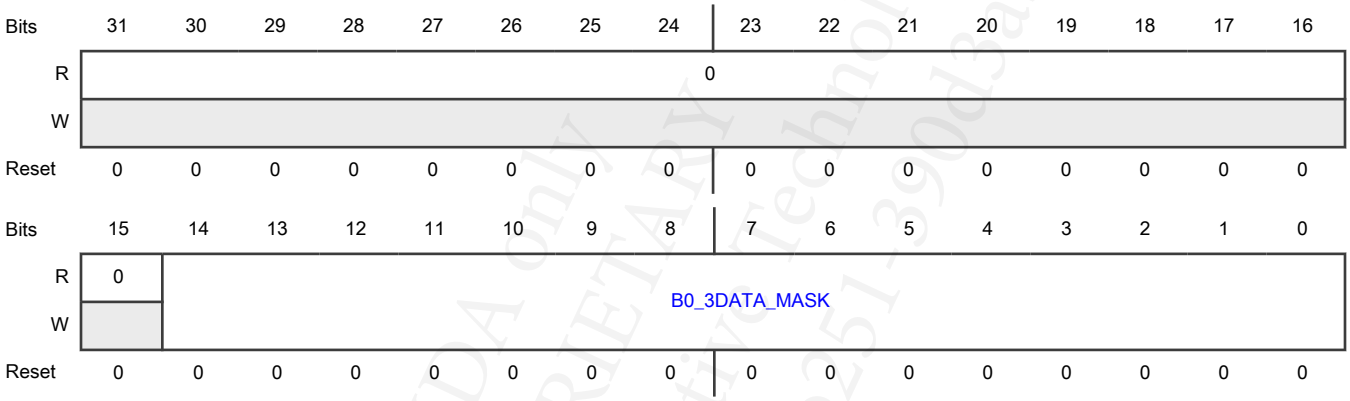
Offset

Register	Offset
EICHD15_WORD1	4C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-15 —	Reserved
14-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJ Automotive Technology Co., Ltd.
c8c8b6e-421-48a8-b251-390d3ab6201

73.10.16 Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)

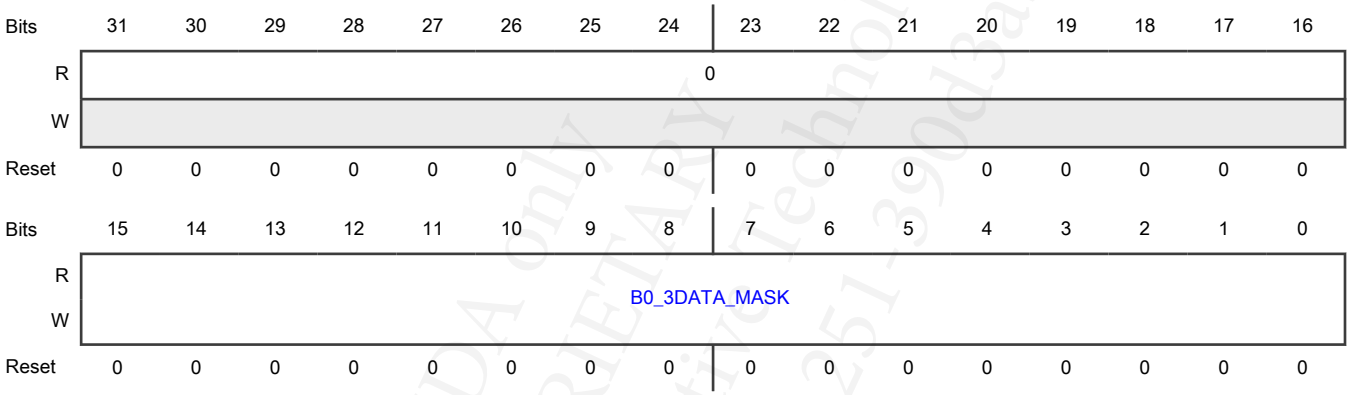
Offset

Register	Offset
EICHD16_WORD1	504h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.11 EIM_PFE0 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.
  - Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 589. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	32	31-0	—	—
1	64	63-32	31-0	—
2	124	123-96	95-64	63-32
3	48	47-32	31-0	—
4	16	15-0	—	—
5	32	31-0	—	—
6	32	31-0	—	—
7	32	31-0	—	—
8	48	47-32	31-0	—

Table continues on the next page...

Table 589. Error injection channel descriptor: DATA_MASK details (continued)

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
9	16	15-0	—	—
10	32	31-0	—	—
11	32	31-0	—	—
12	32	31-0	—	—
13	48	47-32	31-0	—
14	16	15-0	—	—
15	32	31-0	—	—
16	32	31-0	—	—
17	32	31-0	—	—
18	48	47-32	31-0	—
19	16	15-0	—	—
20	32	31-0	—	—
21	32	31-0	—	—
22	32	31-0	—	—
23	48	47-32	31-0	—
24	16	15-0	—	—
25	32	31-0	—	—
26	32	31-0	—	—
27	32	31-0	—	—
28	48	47-32	31-0	—
29	16	15-0	—	—
30	32	31-0	—	—
31	32	31-0	—	—

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-48a8-1251-3a0d3ab624c1

Table 590. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
2	31-0	—	—	—	—

73.11.1 EIM_PFE0 memory map

EIM_PFE0 base address: 4405_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
100h	Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
140h	Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h
180h	Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
18Ch	Error Injection Channel Descriptor 2, Word3 (EICHD2_WORD3)	32	RW	0000_0000h
190h	Error Injection Channel Descriptor 2, Word4 (EICHD2_WORD4)	32	RW	0000_0000h
1C0h	Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
200h	Error Injection Channel Descriptor 4, Word0 (EICHD4_WORD0)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
240h	Error Injection Channel Descriptor 5, Word0 (EICHD5_WORD0)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
280h	Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0)	32	RW	0000_0000h
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h
2C0h	Error Injection Channel Descriptor 7, Word0 (EICHD7_WORD0)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
300h	Error Injection Channel Descriptor 8, Word0 (EICHD8_WORD0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
308h	Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)	32	RW	0000_0000h
340h	Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
380h	Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0)	32	RW	0000_0000h
384h	Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)	32	RW	0000_0000h
3C0h	Error Injection Channel Descriptor 11, Word0 (EICHD11_WORD0)	32	RW	0000_0000h
3C4h	Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)	32	RW	0000_0000h
400h	Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0)	32	RW	0000_0000h
404h	Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)	32	RW	0000_0000h
440h	Error Injection Channel Descriptor 13, Word0 (EICHD13_WORD0)	32	RW	0000_0000h
444h	Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)	32	RW	0000_0000h
448h	Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)	32	RW	0000_0000h
480h	Error Injection Channel Descriptor 14, Word0 (EICHD14_WORD0)	32	RW	0000_0000h
484h	Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)	32	RW	0000_0000h
4C0h	Error Injection Channel Descriptor 15, Word0 (EICHD15_WORD0)	32	RW	0000_0000h
4C4h	Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)	32	RW	0000_0000h
500h	Error Injection Channel Descriptor 16, Word0 (EICHD16_WORD0)	32	RW	0000_0000h
504h	Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)	32	RW	0000_0000h
540h	Error Injection Channel Descriptor 17, Word0 (EICHD17_WORD0)	32	RW	0000_0000h
544h	Error Injection Channel Descriptor 17, Word1 (EICHD17_WORD1)	32	RW	0000_0000h
580h	Error Injection Channel Descriptor 18, Word0 (EICHD18_WORD0)	32	RW	0000_0000h
584h	Error Injection Channel Descriptor 18, Word1 (EICHD18_WORD1)	32	RW	0000_0000h
588h	Error Injection Channel Descriptor 18, Word2 (EICHD18_WORD2)	32	RW	0000_0000h
5C0h	Error Injection Channel Descriptor 19, Word0 (EICHD19_WORD0)	32	RW	0000_0000h
5C4h	Error Injection Channel Descriptor 19, Word1 (EICHD19_WORD1)	32	RW	0000_0000h
600h	Error Injection Channel Descriptor 20, Word0 (EICHD20_WORD0)	32	RW	0000_0000h
604h	Error Injection Channel Descriptor 20, Word1 (EICHD20_WORD1)	32	RW	0000_0000h
640h	Error Injection Channel Descriptor 21, Word0 (EICHD21_WORD0)	32	RW	0000_0000h
644h	Error Injection Channel Descriptor 21, Word1 (EICHD21_WORD1)	32	RW	0000_0000h
680h	Error Injection Channel Descriptor 22, Word0 (EICHD22_WORD0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
684h	Error Injection Channel Descriptor 22, Word1 (EICHD22_WORD1)	32	RW	0000_0000h
6C0h	Error Injection Channel Descriptor 23, Word0 (EICHD23_WORD0)	32	RW	0000_0000h
6C4h	Error Injection Channel Descriptor 23, Word1 (EICHD23_WORD1)	32	RW	0000_0000h
6C8h	Error Injection Channel Descriptor 23, Word2 (EICHD23_WORD2)	32	RW	0000_0000h
700h	Error Injection Channel Descriptor 24, Word0 (EICHD24_WORD0)	32	RW	0000_0000h
704h	Error Injection Channel Descriptor 24, Word1 (EICHD24_WORD1)	32	RW	0000_0000h
740h	Error Injection Channel Descriptor 25, Word0 (EICHD25_WORD0)	32	RW	0000_0000h
744h	Error Injection Channel Descriptor 25, Word1 (EICHD25_WORD1)	32	RW	0000_0000h
780h	Error Injection Channel Descriptor 26, Word0 (EICHD26_WORD0)	32	RW	0000_0000h
784h	Error Injection Channel Descriptor 26, Word1 (EICHD26_WORD1)	32	RW	0000_0000h
7C0h	Error Injection Channel Descriptor 27, Word0 (EICHD27_WORD0)	32	RW	0000_0000h
7C4h	Error Injection Channel Descriptor 27, Word1 (EICHD27_WORD1)	32	RW	0000_0000h
800h	Error Injection Channel Descriptor 28, Word0 (EICHD28_WORD0)	32	RW	0000_0000h
804h	Error Injection Channel Descriptor 28, Word1 (EICHD28_WORD1)	32	RW	0000_0000h
808h	Error Injection Channel Descriptor 28, Word2 (EICHD28_WORD2)	32	RW	0000_0000h
840h	Error Injection Channel Descriptor 29, Word0 (EICHD29_WORD0)	32	RW	0000_0000h
844h	Error Injection Channel Descriptor 29, Word1 (EICHD29_WORD1)	32	RW	0000_0000h
880h	Error Injection Channel Descriptor 30, Word0 (EICHD30_WORD0)	32	RW	0000_0000h
884h	Error Injection Channel Descriptor 30, Word1 (EICHD30_WORD1)	32	RW	0000_0000h
8C0h	Error Injection Channel Descriptor 31, Word0 (EICHD31_WORD0)	32	RW	0000_0000h
8C4h	Error Injection Channel Descriptor 31, Word1 (EICHD31_WORD1)	32	RW	0000_0000h

73.11.2 Error Injection Module Configuration Register (EIMCR)

Offset

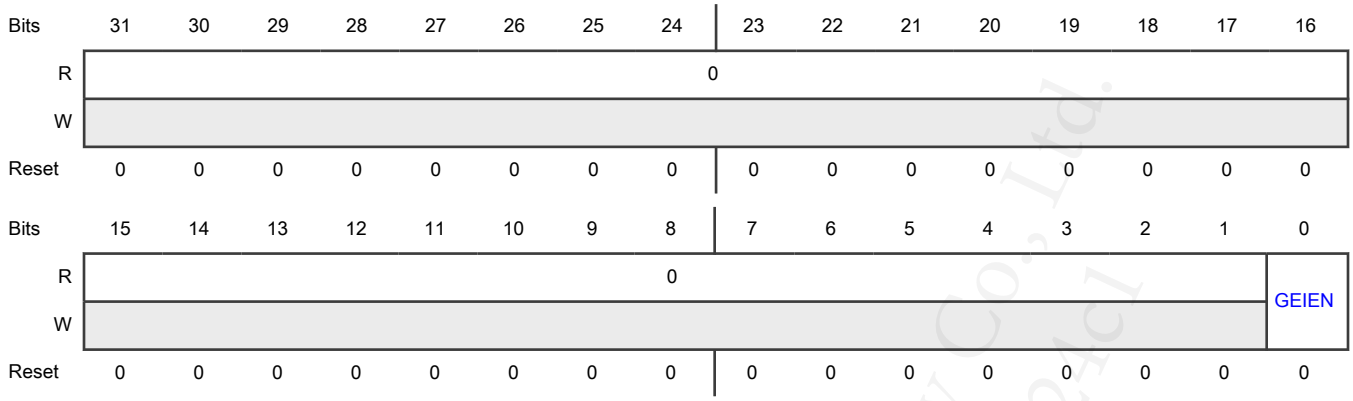
Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Provided under NXP SEMICONDUCTORS COMPANY CONFIDENTIAL Beijing CHIPHOME Technology Co., Ltd. c84625-48124 2020-09-08 08:04:51

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

73.11.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-821-48a8-b251-390d8ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0	EICH1	EICH2	EICH3	EICH4	EICH5	EICH6	EICH7	EICH8	EICH9	EICH1	EICH1	EICH1	EICH1	EICH1	EICH1
W	EN	EN	EN	EN	EN	EN	EN	EN	EN	EN	0EN	1EN	2EN	3EN	4EN	5EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EICH1	EICH1	EICH1	EICH1	EICH2	EICH2	EICH2	EICH2	EICH2	EICH2	EICH2	EICH2	EICH2	EICH2	EICH3	EICH3
W	6EN	7EN	8EN	9EN	0EN	1EN	2EN	3EN	4EN	5EN	6EN	7EN	8EN	9EN	0EN	1EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 EICH10EN	Error Injection Channel 10 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 10 1b - Error injection is enabled on Error Injection Channel 10
20 EICH11EN	Error Injection Channel 11 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 11 1b - Error injection is enabled on Error Injection Channel 11
19 EICH12EN	Error Injection Channel 12 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 12 1b - Error injection is enabled on Error Injection Channel 12
18 EICH13EN	Error Injection Channel 13 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 13 1b - Error injection is enabled on Error Injection Channel 13
17 EICH14EN	Error Injection Channel 14 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 14 1b - Error injection is enabled on Error Injection Channel 14
16 EICH15EN	Error Injection Channel 15 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 15 1b - Error injection is enabled on Error Injection Channel 15
15 EICH16EN	Error Injection Channel 16 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 16 1b - Error injection is enabled on Error Injection Channel 16
14 EICH17EN	Error Injection Channel 17 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 17 1b - Error injection is enabled on Error Injection Channel 17
13 EICH18EN	Error Injection Channel 18 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 18 1b - Error injection is enabled on Error Injection Channel 18
12 EICH19EN	Error Injection Channel 19 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 19 1b - Error injection is enabled on Error Injection Channel 19
11 EICH20EN	Error Injection Channel 20 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 20 1b - Error injection is enabled on Error Injection Channel 20

Table continues on the next page...

Table continued from the previous page...

Field	Function
10 EICH21EN	Error Injection Channel 21 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 21 1b - Error injection is enabled on Error Injection Channel 21
9 EICH22EN	Error Injection Channel 22 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 22 1b - Error injection is enabled on Error Injection Channel 22
8 EICH23EN	Error Injection Channel 23 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 23 1b - Error injection is enabled on Error Injection Channel 23
7 EICH24EN	Error Injection Channel 24 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 24 1b - Error injection is enabled on Error Injection Channel 24
6 EICH25EN	Error Injection Channel 25 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 25 1b - Error injection is enabled on Error Injection Channel 25
5 EICH26EN	Error Injection Channel 26 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 26 1b - Error injection is enabled on Error Injection Channel 26
4 EICH27EN	Error Injection Channel 27 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 27 1b - Error injection is enabled on Error Injection Channel 27
3 EICH28EN	Error Injection Channel 28 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 28 1b - Error injection is enabled on Error Injection Channel 28
2 EICH29EN	Error Injection Channel 29 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 29 1b - Error injection is enabled on Error Injection Channel 29
1 EICH30EN	Error Injection Channel 30 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 30 1b - Error injection is enabled on Error Injection Channel 30
0 EICH31EN	Error Injection Channel 31 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 31 1b - Error injection is enabled on Error Injection Channel 31

73.11.4 Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)

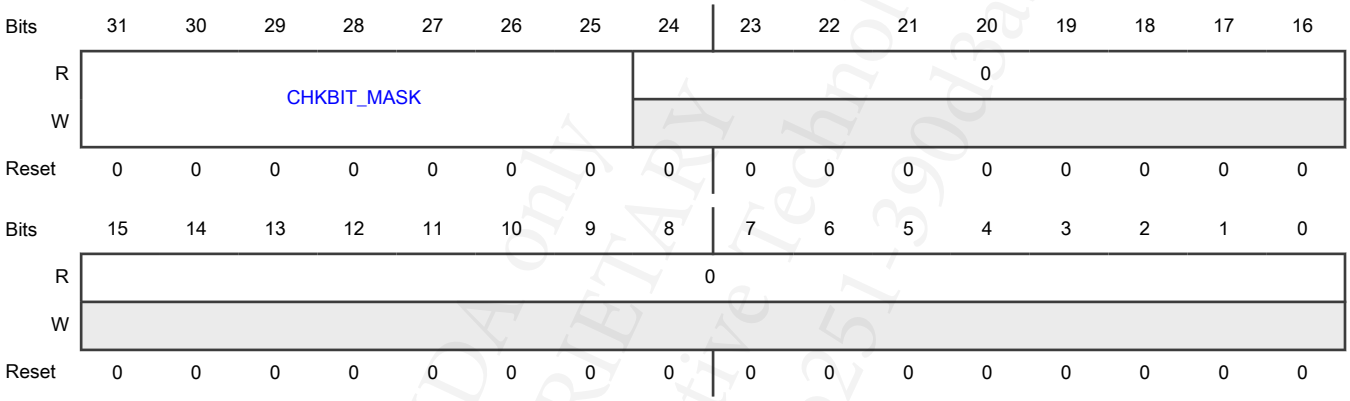
Offset

Register	Offset
EICHD0_WORD0	100h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.11.5 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD1_WORD1)

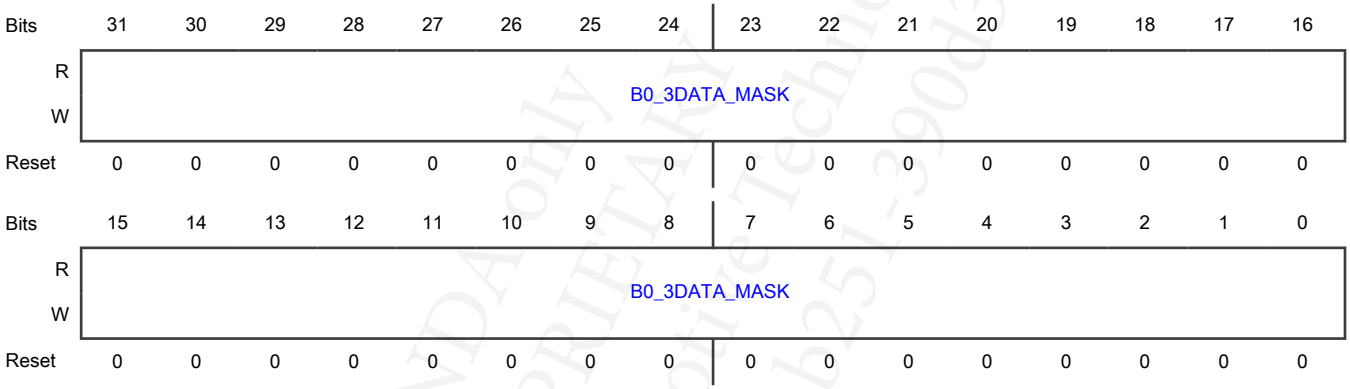
Offset

Register	Offset
EICHD0_WORD1	104h
EICHD1_WORD1	144h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-8187-48a8-b251-590d3ab64c1

73.11.6 Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)

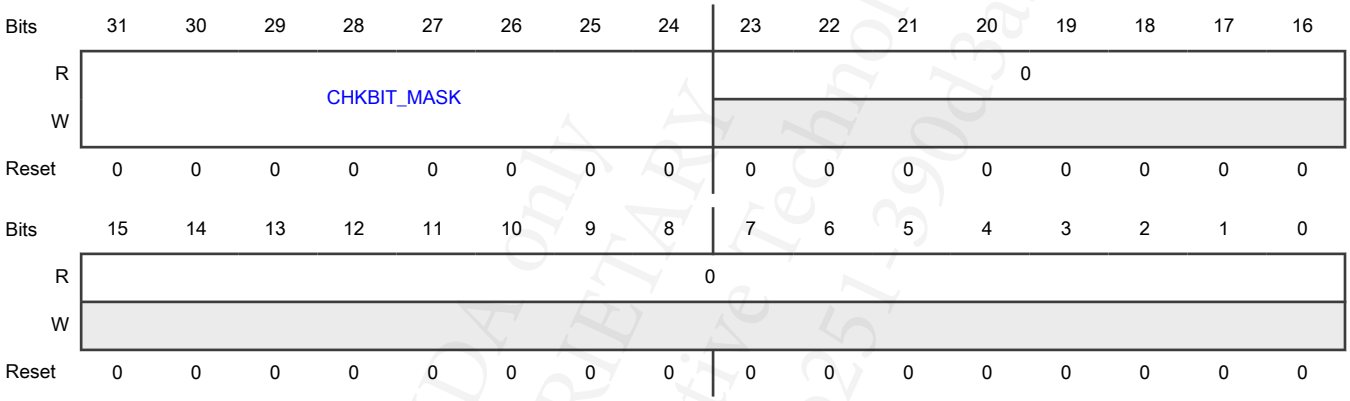
Offset

Register	Offset
EICHD1_WORD0	140h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.11.7 Error Injection Channel Descriptor n, Word2 (EICHD1_WORD2 - EICHD28_WORD2)

Offset

Register	Offset
EICHD1_WORD2	148h
EICHD2_WORD2	188h
EICHD3_WORD2	1C8h
EICHD8_WORD2	308h
EICHD13_WORD2	448h
EICHD18_WORD2	588h
EICHD23_WORD2	6C8h
EICHD28_WORD2	808h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Bytes 4-7 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details.

Table continues on the next page...

Field	Function
	0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

73.11.8 Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0)

Offset

Register	Offset
EICHD2_WORD0	180h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	CHKBIT_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

73.11.9 Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)

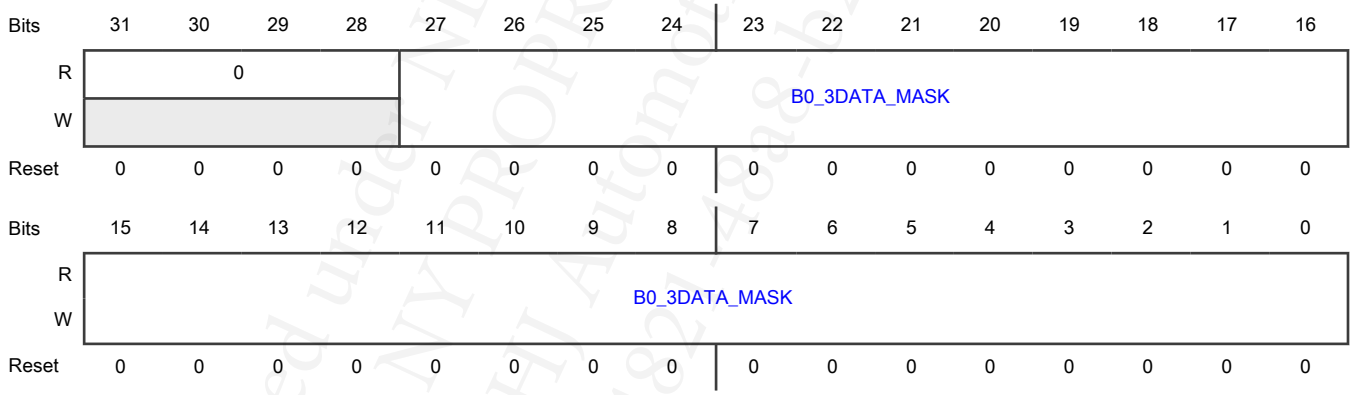
Offset

Register	Offset
EICHD2_WORD1	184h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-28 —	Reserved
27-0	Data Mask Bytes 0-3

Table continues on the next page...

Table continued from the previous page...

Field	Function
B0_3DATA_MASK	This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.11.10 Error Injection Channel Descriptor 2, Word3 (EICHD2_WORD3)

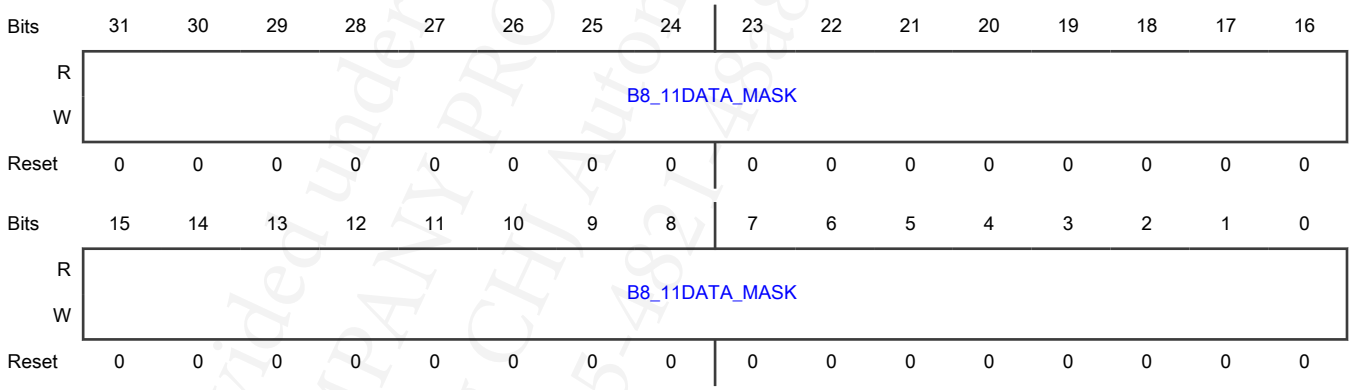
Offset

Register	Offset
EICHD2_WORD3	18Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to bytes 8–11 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0	Data Mask Bytes 8-11

Table continues on the next page...

Field	Function
B8_11DATA_MASK	This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 8-11 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 8-11 on the read data bus is inverted.

73.11.11 Error Injection Channel Descriptor 2, Word4 (EICHD2_WORD4)

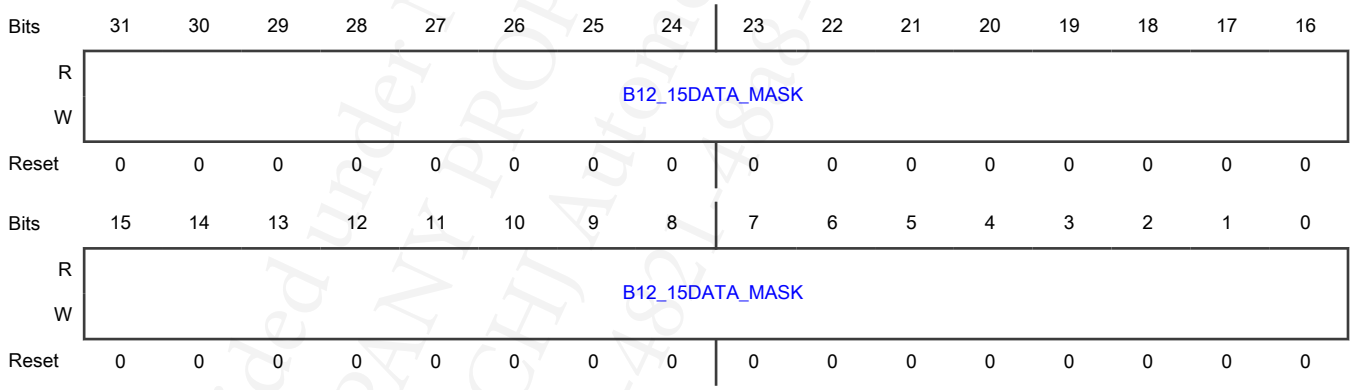
Offset

Register	Offset
EICHD2_WORD4	190h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to bytes 12–15 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Bytes 12-15

Table continues on the next page...

Field	Function
	This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 12-15 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 12-15 on the read data bus is inverted.

73.11.12 Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)

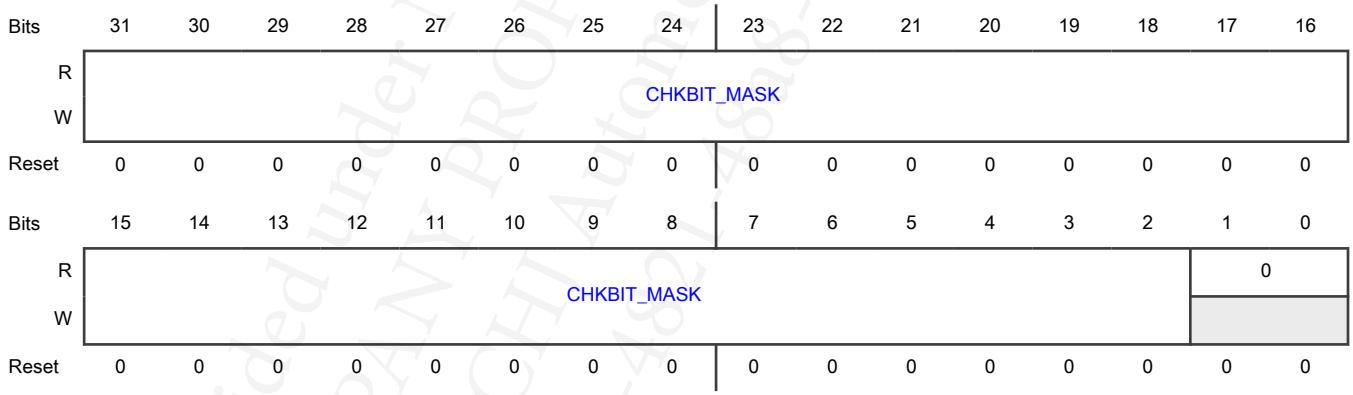
Offset

Register	Offset
EICHD3_WORD0	1C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-2 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[29:0] (30 bits wide), CHKBIT_MASK[29] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
1-0 —	Reserved

73.11.13 Error Injection Channel Descriptor n, Word1 (EICHD3_WORD1 - EICHD4_WORD1)

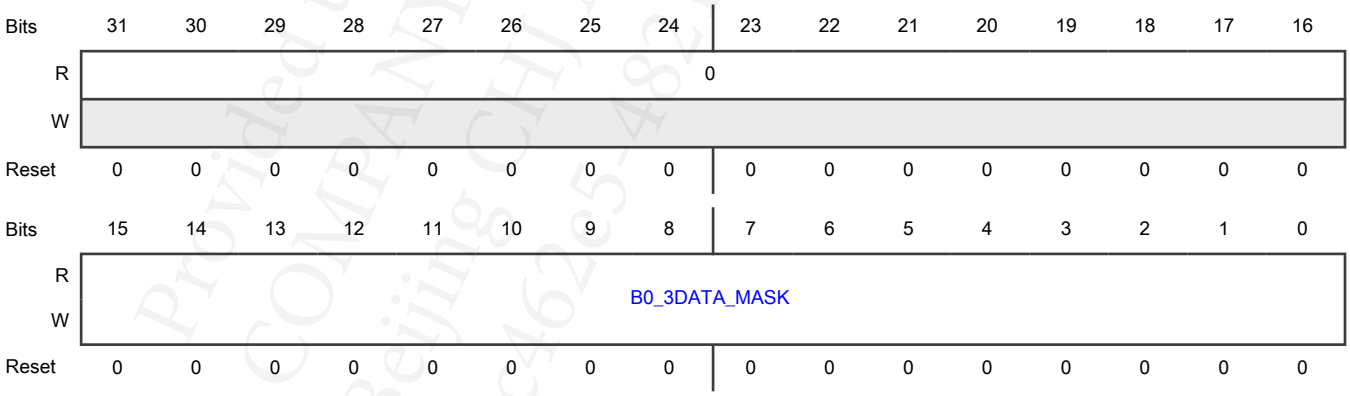
Offset

Register	Offset
EICHD3_WORD1	1C4h
EICHD4_WORD1	204h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.11.14 Error Injection Channel Descriptor 4, Word0 (EICHD4_WORD0)

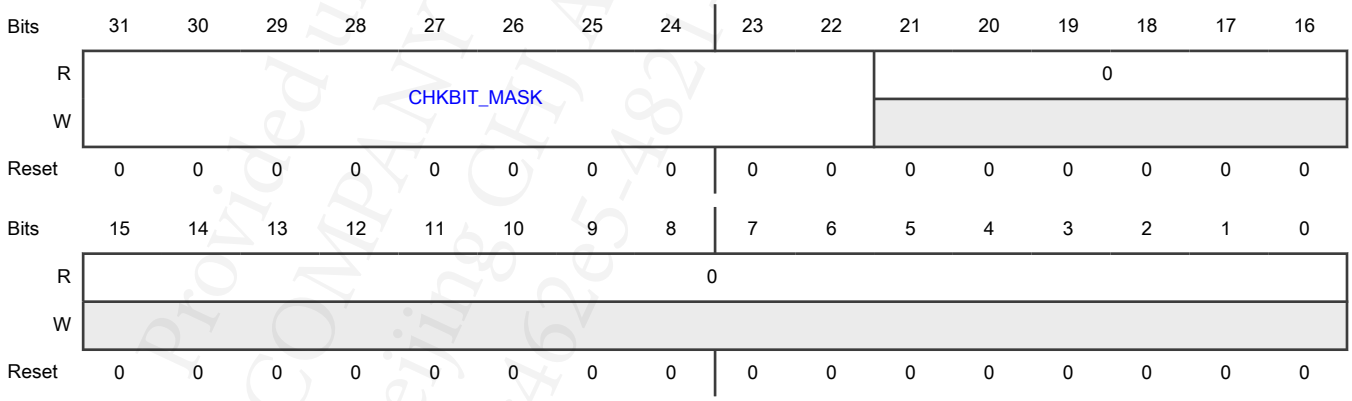
Offset

Register	Offset
EICHD4_WORD0	200h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482a-b251-3903-8b6240

Fields

Field	Function
31-22 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[9:0] (10 bits wide), CHKBIT_MASK[9] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
21-0 —	Reserved

73.11.15 Error Injection Channel Descriptor n, Word0 (EICHD5_WORD0 - EICHD6_WORD0)

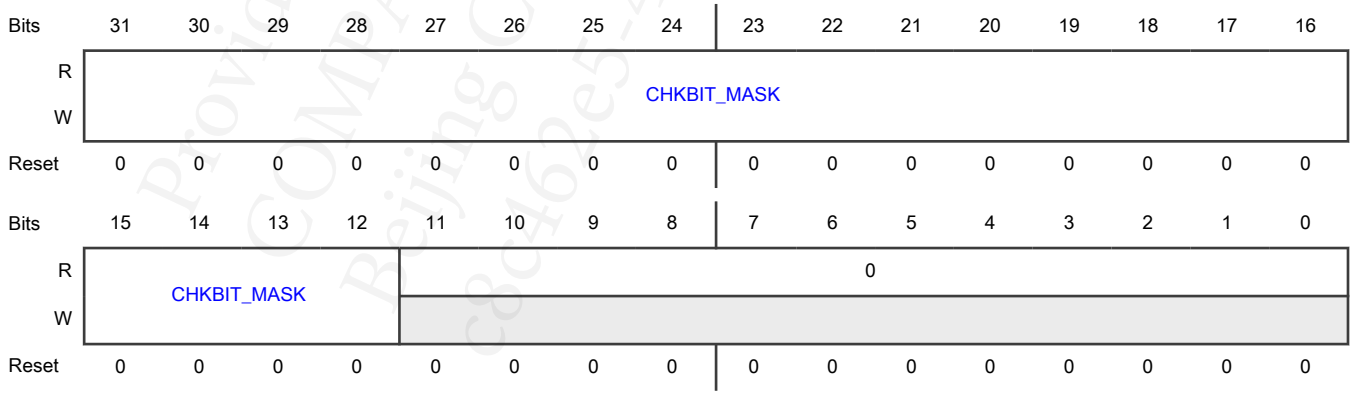
Offset

Register	Offset
EICHD5_WORD0	240h
EICHD6_WORD0	280h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[19:0] (20 bits wide), CHKBIT_MASK[19] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
11-0 —	Reserved

73.11.16 Error Injection Channel Descriptor n, Word1 (EICHD5_WORD1 - EICHD7_WORD1)

Offset

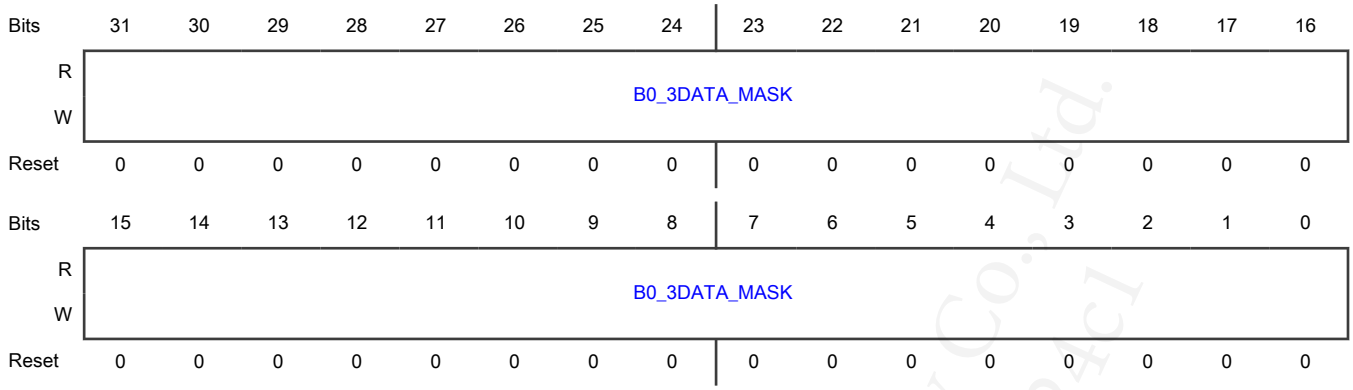
Register	Offset
EICHD5_WORD1	244h
EICHD6_WORD1	284h
EICHD7_WORD1	2C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHTC Automotive Technology Co. Ltd.
c8c462e5-487f-48a8-b251-590d3ab624c1

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.11.17 Error Injection Channel Descriptor 7, Word0 (EICHD7_WORD0)

Offset

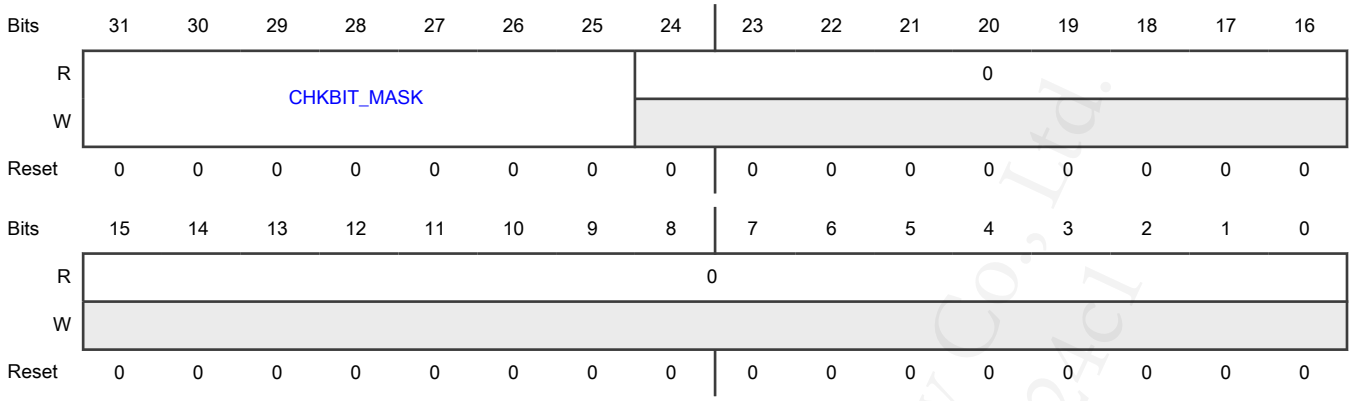
Register	Offset
EICHD7_WORD0	2C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing HHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a6-b251-39d3ab624c1

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.11.18 Error Injection Channel Descriptor 8, Word0 (EICHD8_WORD0)

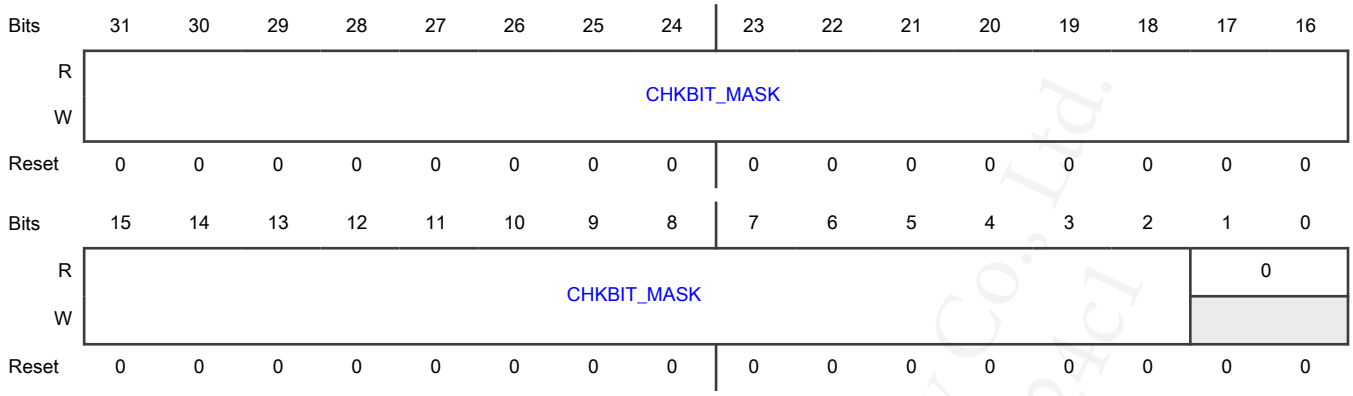
Offset

Register	Offset
EICHD8_WORD0	300h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-2 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[29:0] (30 bits wide), CHKBIT_MASK[29] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
1-0 —	Reserved

73.11.19 Error Injection Channel Descriptor n, Word1 (EICHD8_WORD1 - EICHD9_WORD1)

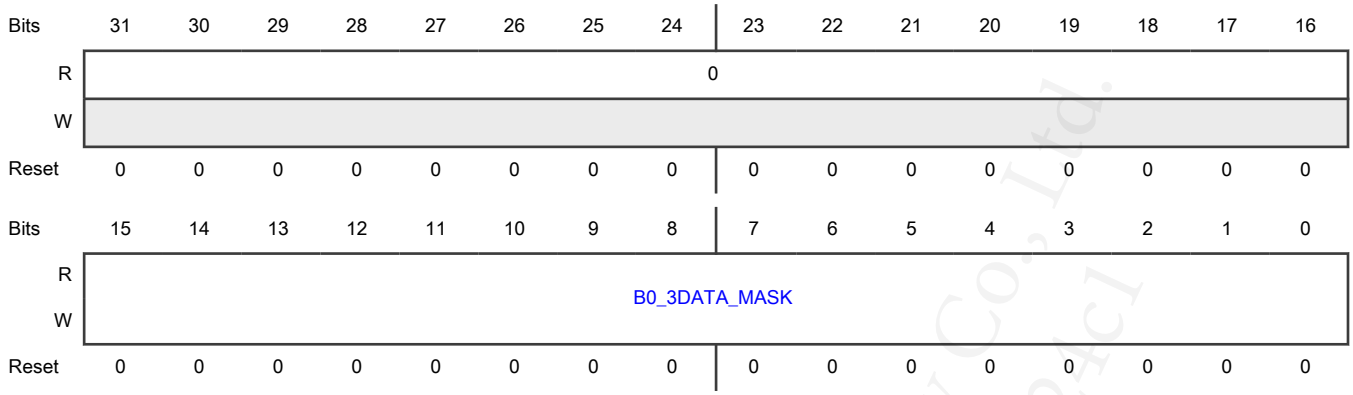
Offset

Register	Offset
EICHD8_WORD1	304h
EICHD9_WORD1	344h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.11.20 Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0)

Offset

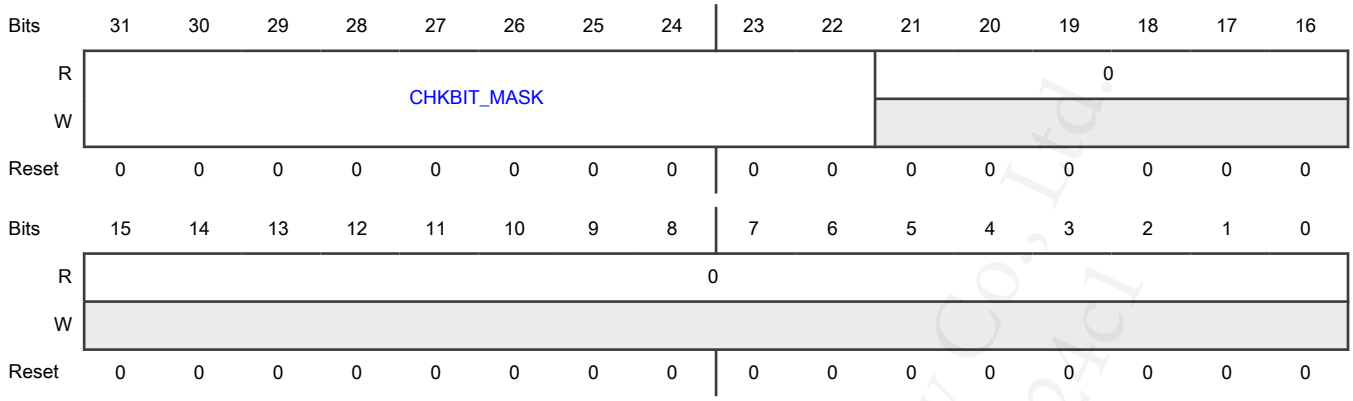
Register	Offset
EICHD9_WORD0	340h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b513-90d3ab624c1

Diagram



Fields

Field	Function
31-22 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[9:0] (10 bits wide), CHKBIT_MASK[9] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
21-0 —	Reserved

73.11.21 Error Injection Channel Descriptor n, Word0 (EICHD10_WORD0 - EICHD11_WORD0)

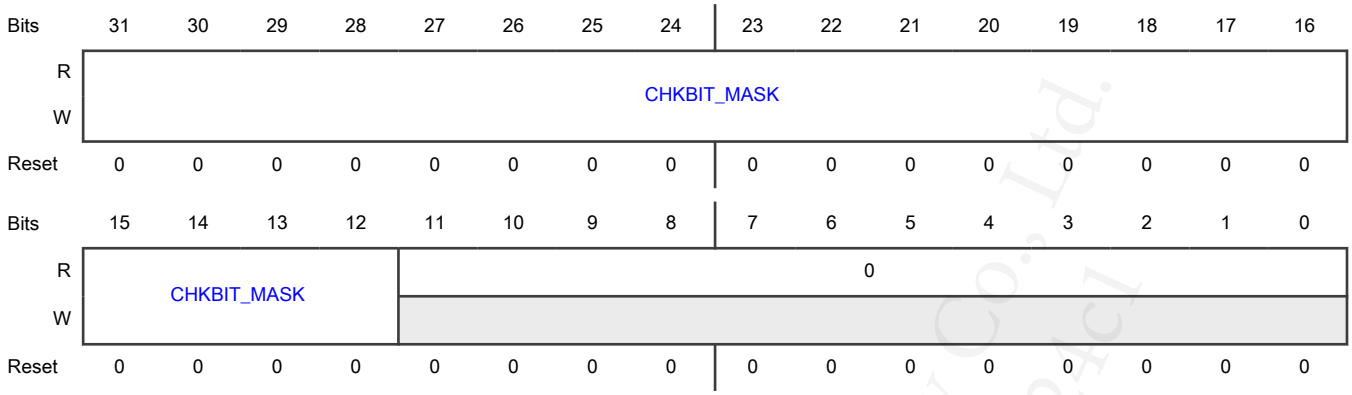
Offset

Register	Offset
EICHD10_WORD0	380h
EICHD11_WORD0	3C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[19:0] (20 bits wide), CHKBIT_MASK[19] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
11-0 —	Reserved

73.11.22 Error Injection Channel Descriptor n, Word1 (EICHD10_WORD1 - EICHD12_WORD1)

Offset

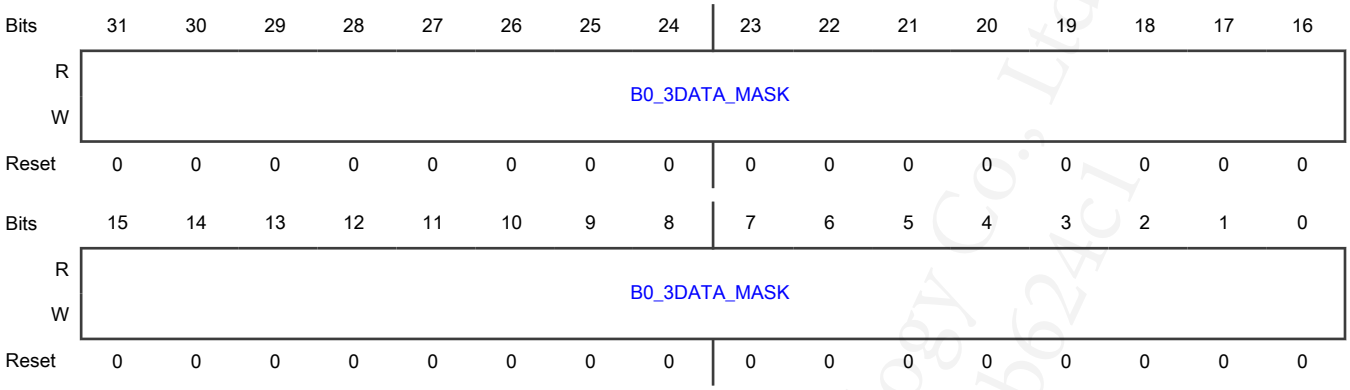
Register	Offset
EICHD10_WORD1	384h
EICHD11_WORD1	3C4h
EICHD12_WORD1	404h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target

RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.11.23 Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0)

Offset

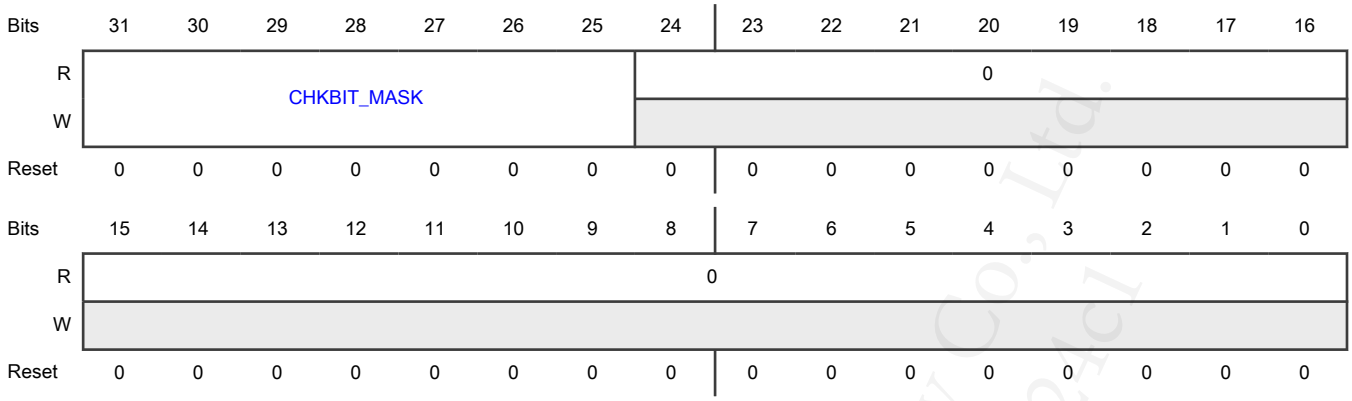
Register	Offset
EICHD12_WORD0	400h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Beijing CHJ Automotive Technology Co., Ltd.
COMPANY PROPRIETARY
Provided under NDA only
c8c4655-4821-4a3b-b251-590d3ab624c1

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.11.24 Error Injection Channel Descriptor 13, Word0 (EICHD13_WORD0)

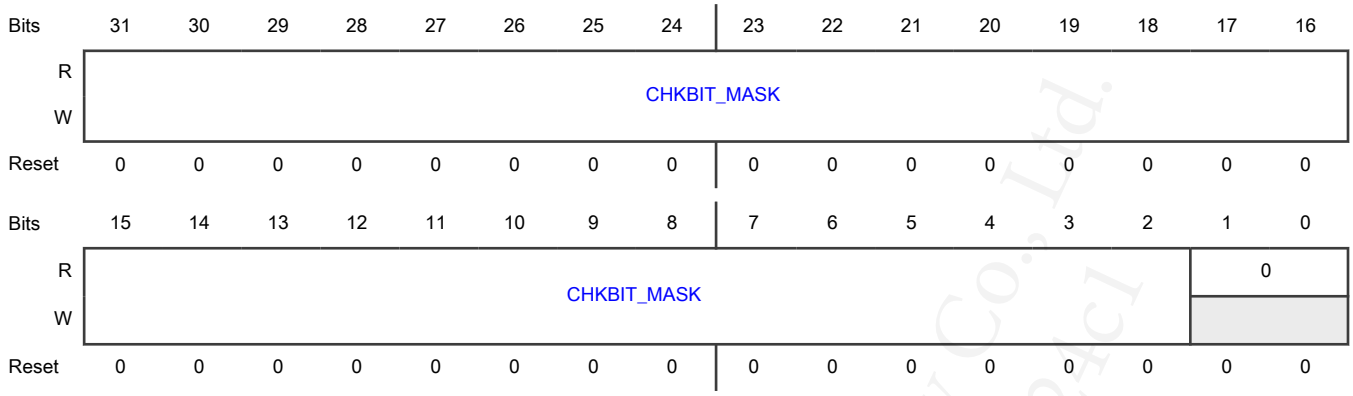
Offset

Register	Offset
EICHD13_WORD0	440h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-2 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[29:0] (30 bits wide), CHKBIT_MASK[29] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
1-0 —	Reserved

73.11.25 Error Injection Channel Descriptor n, Word1 (EICHD13_WORD1 - EICHD14_WORD1)

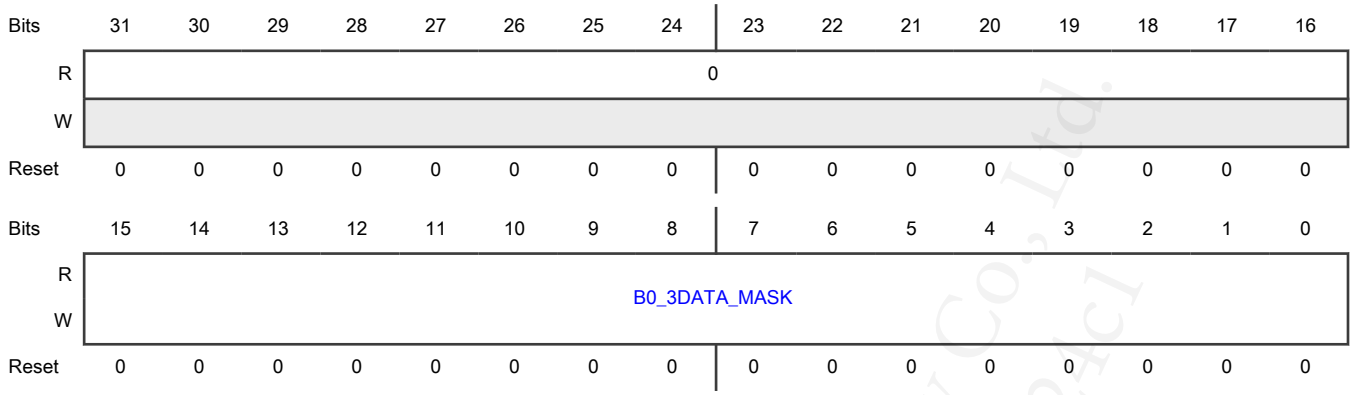
Offset

Register	Offset
EICHD13_WORD1	444h
EICHD14_WORD1	484h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.11.26 Error Injection Channel Descriptor 14, Word0 (EICHD14_WORD0)

Offset

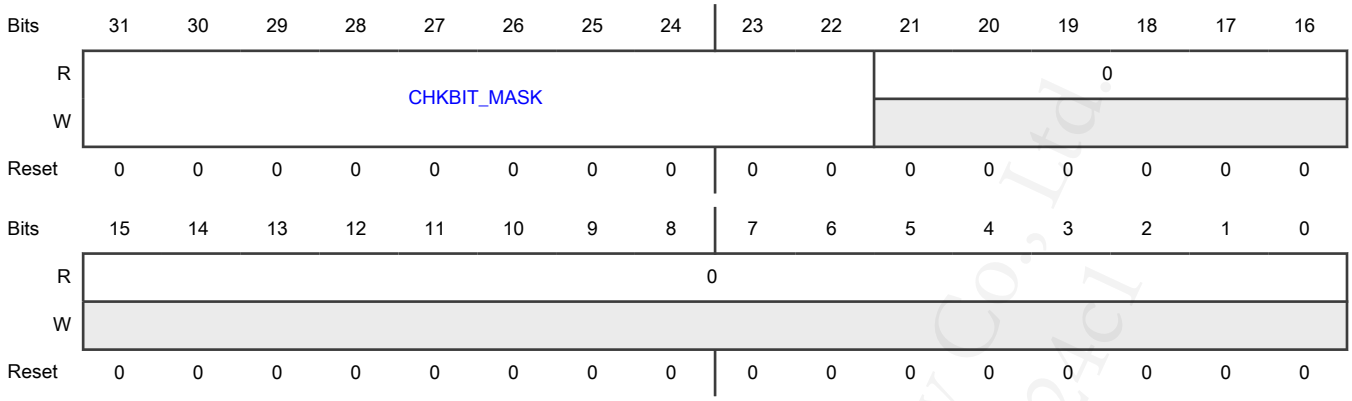
Register	Offset
EICHD14_WORD0	480h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b513-90d3ab624c1

Diagram



Fields

Field	Function
31-22 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[9:0] (10 bits wide), CHKBIT_MASK[9] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
21-0 —	Reserved

73.11.27 Error Injection Channel Descriptor n, Word0 (EICHD15_WORD0 - EICHD16_WORD0)

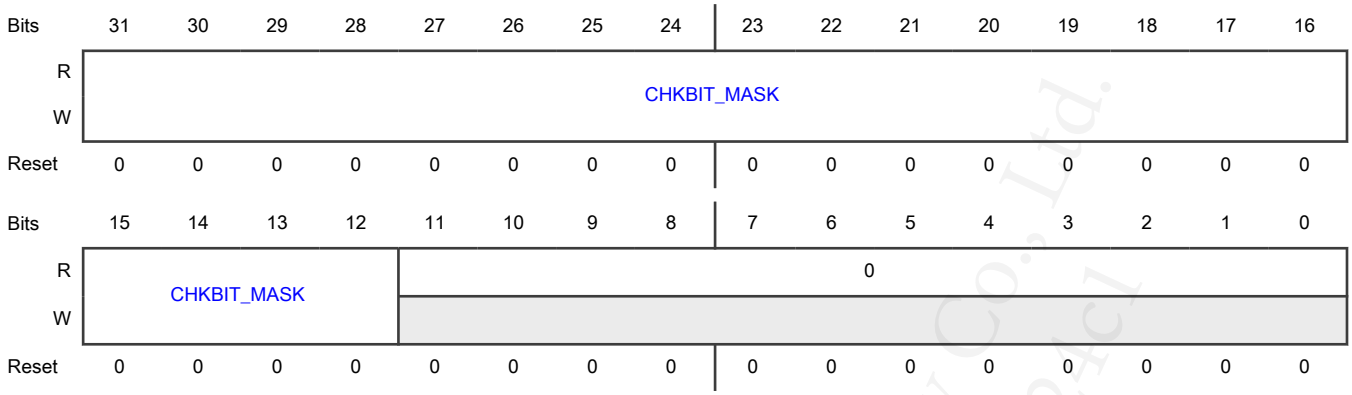
Offset

Register	Offset
EICHD15_WORD0	4C0h
EICHD16_WORD0	500h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[19:0] (20 bits wide), CHKBIT_MASK[19] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
11-0 —	Reserved

73.11.28 Error Injection Channel Descriptor n, Word1 (EICHD15_WORD1 - EICHD17_WORD1)

Offset

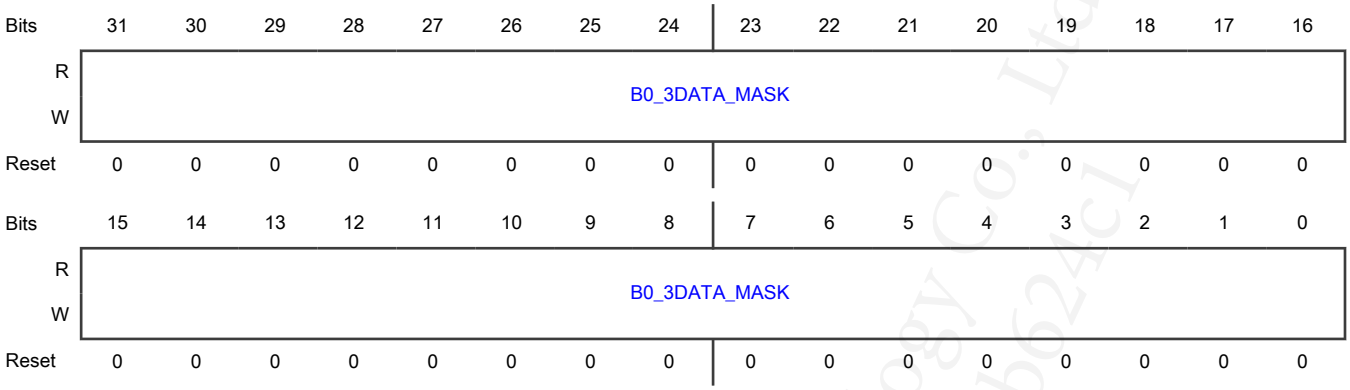
Register	Offset
EICHD15_WORD1	4C4h
EICHD16_WORD1	504h
EICHD17_WORD1	544h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target

RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.11.29 Error Injection Channel Descriptor 17, Word0 (EICHD17_WORD0)

Offset

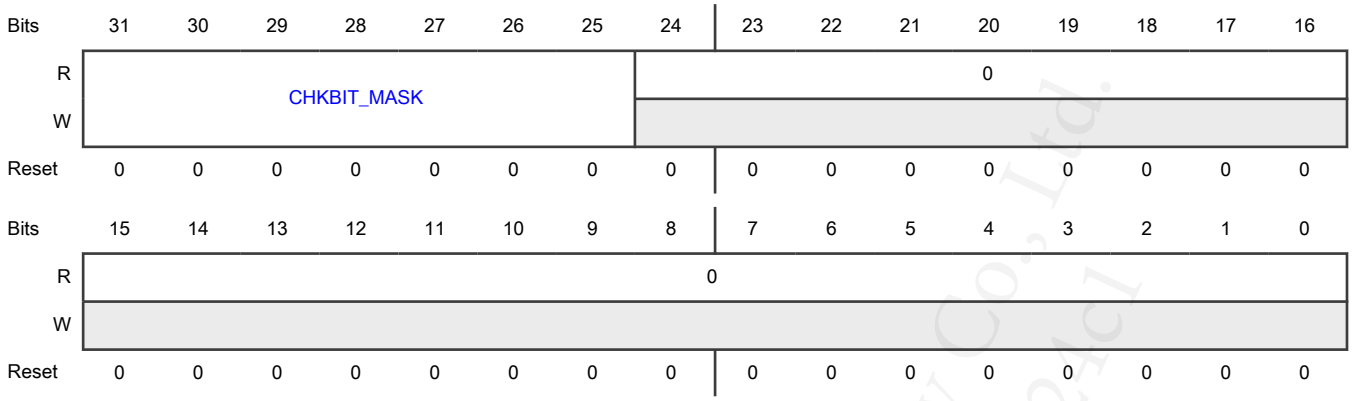
Register	Offset
EICHD17_WORD0	540h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NXP only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4655-4821-8a3b-2513-90d3ab624c1

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.11.30 Error Injection Channel Descriptor 18, Word0 (EICHD18_WORD0)

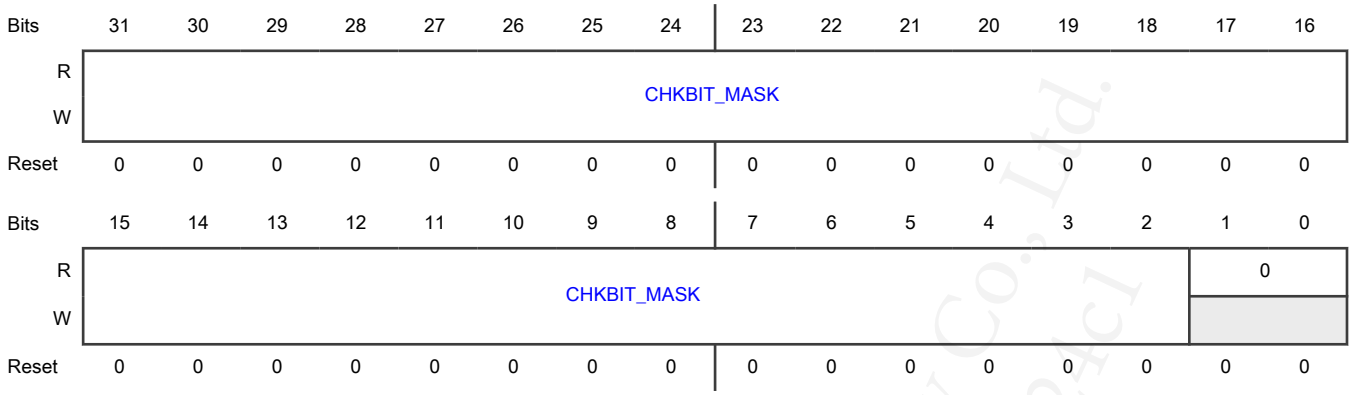
Offset

Register	Offset
EICHD18_WORD0	580h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-2 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[29:0] (30 bits wide), CHKBIT_MASK[29] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
1-0 —	Reserved

73.11.31 Error Injection Channel Descriptor n, Word1 (EICHD18_WORD1 - EICHD19_WORD1)

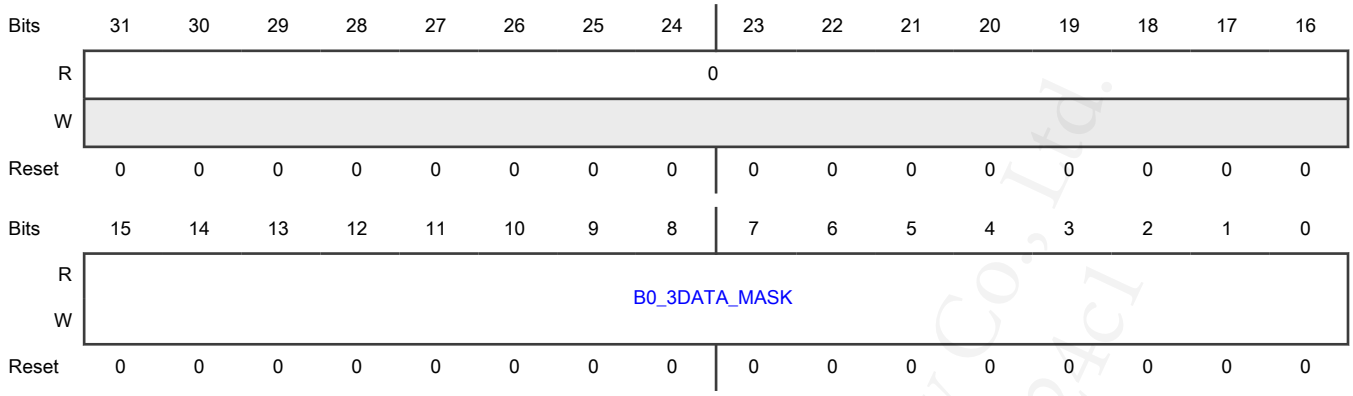
Offset

Register	Offset
EICHD18_WORD1	584h
EICHD19_WORD1	5C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.11.32 Error Injection Channel Descriptor 19, Word0 (EICHD19_WORD0)

Offset

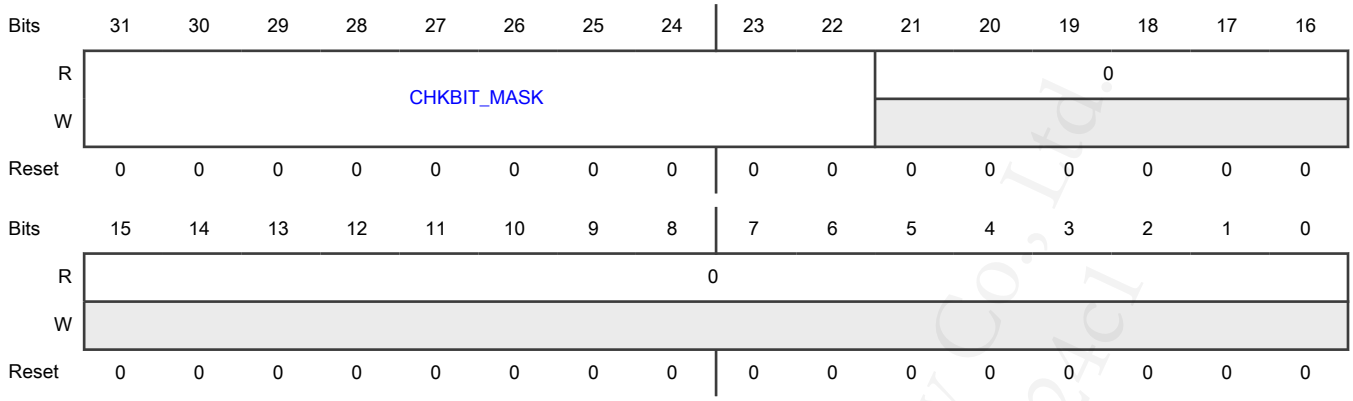
Register	Offset
EICHD19_WORD0	5C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b513-90d3ab624c1

Diagram



Fields

Field	Function
31-22 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[9:0] (10 bits wide), CHKBIT_MASK[9] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
21-0 —	Reserved

73.11.33 Error Injection Channel Descriptor n, Word0 (EICHD20_WORD0 - EICHD21_WORD0)

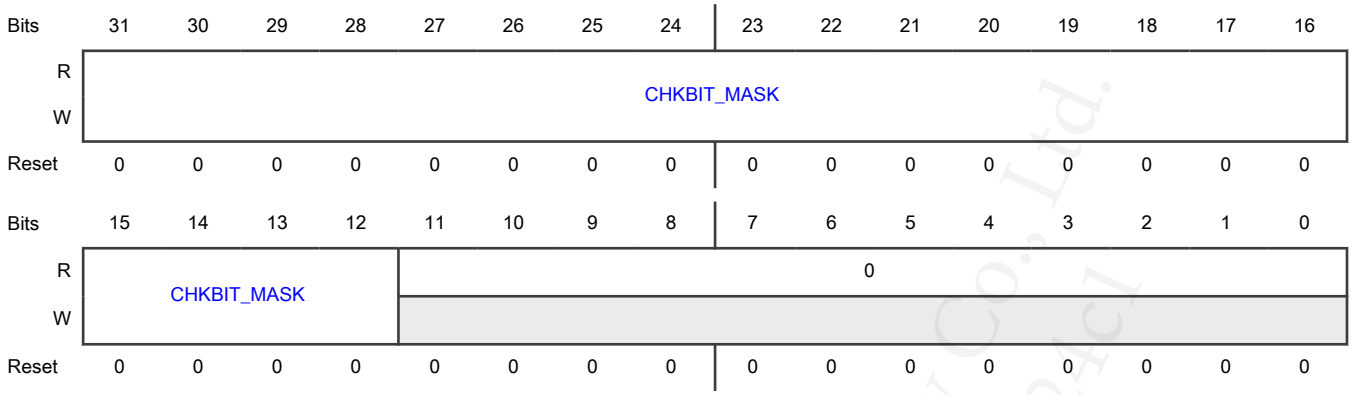
Offset

Register	Offset
EICHD20_WORD0	600h
EICHD21_WORD0	640h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[19:0] (20 bits wide), CHKBIT_MASK[19] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
11-0 —	Reserved

73.11.34 Error Injection Channel Descriptor n, Word1 (EICHD20_WORD1 - EICHD22_WORD1)

Offset

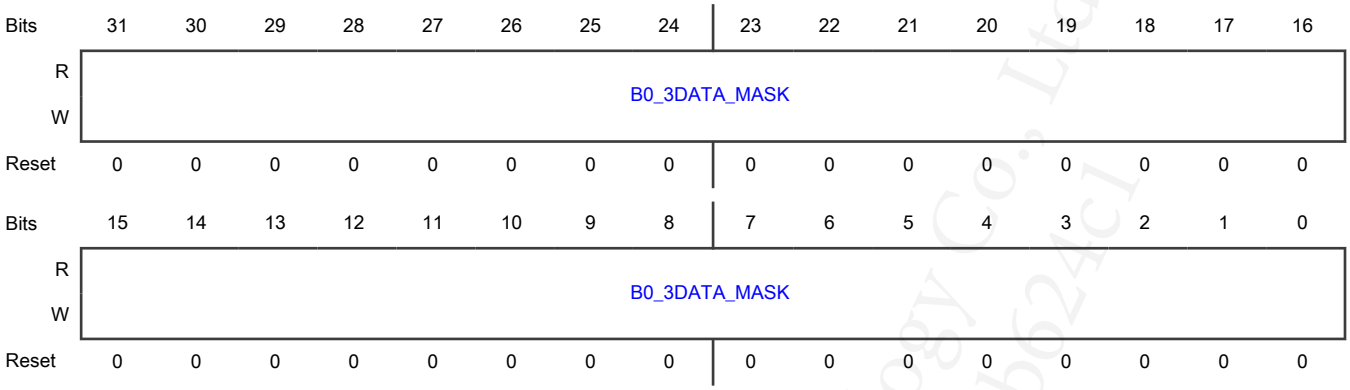
Register	Offset
EICHD20_WORD1	604h
EICHD21_WORD1	644h
EICHD22_WORD1	684h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target

RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.11.35 Error Injection Channel Descriptor 22, Word0 (EICHD22_WORD0)

Offset

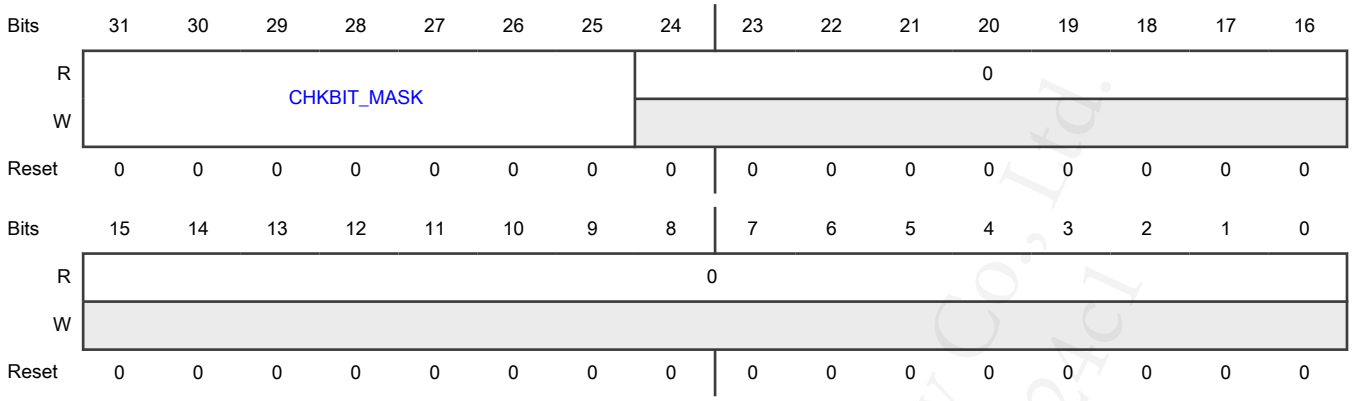
Register	Offset
EICHD22_WORD0	680h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NXP Only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4655-4821-4a3b-b251-90d3ab624c1

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.11.36 Error Injection Channel Descriptor 23, Word0 (EICHD23_WORD0)

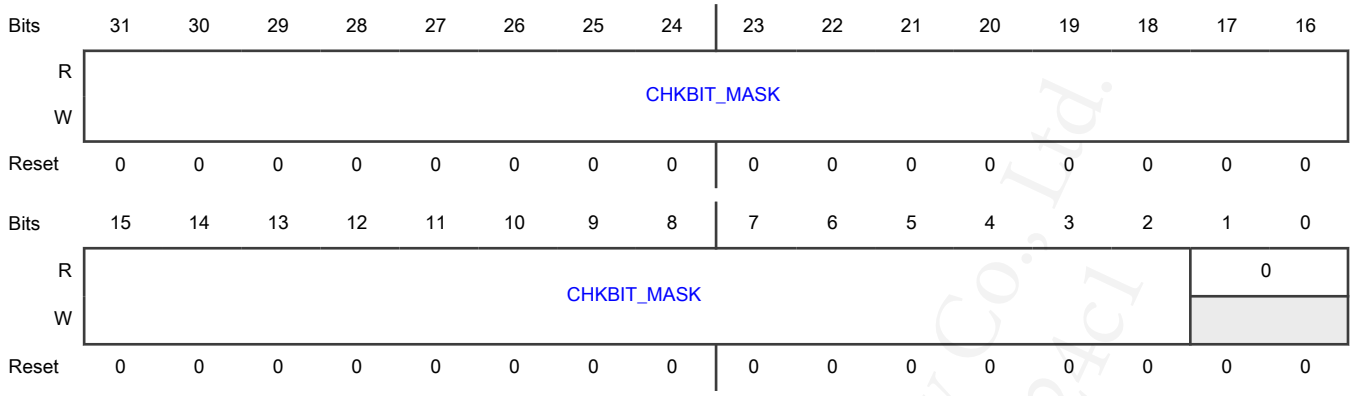
Offset

Register	Offset
EICHD23_WORD0	6C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-2 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[29:0] (30 bits wide), CHKBIT_MASK[29] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
1-0 —	Reserved

73.11.37 Error Injection Channel Descriptor n, Word1 (EICHD23_WORD1 - EICHD24_WORD1)

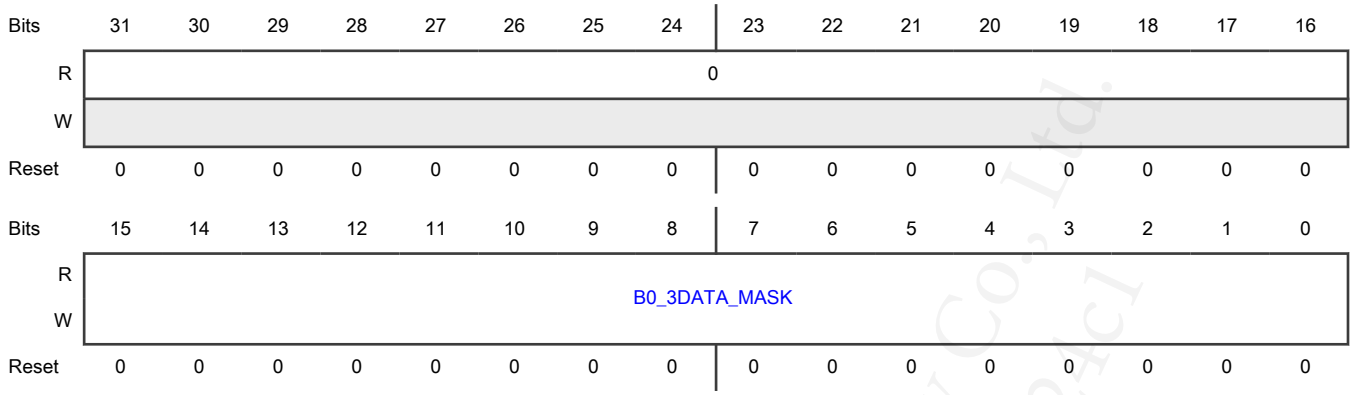
Offset

Register	Offset
EICHD23_WORD1	6C4h
EICHD24_WORD1	704h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.11.38 Error Injection Channel Descriptor 24, Word0 (EICHD24_WORD0)

Offset

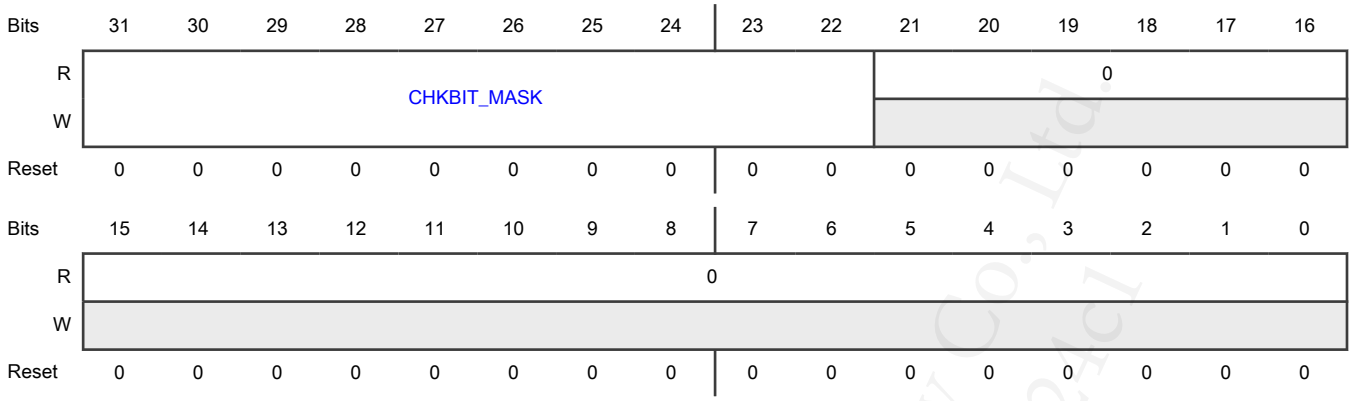
Register	Offset
EICHD24_WORD0	700h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b513-90d3ab624c1

Diagram



Fields

Field	Function
31-22 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[9:0] (10 bits wide), CHKBIT_MASK[9] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
21-0 —	Reserved

73.11.39 Error Injection Channel Descriptor n, Word0 (EICHD25_WORD0 - EICHD26_WORD0)

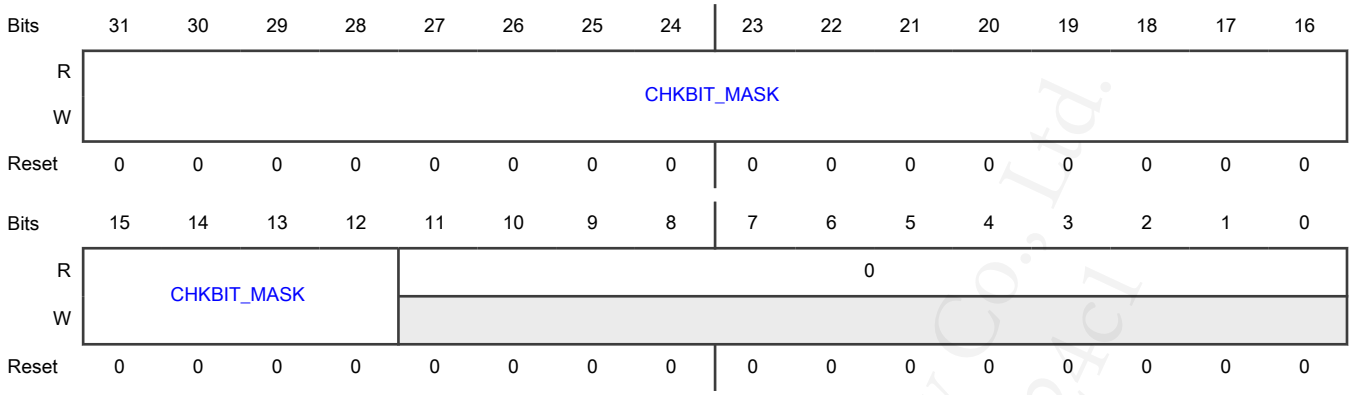
Offset

Register	Offset
EICHD25_WORD0	740h
EICHD26_WORD0	780h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[19:0] (20 bits wide), CHKBIT_MASK[19] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
11-0 —	Reserved

73.11.40 Error Injection Channel Descriptor n, Word1 (EICHD25_WORD1 - EICHD27_WORD1)

Offset

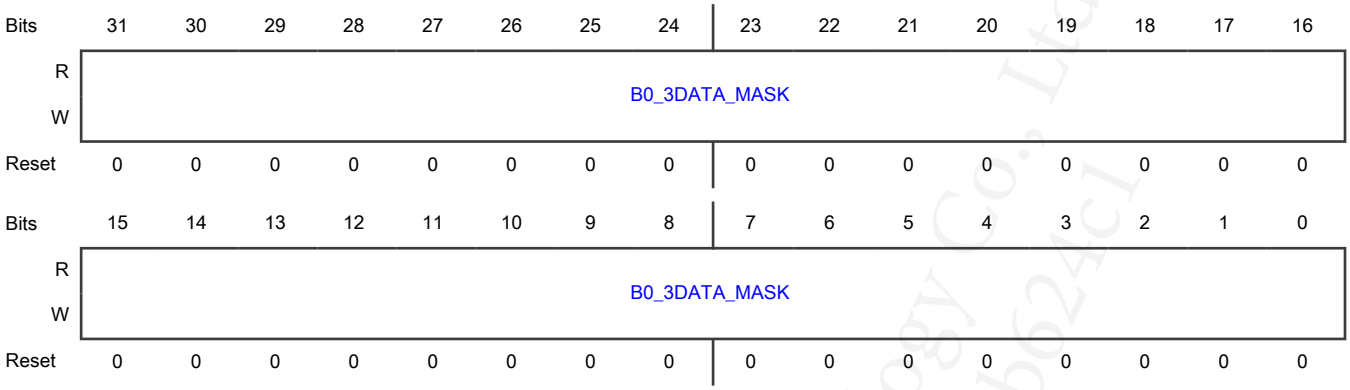
Register	Offset
EICHD25_WORD1	744h
EICHD26_WORD1	784h
EICHD27_WORD1	7C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target

RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.11.41 Error Injection Channel Descriptor 27, Word0 (EICHD27_WORD0)

Offset

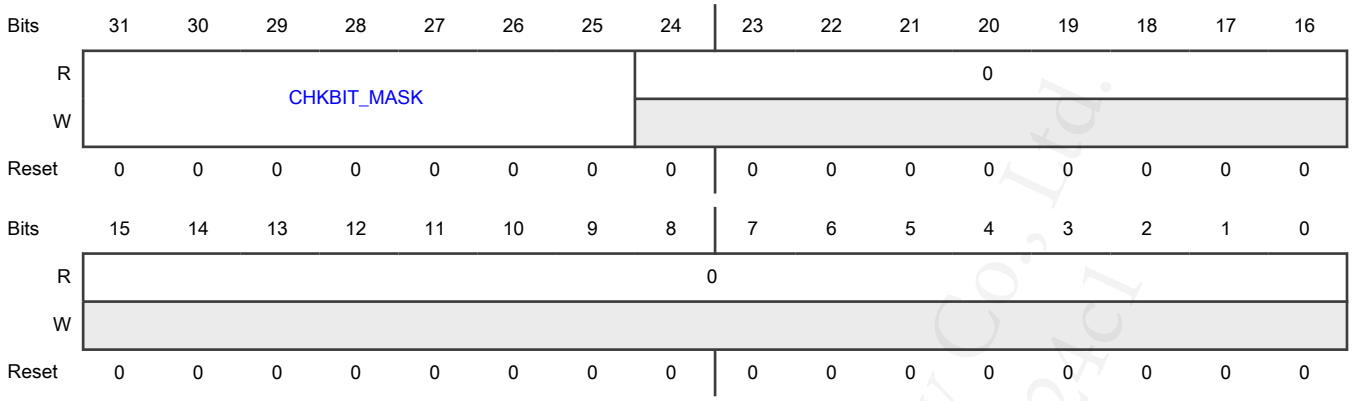
Register	Offset
EICHD27_WORD0	7C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NXP Only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4655-4821-4a3b-b251-590d3ab624c1

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.11.42 Error Injection Channel Descriptor 28, Word0 (EICHD28_WORD0)

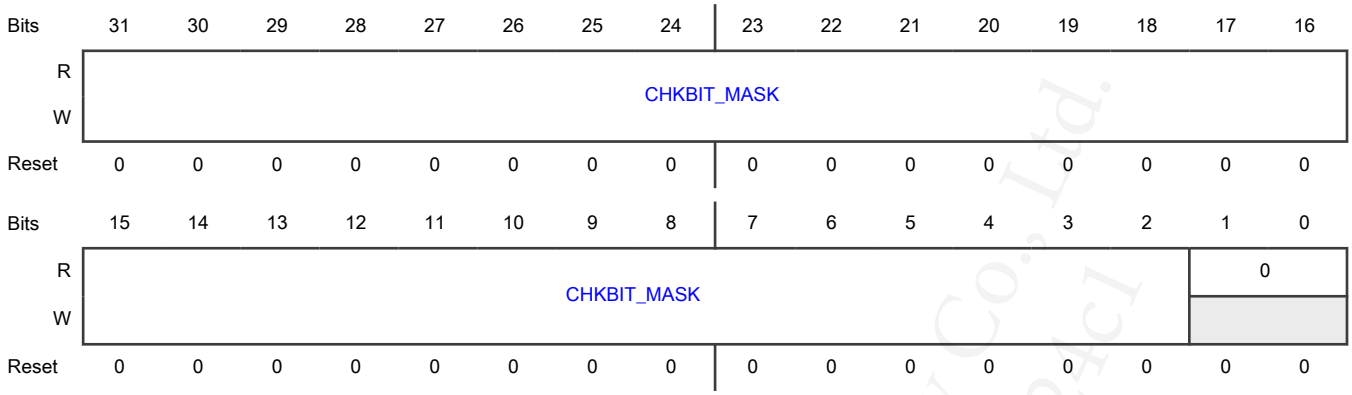
Offset

Register	Offset
EICHD28_WORD0	800h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-2 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[29:0] (30 bits wide), CHKBIT_MASK[29] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
1-0 —	Reserved

73.11.43 Error Injection Channel Descriptor n, Word1 (EICHD28_WORD1 - EICHD29_WORD1)

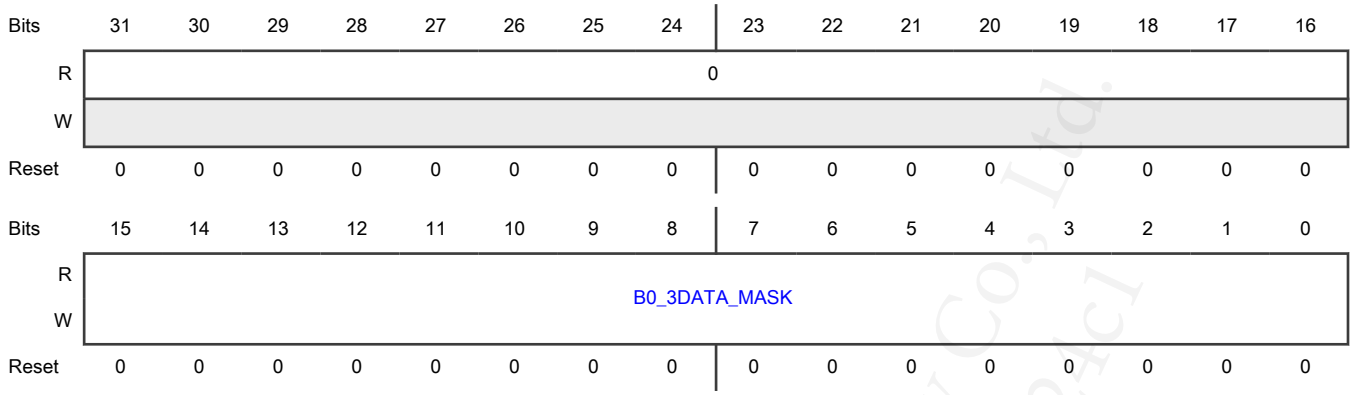
Offset

Register	Offset
EICHD28_WORD1	804h
EICHD29_WORD1	844h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.11.44 Error Injection Channel Descriptor 29, Word0 (EICHD29_WORD0)

Offset

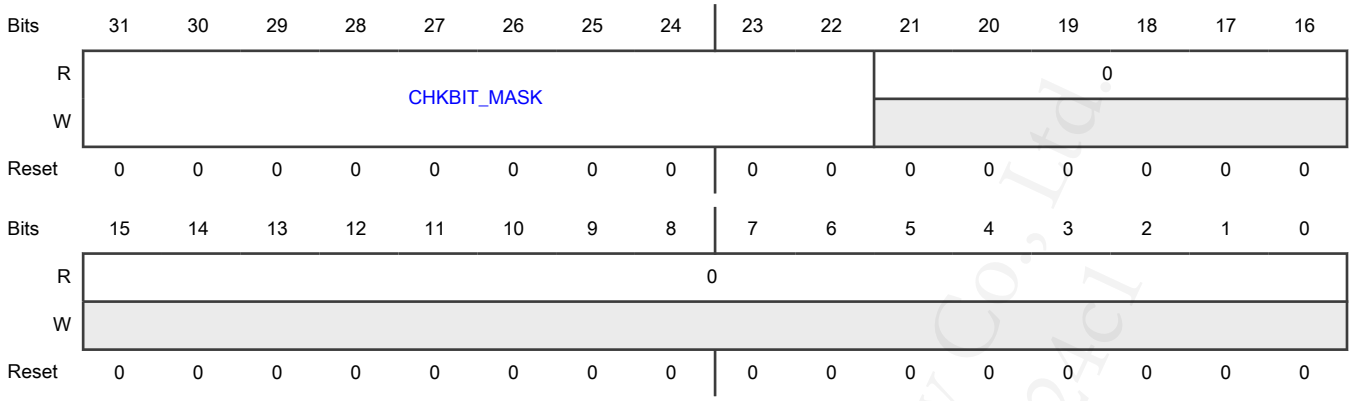
Register	Offset
EICHD29_WORD0	840h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b513-90d3ab624c1

Diagram



Fields

Field	Function
31-22 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[9:0] (10 bits wide), CHKBIT_MASK[9] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
21-0 —	Reserved

73.11.45 Error Injection Channel Descriptor n, Word0 (EICHD30_WORD0 - EICHD31_WORD0)

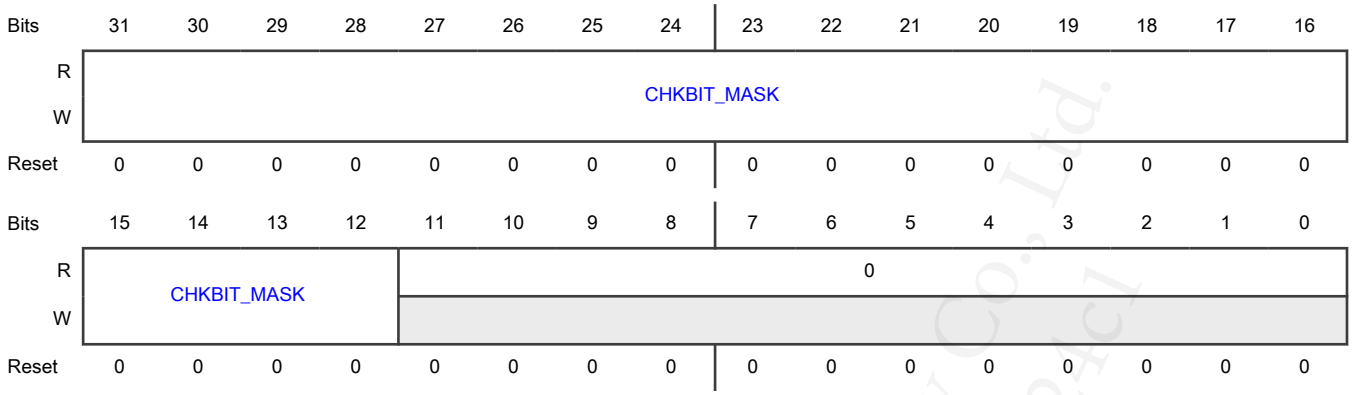
Offset

Register	Offset
EICHD30_WORD0	880h
EICHD31_WORD0	8C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[19:0] (20 bits wide), CHKBIT_MASK[19] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
11-0 —	Reserved

73.11.46 Error Injection Channel Descriptor n, Word1 (EICHD30_WORD1 - EICHD31_WORD1)

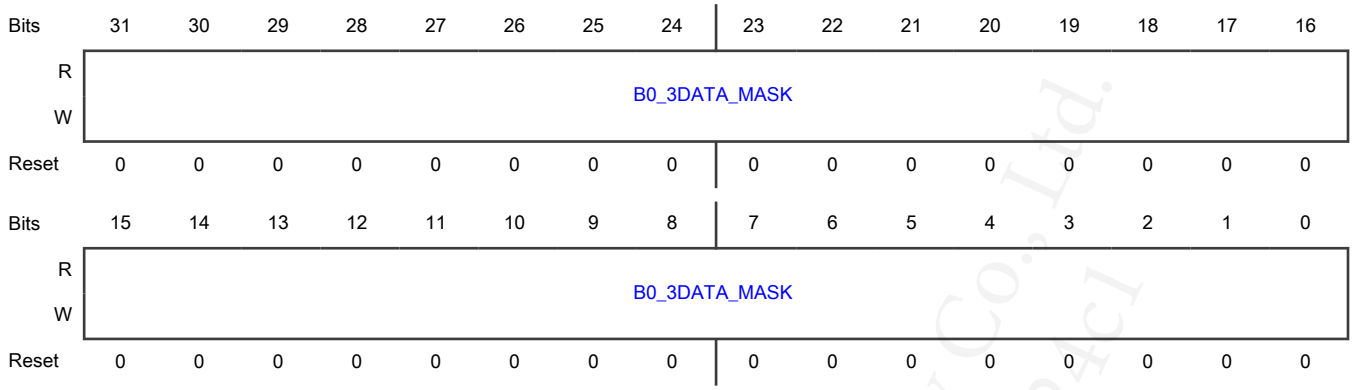
Offset

Register	Offset
EICHD30_WORD1	884h
EICHD31_WORD1	8C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.12 EIM_PFE1 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.

— Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.

— Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 591. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	32	31-0	—	—
1	48	47-32	31-0	—
2	16	15-0	—	—
3	32	31-0	—	—
4	32	31-0	—	—
5	32	31-0	—	—
6	48	47-32	31-0	—
7	16	15-0	—	—
8	32	31-0	—	—
9	32	31-0	—	—
10	32	31-0	—	—
11	32	31-0	—	—
12	64	63-32	31-0	—
13	128	127-96	95-64	63-32
14	100	99-96	95-64	63-32

Table 592. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
13	31-0	—	—	—	—
14	31-0	—	—	—	—

73.12.1 EIM_PFE1 memory map

EIM_PFE1 base address: 4405_5000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
100h	Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
140h	Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h
180h	Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
1C0h	Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
200h	Error Injection Channel Descriptor 4, Word0 (EICHD4_WORD0)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
240h	Error Injection Channel Descriptor 5, Word0 (EICHD5_WORD0)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
280h	Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0)	32	RW	0000_0000h
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h
288h	Error Injection Channel Descriptor 6, Word2 (EICHD6_WORD2)	32	RW	0000_0000h
2C0h	Error Injection Channel Descriptor 7, Word0 (EICHD7_WORD0)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
300h	Error Injection Channel Descriptor 8, Word0 (EICHD8_WORD0)	32	RW	0000_0000h
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
340h	Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
380h	Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0)	32	RW	0000_0000h
384h	Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)	32	RW	0000_0000h
3C0h	Error Injection Channel Descriptor 11, Word0 (EICHD11_WORD0)	32	RW	0000_0000h
3C4h	Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)	32	RW	0000_0000h
400h	Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0)	32	RW	0000_0000h
404h	Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
408h	Error Injection Channel Descriptor 12, Word2 (EICHD12_WORD2)	32	RW	0000_0000h
440h	Error Injection Channel Descriptor 13, Word0 (EICHD13_WORD0)	32	RW	0000_0000h
444h	Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)	32	RW	0000_0000h
448h	Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)	32	RW	0000_0000h
44Ch	Error Injection Channel Descriptor 13, Word3 (EICHD13_WORD3)	32	RW	0000_0000h
450h	Error Injection Channel Descriptor 13, Word4 (EICHD13_WORD4)	32	RW	0000_0000h
480h	Error Injection Channel Descriptor 14, Word0 (EICHD14_WORD0)	32	RW	0000_0000h
484h	Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)	32	RW	0000_0000h
488h	Error Injection Channel Descriptor 14, Word2 (EICHD14_WORD2)	32	RW	0000_0000h
48Ch	Error Injection Channel Descriptor 14, Word3 (EICHD14_WORD3)	32	RW	0000_0000h
490h	Error Injection Channel Descriptor 14, Word4 (EICHD14_WORD4)	32	RW	0000_0000h

73.12.2 Error Injection Module Configuration Register (EIMCR)

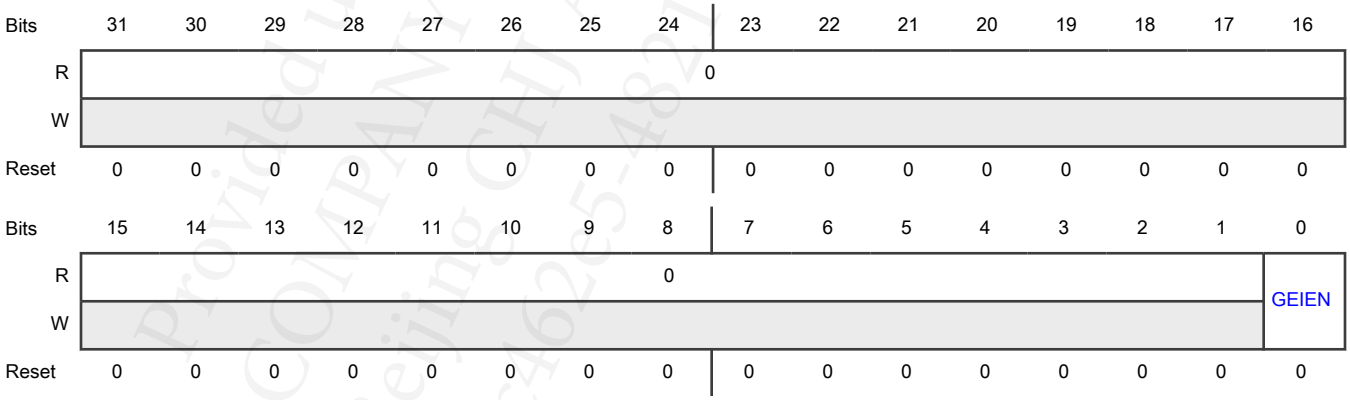
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c16225-4821-4ea8-b251-3900cbb6a1a1

Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

73.12.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	EICH1EN	EICH2EN	EICH3EN	EICH4EN	EICH5EN	EICH6EN	EICH7EN	EICH8EN	EICH9EN	EICH10EN	EICH11EN	EICH12EN	EICH13EN	EICH14EN	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Product under NDA only
COMPANY PROPRIETARY
Beijing CHT Electronics Technology Co., Ltd.
c8c467e5-48a2-4a8-b251-390d3ab624c1

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9
21 EICH10EN	Error Injection Channel 10 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 10 1b - Error injection is enabled on Error Injection Channel 10

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 EICH11EN	Error Injection Channel 11 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 11 1b - Error injection is enabled on Error Injection Channel 11
19 EICH12EN	Error Injection Channel 12 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 12 1b - Error injection is enabled on Error Injection Channel 12
18 EICH13EN	Error Injection Channel 13 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 13 1b - Error injection is enabled on Error Injection Channel 13
17 EICH14EN	Error Injection Channel 14 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 14 1b - Error injection is enabled on Error Injection Channel 14
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

73.12.4 Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)

Offset

Register	Offset
EICHD0_WORD0	100h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CHKBIT_MASK								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c16225-4621-484f-b251-390a3ab624c1

Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.12.5 Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)

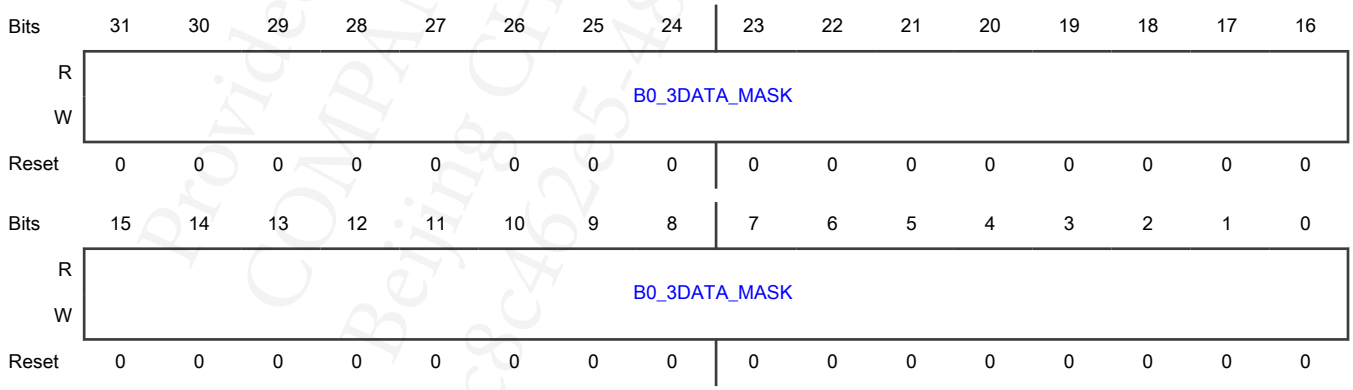
Offset

Register	Offset
EICHD0_WORD1	104h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46255-4821-4a8-b251-590d3ab624c1

Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.12.6 Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)

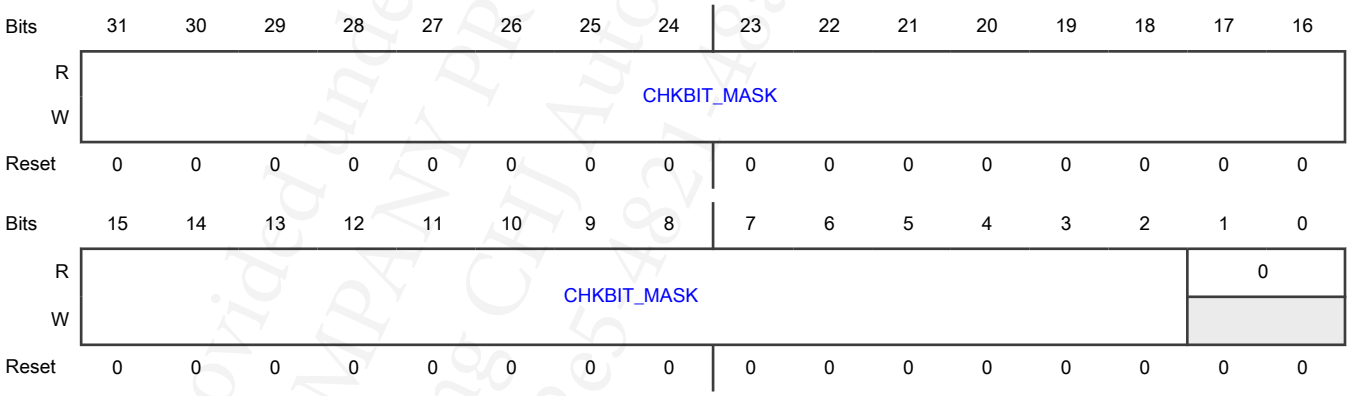
Offset

Register	Offset
EICHD1_WORD0	140h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-2	Checkbit Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
CHKBIT_MASK	This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[29:0] (30 bits wide), CHKBIT_MASK[29] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
1-0 —	Reserved

73.12.7 Error Injection Channel Descriptor n, Word1 (EICHD1_WORD1 - EICHD2_WORD1)

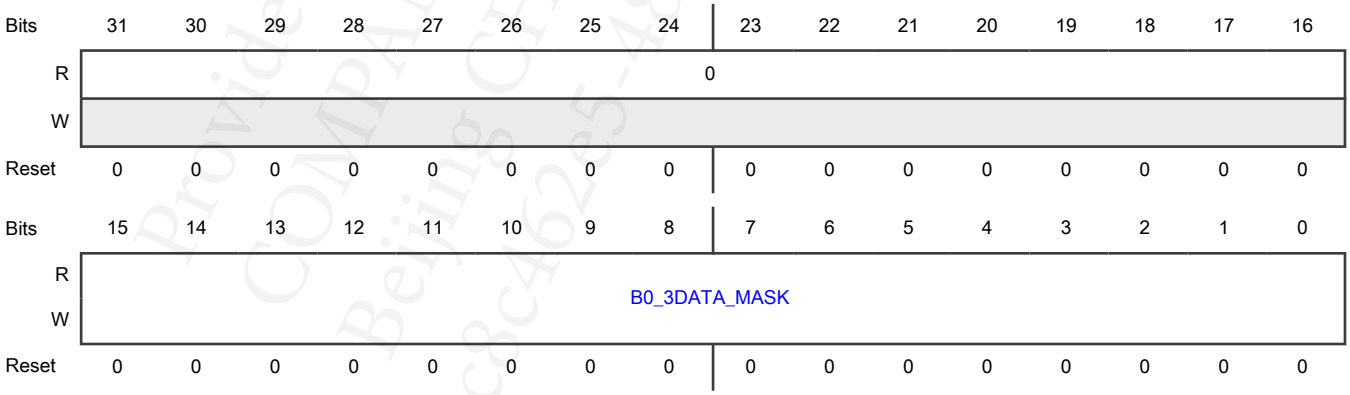
Offset

Register	Offset
EICHD1_WORD1	144h
EICHD2_WORD1	184h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.12.8 Error Injection Channel Descriptor n, Word2 (EICHD1_WORD2 - EICHD14_WORD2)

Offset

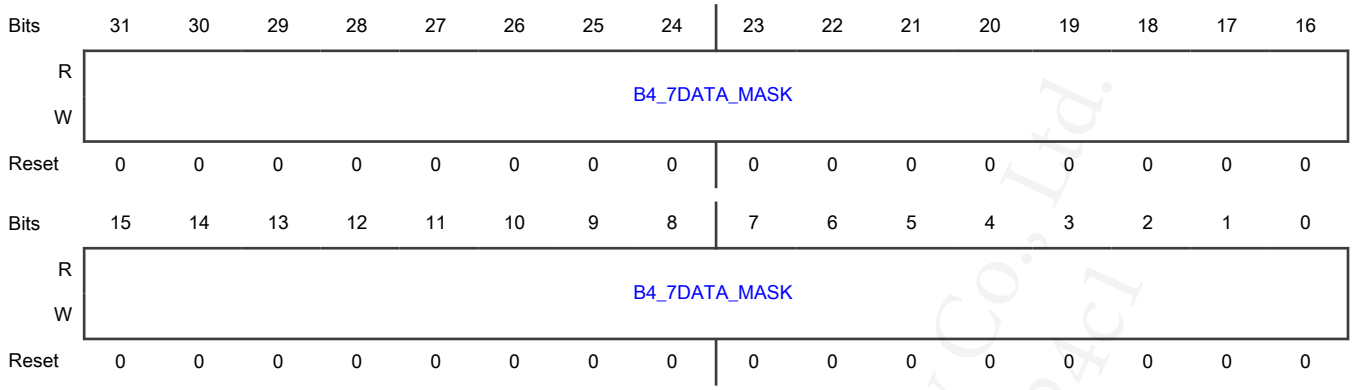
Register	Offset
EICHD1_WORD2	148h
EICHD6_WORD2	288h
EICHD12_WORD2	408h
EICHD13_WORD2	448h
EICHD14_WORD2	488h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CFAuto Automotive Technology Co., Ltd.
c8c462e5-48a8-b251-3901-3b6240

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Bytes 4-7 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

73.12.9 Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0)

Offset

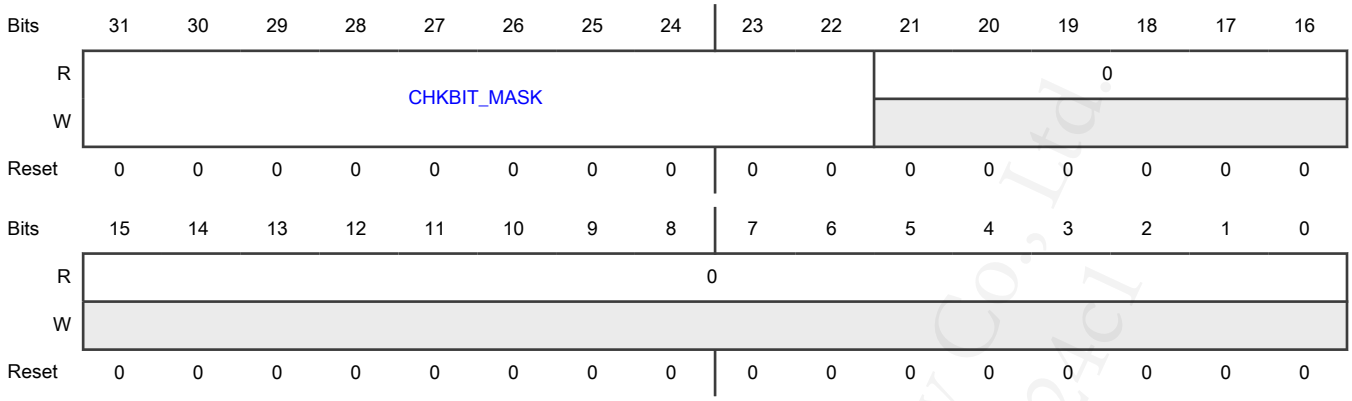
Register	Offset
EICHD2_WORD0	180h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing HHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b2f3-3903d3ab624c1

Diagram



Fields

Field	Function
31-22 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[9:0] (10 bits wide), CHKBIT_MASK[9] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
21-0 —	Reserved

73.12.10 Error Injection Channel Descriptor n, Word0 (EICHD3_WORD0 - EICHD4_WORD0)

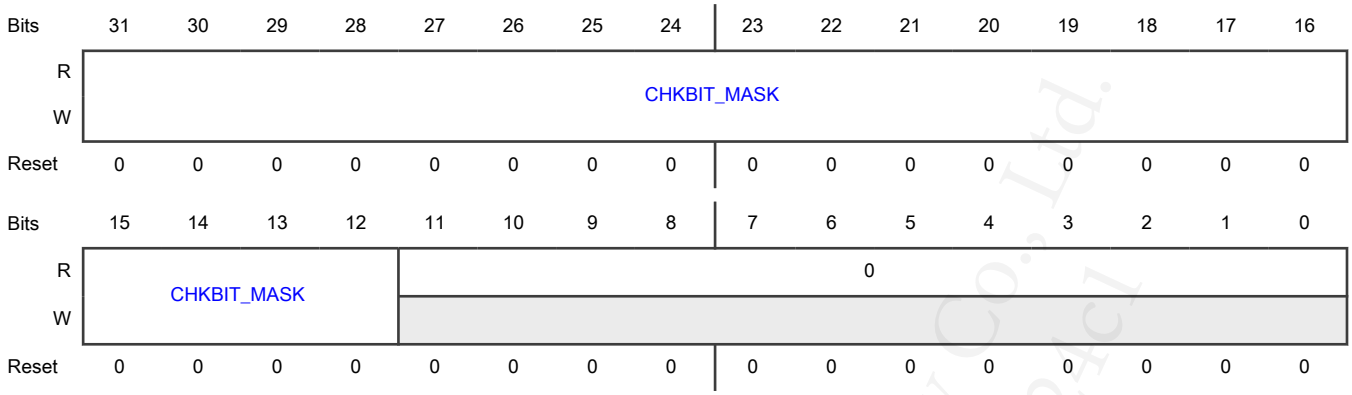
Offset

Register	Offset
EICHD3_WORD0	1C0h
EICHD4_WORD0	200h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[19:0] (20 bits wide), CHKBIT_MASK[19] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
11-0 —	Reserved

73.12.11 Error Injection Channel Descriptor n, Word1 (EICHD3_WORD1 - EICHD5_WORD1)

Offset

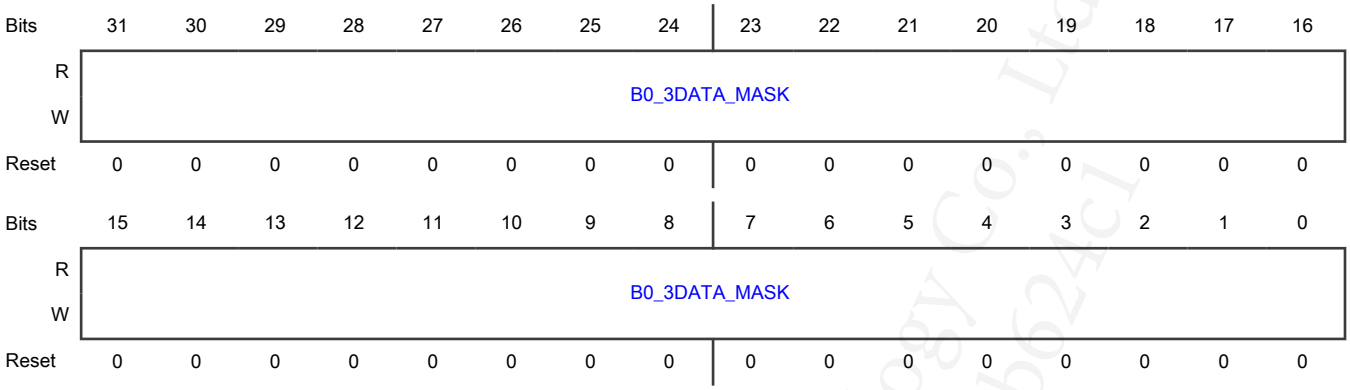
Register	Offset
EICHD3_WORD1	1C4h
EICHD4_WORD1	204h
EICHD5_WORD1	244h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target

RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.12.12 Error Injection Channel Descriptor 5, Word0 (EICHD5_WORD0)

Offset

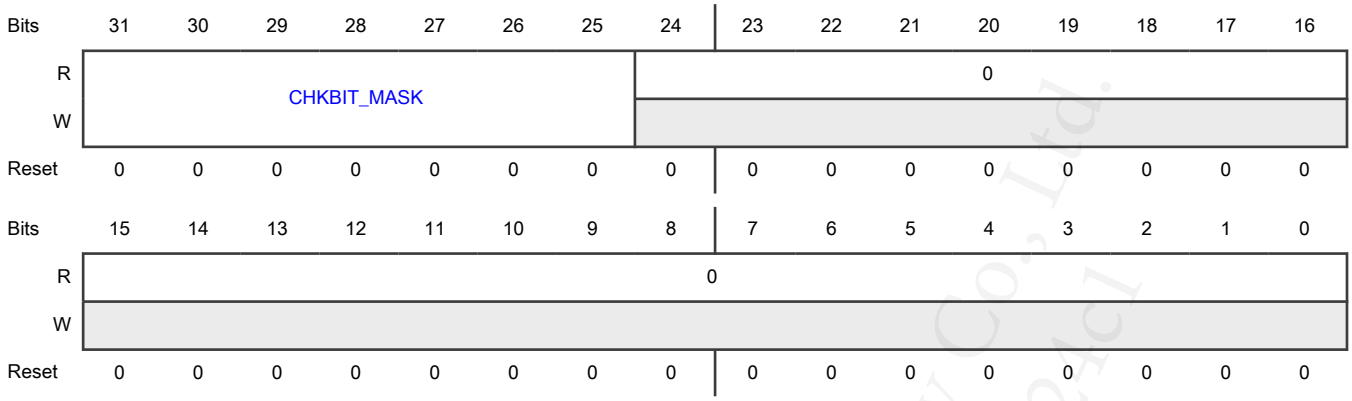
Register	Offset
EICHD5_WORD0	240h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Beijing CHJ Automotive Technology Co., Ltd.
COMPANY PROPRIETARY
Provided under NDA only
c8c4655-4821-4a3b-b251-590d3ab624c1

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.12.13 Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0)

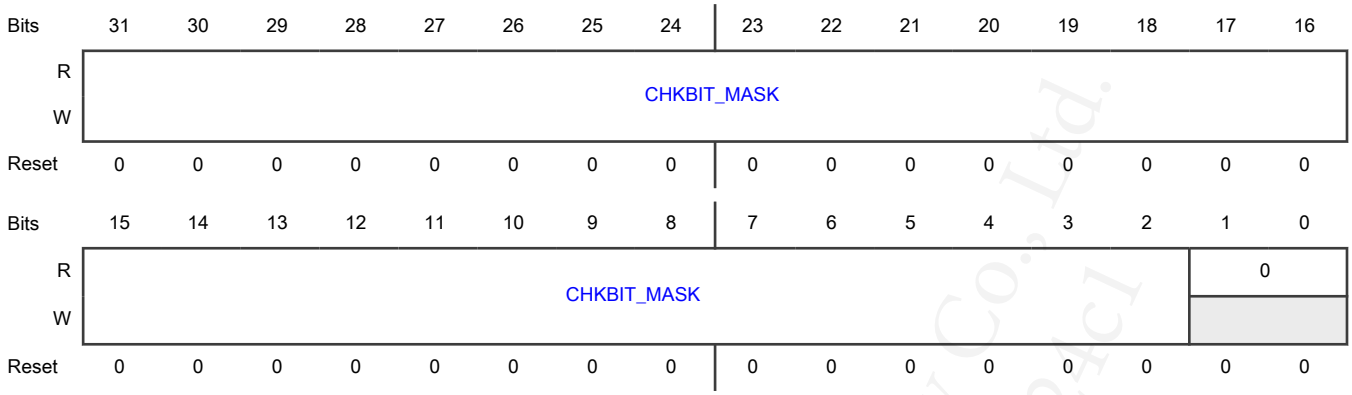
Offset

Register	Offset
EICHD6_WORD0	280h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-2 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[29:0] (30 bits wide), CHKBIT_MASK[29] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
1-0 —	Reserved

73.12.14 Error Injection Channel Descriptor n, Word1 (EICHD6_WORD1 - EICHD7_WORD1)

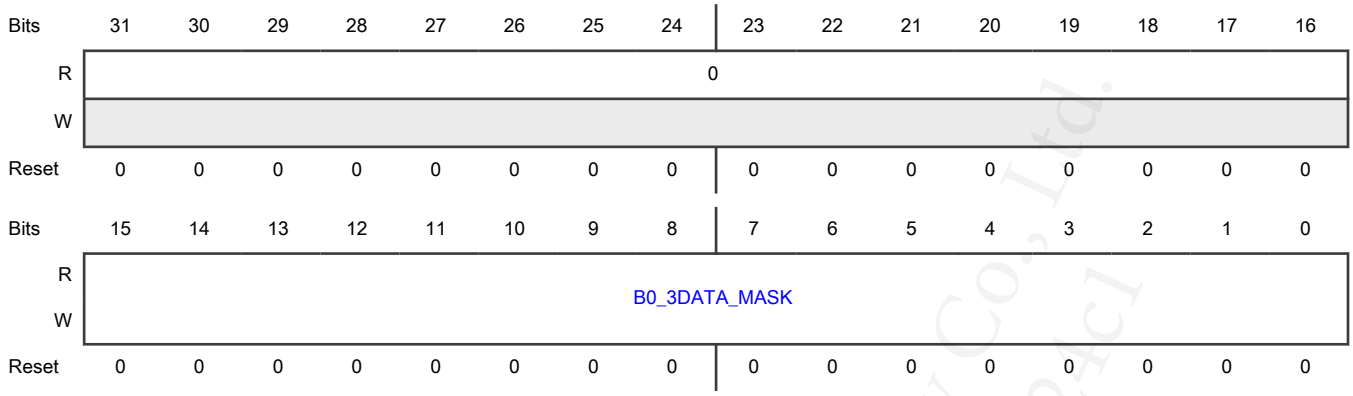
Offset

Register	Offset
EICHD6_WORD1	284h
EICHD7_WORD1	2C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.12.15 Error Injection Channel Descriptor 7, Word0 (EICHD7_WORD0)

Offset

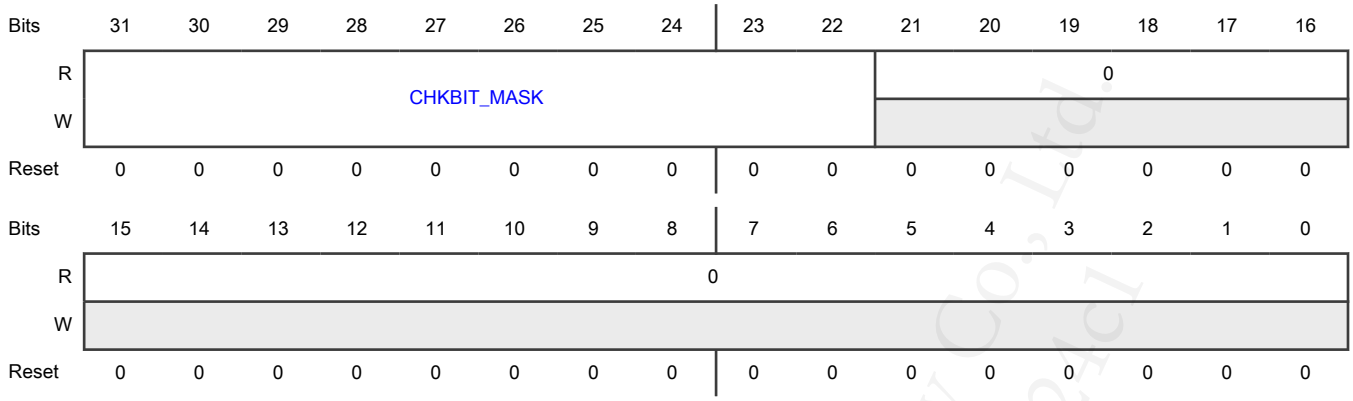
Register	Offset
EICHD7_WORD0	2C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b513-90d3ab624c1

Diagram



Fields

Field	Function
31-22 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[9:0] (10 bits wide), CHKBIT_MASK[9] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
21-0 —	Reserved

73.12.16 Error Injection Channel Descriptor n, Word0 (EICHD8_WORD0 - EICHD9_WORD0)

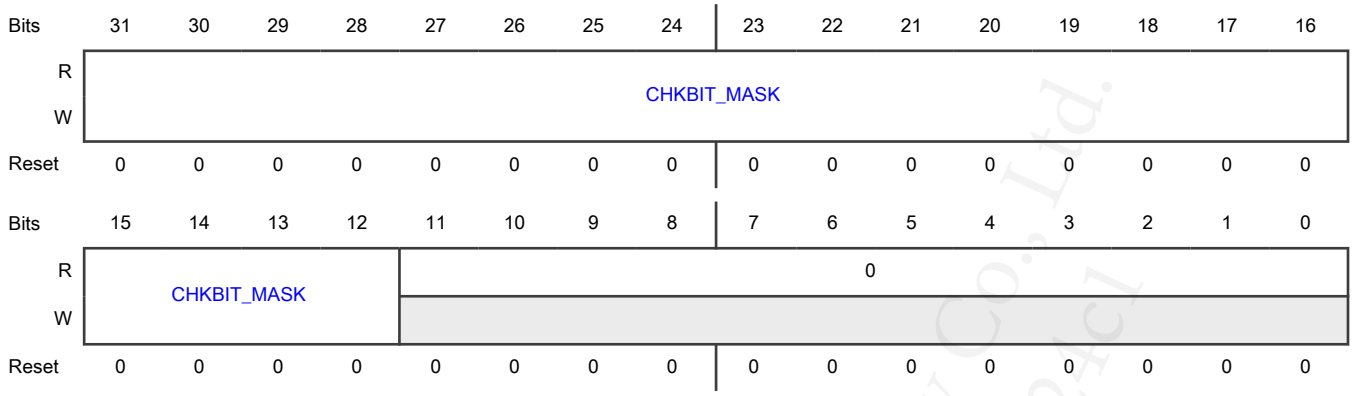
Offset

Register	Offset
EICHD8_WORD0	300h
EICHD9_WORD0	340h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[19:0] (20 bits wide), CHKBIT_MASK[19] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
11-0 —	Reserved

73.12.17 Error Injection Channel Descriptor n, Word1 (EICHD8_WORD1 - EICHD13_WORD1)

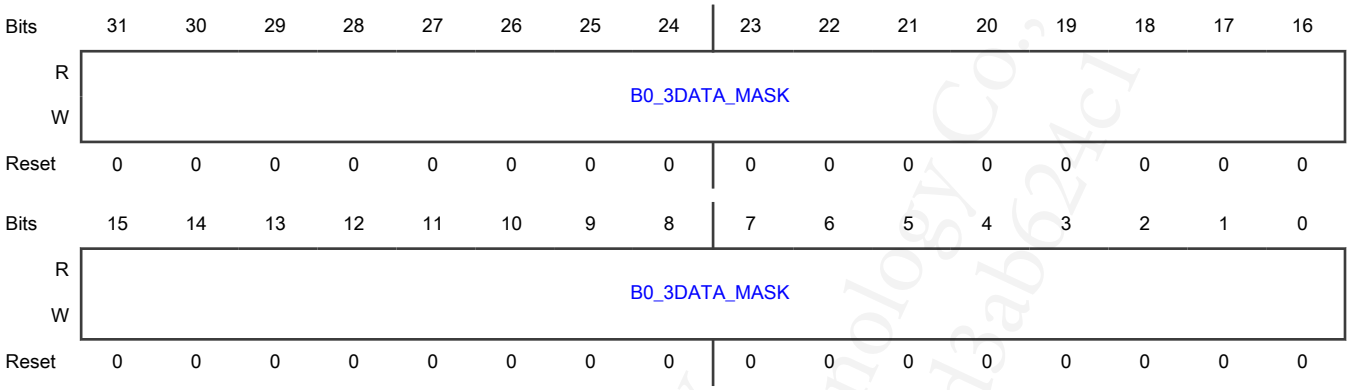
Offset

Register	Offset
EICHD8_WORD1	304h
EICHD9_WORD1	344h
EICHD10_WORD1	384h
EICHD11_WORD1	3C4h
EICHD12_WORD1	404h
EICHD13_WORD1	444h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.12.18 Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0)

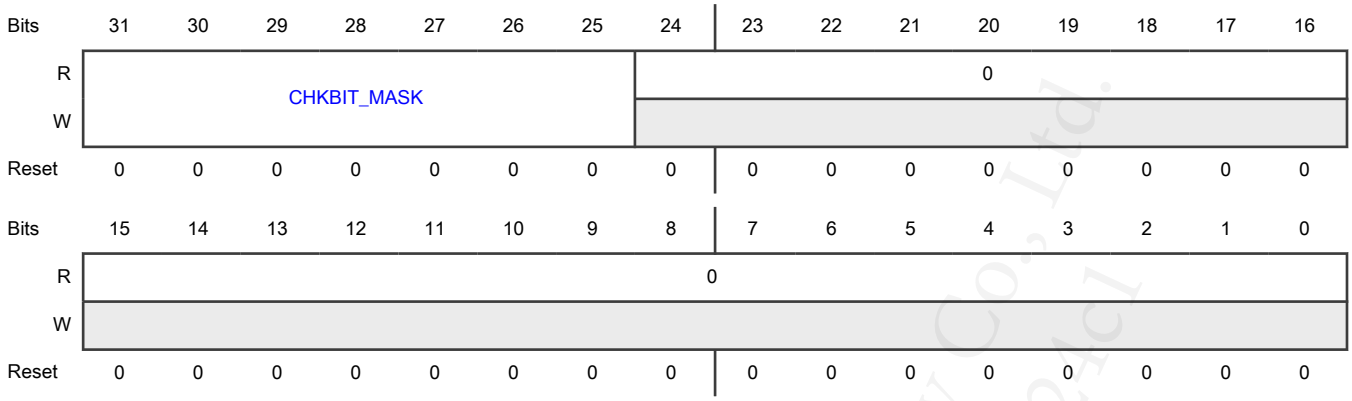
Offset

Register	Offset
EICHD10_WORD0	380h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.12.19 Error Injection Channel Descriptor 11, Word0 (EICHD11_WORD0)

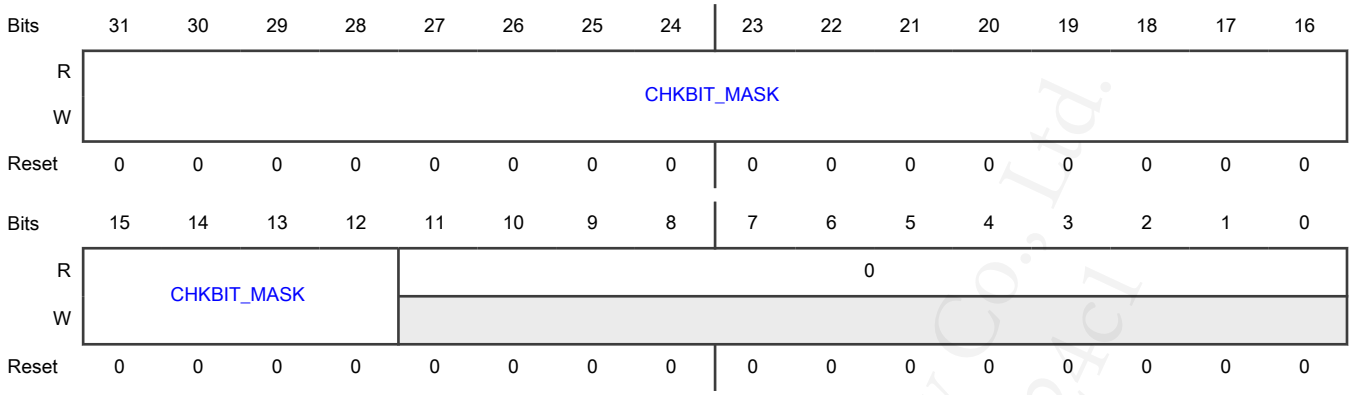
Offset

Register	Offset
EICHD11_WORD0	3C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[19:0] (20 bits wide), CHKBIT_MASK[19] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
11-0 —	Reserved

73.12.20 Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0)

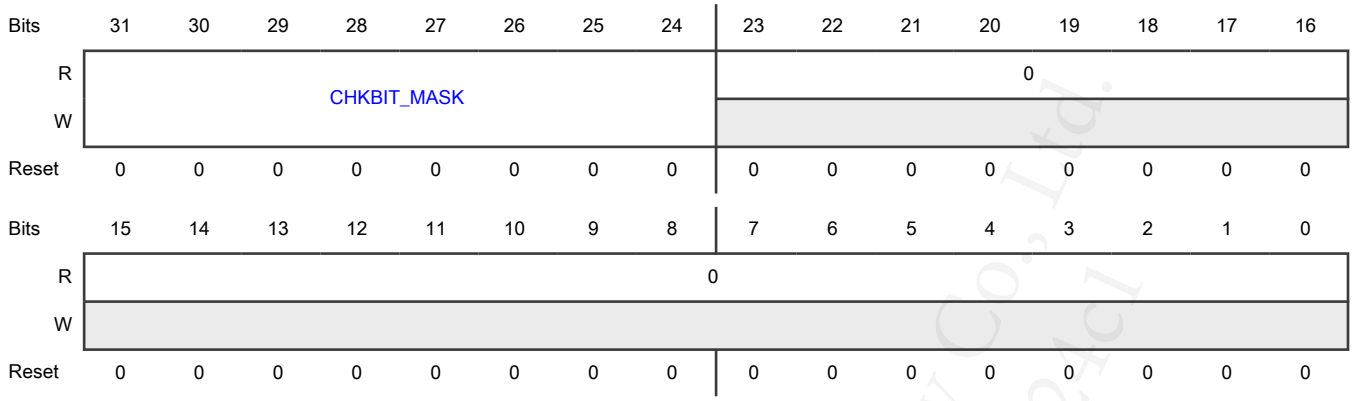
Offset

Register	Offset
EICHD12_WORD0	400h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.12.21 Error Injection Channel Descriptor n, Word0 (EICHD13_WORD0 - EICHD14_WORD0)

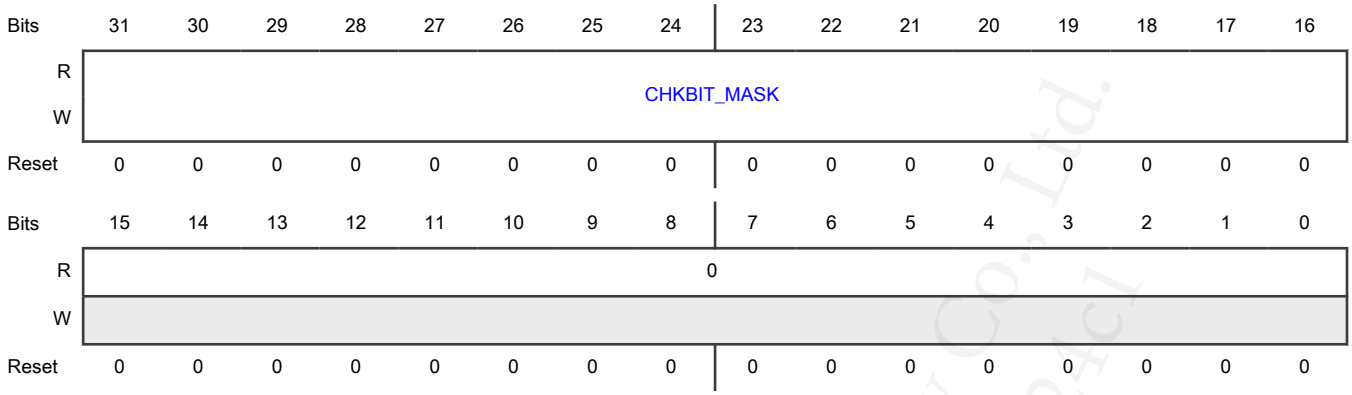
Offset

Register	Offset
EICHD13_WORD0	440h
EICHD14_WORD0	480h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[15:0] (16 bits wide), CHKBIT_MASK[15] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
15-0 —	Reserved

73.12.22 Error Injection Channel Descriptor n, Word3 (EICHD13_WORD3 - EICHD14_WORD3)

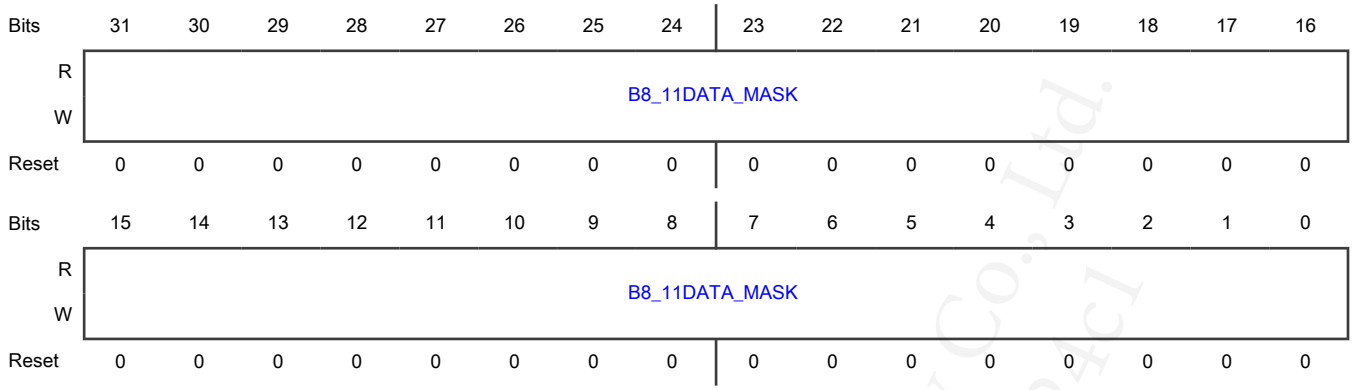
Offset

Register	Offset
EICHD13_WORD3	44Ch
EICHD14_WORD3	48Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to bytes 8–11 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Bytes 8-11 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 8-11 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 8-11 on the read data bus is inverted.

73.12.23 Error Injection Channel Descriptor n, Word4 (EICHD13_WORD4 - EICHD14_WORD4)

Offset

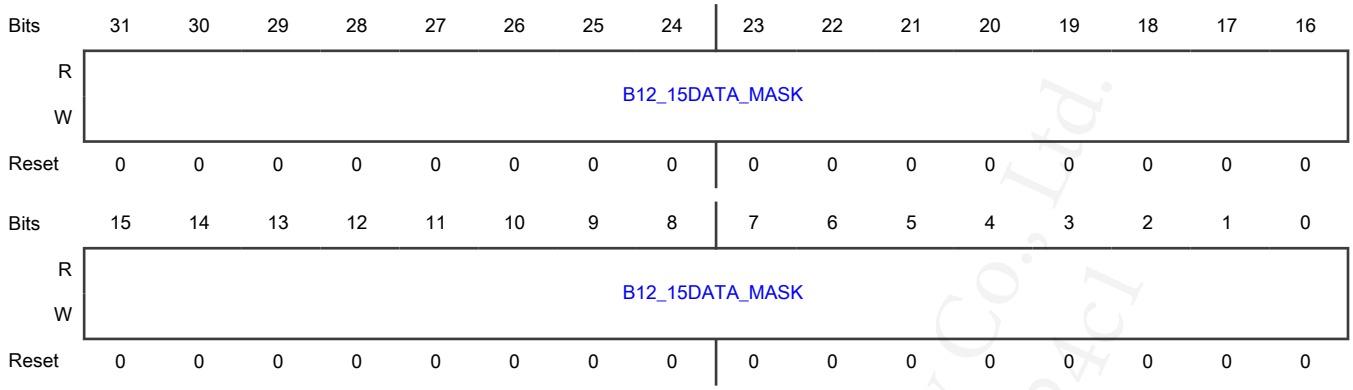
Register	Offset
EICHD13_WORD4	450h
EICHD14_WORD4	490h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to bytes 12–15 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co., Ltd. c8c46a2-4821-48a8-b2b1-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Bytes 12-15 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 12-15 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 12-15 on the read data bus is inverted.

73.12.24 Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)

Offset

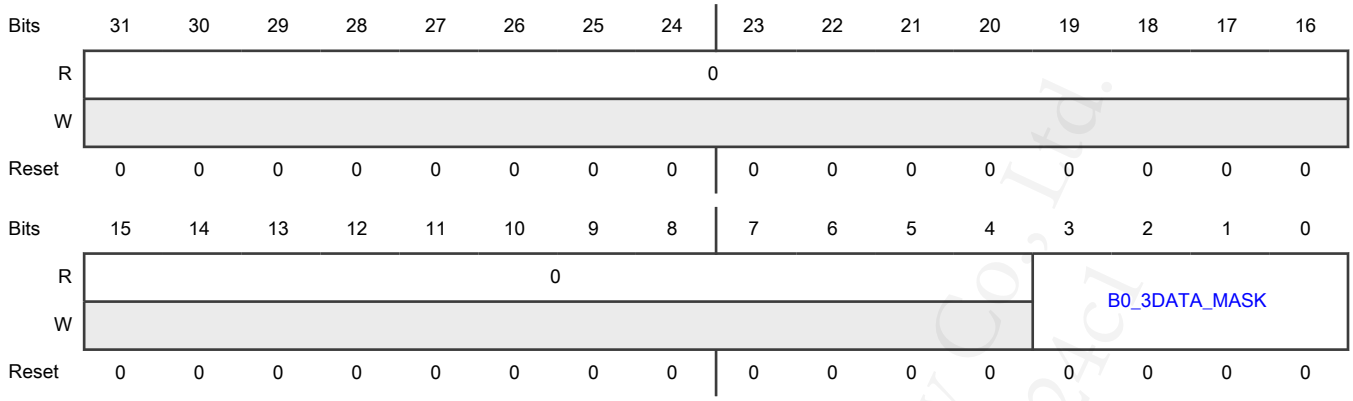
Register	Offset
EICHD14_WORD1	484h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e4-821-48a6-b2-1390d3ab624c1

Diagram



Fields

Field	Function
31-4 —	Reserved
3-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.13 EIM_PFE2 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.
  - Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 593. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	16	15-0	—	—
1	16	15-0	—	—
2	16	15-0	—	—
3	16	15-0	—	—
4	8	7-0	—	—
5	8	7-0	—	—
6	16	15-0	—	—
7	8	7-0	—	—
8	8	7-0	—	—
9	16	15-0	—	—
10	8	7-0	—	—
11	8	7-0	—	—
12	16	15-0	—	—
13	8	7-0	—	—
14	8	7-0	—	—
15	16	15-0	—	—
16	64	63-32	31-0	—

Table continues on the next page...

Table 593. Error injection channel descriptor: DATA_MASK details (continued)

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
17	64	63-32	31-0	—
18	64	63-32	31-0	—
19	64	63-32	31-0	—
20	64	63-32	31-0	—
21	64	63-32	31-0	—
22	256	255-224	223-192	191-160
23	64	63-32	31-0	—
24	64	63-32	31-0	—
25	64	63-32	31-0	—
26	64	63-32	31-0	—
27	64	63-32	31-0	—
28	64	63-32	31-0	—
29	64	63-32	31-0	—
30	64	63-32	31-0	—
31	16	15-0	—	—

Table 594. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
22	159-128	127-96	95-64	63-32	31-0

73.13.1 EIM_PFE2 memory map

EIM_PFE2 base address: 4405_6000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
384h	Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)	32	RW	0000_0000h
3C4h	Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)	32	RW	0000_0000h
404h	Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)	32	RW	0000_0000h
444h	Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)	32	RW	0000_0000h
484h	Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)	32	RW	0000_0000h
4C4h	Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)	32	RW	0000_0000h
500h	Error Injection Channel Descriptor 16, Word0 (EICHD16_WORD0)	32	RW	0000_0000h
504h	Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)	32	RW	0000_0000h
508h	Error Injection Channel Descriptor 16, Word2 (EICHD16_WORD2)	32	RW	0000_0000h
540h	Error Injection Channel Descriptor 17, Word0 (EICHD17_WORD0)	32	RW	0000_0000h
544h	Error Injection Channel Descriptor 17, Word1 (EICHD17_WORD1)	32	RW	0000_0000h
548h	Error Injection Channel Descriptor 17, Word2 (EICHD17_WORD2)	32	RW	0000_0000h
580h	Error Injection Channel Descriptor 18, Word0 (EICHD18_WORD0)	32	RW	0000_0000h
584h	Error Injection Channel Descriptor 18, Word1 (EICHD18_WORD1)	32	RW	0000_0000h
588h	Error Injection Channel Descriptor 18, Word2 (EICHD18_WORD2)	32	RW	0000_0000h
5C0h	Error Injection Channel Descriptor 19, Word0 (EICHD19_WORD0)	32	RW	0000_0000h
5C4h	Error Injection Channel Descriptor 19, Word1 (EICHD19_WORD1)	32	RW	0000_0000h
5C8h	Error Injection Channel Descriptor 19, Word2 (EICHD19_WORD2)	32	RW	0000_0000h
600h	Error Injection Channel Descriptor 20, Word0 (EICHD20_WORD0)	32	RW	0000_0000h
604h	Error Injection Channel Descriptor 20, Word1 (EICHD20_WORD1)	32	RW	0000_0000h
608h	Error Injection Channel Descriptor 20, Word2 (EICHD20_WORD2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
640h	Error Injection Channel Descriptor 21, Word0 (EICHD21_WORD0)	32	RW	0000_0000h
644h	Error Injection Channel Descriptor 21, Word1 (EICHD21_WORD1)	32	RW	0000_0000h
648h	Error Injection Channel Descriptor 21, Word2 (EICHD21_WORD2)	32	RW	0000_0000h
680h	Error Injection Channel Descriptor 22, Word0 (EICHD22_WORD0)	32	RW	0000_0000h
684h	Error Injection Channel Descriptor 22, Word1 (EICHD22_WORD1)	32	RW	0000_0000h
688h	Error Injection Channel Descriptor 22, Word2 (EICHD22_WORD2)	32	RW	0000_0000h
68Ch	Error Injection Channel Descriptor 22, Word3 (EICHD22_WORD3)	32	RW	0000_0000h
690h	Error Injection Channel Descriptor 22, Word4 (EICHD22_WORD4)	32	RW	0000_0000h
694h	Error Injection Channel Descriptor 22, Word5 (EICHD22_WORD5)	32	RW	0000_0000h
698h	Error Injection Channel Descriptor 22, Word6 (EICHD22_WORD6)	32	RW	0000_0000h
69Ch	Error Injection Channel Descriptor 22, Word7 (EICHD22_WORD7)	32	RW	0000_0000h
6A0h	Error Injection Channel Descriptor 22, Word8 (EICHD22_WORD8)	32	RW	0000_0000h
6C0h	Error Injection Channel Descriptor 23, Word0 (EICHD23_WORD0)	32	RW	0000_0000h
6C4h	Error Injection Channel Descriptor 23, Word1 (EICHD23_WORD1)	32	RW	0000_0000h
6C8h	Error Injection Channel Descriptor 23, Word2 (EICHD23_WORD2)	32	RW	0000_0000h
700h	Error Injection Channel Descriptor 24, Word0 (EICHD24_WORD0)	32	RW	0000_0000h
704h	Error Injection Channel Descriptor 24, Word1 (EICHD24_WORD1)	32	RW	0000_0000h
708h	Error Injection Channel Descriptor 24, Word2 (EICHD24_WORD2)	32	RW	0000_0000h
740h	Error Injection Channel Descriptor 25, Word0 (EICHD25_WORD0)	32	RW	0000_0000h
744h	Error Injection Channel Descriptor 25, Word1 (EICHD25_WORD1)	32	RW	0000_0000h
748h	Error Injection Channel Descriptor 25, Word2 (EICHD25_WORD2)	32	RW	0000_0000h
780h	Error Injection Channel Descriptor 26, Word0 (EICHD26_WORD0)	32	RW	0000_0000h
784h	Error Injection Channel Descriptor 26, Word1 (EICHD26_WORD1)	32	RW	0000_0000h
788h	Error Injection Channel Descriptor 26, Word2 (EICHD26_WORD2)	32	RW	0000_0000h
7C0h	Error Injection Channel Descriptor 27, Word0 (EICHD27_WORD0)	32	RW	0000_0000h
7C4h	Error Injection Channel Descriptor 27, Word1 (EICHD27_WORD1)	32	RW	0000_0000h
7C8h	Error Injection Channel Descriptor 27, Word2 (EICHD27_WORD2)	32	RW	0000_0000h
800h	Error Injection Channel Descriptor 28, Word0 (EICHD28_WORD0)	32	RW	0000_0000h
804h	Error Injection Channel Descriptor 28, Word1 (EICHD28_WORD1)	32	RW	0000_0000h
808h	Error Injection Channel Descriptor 28, Word2 (EICHD28_WORD2)	32	RW	0000_0000h
840h	Error Injection Channel Descriptor 29, Word0 (EICHD29_WORD0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
844h	Error Injection Channel Descriptor 29, Word1 (EICHD29_WORD1)	32	RW	0000_0000h
848h	Error Injection Channel Descriptor 29, Word2 (EICHD29_WORD2)	32	RW	0000_0000h
880h	Error Injection Channel Descriptor 30, Word0 (EICHD30_WORD0)	32	RW	0000_0000h
884h	Error Injection Channel Descriptor 30, Word1 (EICHD30_WORD1)	32	RW	0000_0000h
888h	Error Injection Channel Descriptor 30, Word2 (EICHD30_WORD2)	32	RW	0000_0000h
8C0h	Error Injection Channel Descriptor 31, Word0 (EICHD31_WORD0)	32	RW	0000_0000h
8C4h	Error Injection Channel Descriptor 31, Word1 (EICHD31_WORD1)	32	RW	0000_0000h

73.13.2 Error Injection Module Configuration Register (EIMCR)

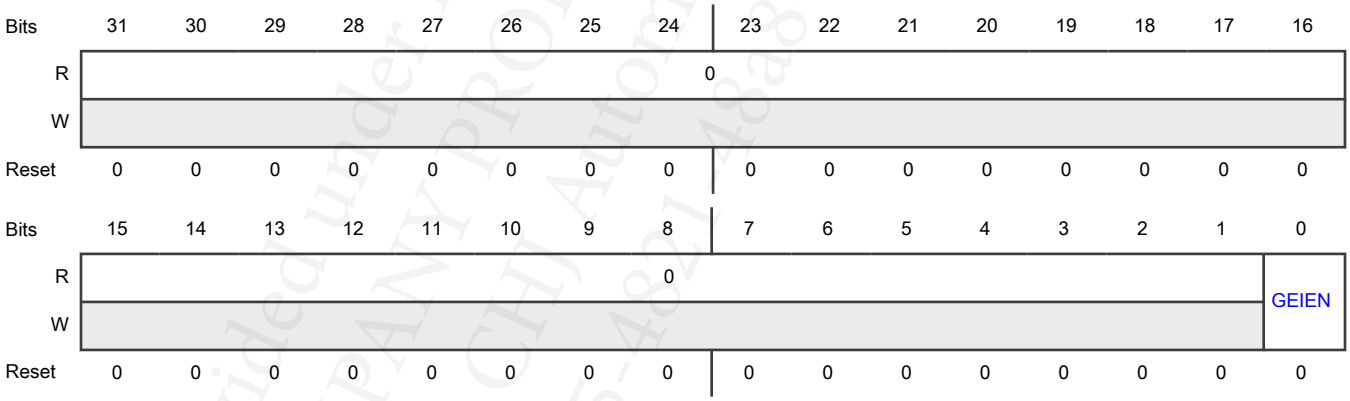
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Fields

Field	Function
31-1 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

73.13.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EICH0EN	EICH1EN	EICH2EN	EICH3EN	EICH4EN	EICH5EN	EICH6EN	EICH7EN	EICH8EN	EICH9EN	EICH10EN	EICH11EN	EICH12EN	EICH13EN	EICH14EN	EICH15EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EICH16EN	EICH17EN	EICH18EN	EICH19EN	EICH20EN	EICH21EN	EICH22EN	EICH23EN	EICH24EN	EICH25EN	EICH26EN	EICH27EN	EICH28EN	EICH29EN	EICH30EN	EICH31EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9
21 EICH10EN	Error Injection Channel 10 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 10 1b - Error injection is enabled on Error Injection Channel 10
20 EICH11EN	Error Injection Channel 11 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 11 1b - Error injection is enabled on Error Injection Channel 11
19 EICH12EN	Error Injection Channel 12 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 12 1b - Error injection is enabled on Error Injection Channel 12
18 EICH13EN	Error Injection Channel 13 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 13 1b - Error injection is enabled on Error Injection Channel 13
17 EICH14EN	Error Injection Channel 14 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 14 1b - Error injection is enabled on Error Injection Channel 14

Table continues on the next page...

Table continued from the previous page...

Field	Function
16 EICH15EN	Error Injection Channel 15 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 15 1b - Error injection is enabled on Error Injection Channel 15
15 EICH16EN	Error Injection Channel 16 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 16 1b - Error injection is enabled on Error Injection Channel 16
14 EICH17EN	Error Injection Channel 17 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 17 1b - Error injection is enabled on Error Injection Channel 17
13 EICH18EN	Error Injection Channel 18 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 18 1b - Error injection is enabled on Error Injection Channel 18
12 EICH19EN	Error Injection Channel 19 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 19 1b - Error injection is enabled on Error Injection Channel 19
11 EICH20EN	Error Injection Channel 20 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 20 1b - Error injection is enabled on Error Injection Channel 20
10 EICH21EN	Error Injection Channel 21 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 21 1b - Error injection is enabled on Error Injection Channel 21
9 EICH22EN	Error Injection Channel 22 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 22 1b - Error injection is enabled on Error Injection Channel 22
8 EICH23EN	Error Injection Channel 23 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 23 1b - Error injection is enabled on Error Injection Channel 23
7 EICH24EN	Error Injection Channel 24 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 24 1b - Error injection is enabled on Error Injection Channel 24
6 EICH25EN	Error Injection Channel 25 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 25 1b - Error injection is enabled on Error Injection Channel 25

Table continues on the next page...

Table continued from the previous page...

Field	Function
5 EICH26EN	Error Injection Channel 26 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 26 1b - Error injection is enabled on Error Injection Channel 26
4 EICH27EN	Error Injection Channel 27 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 27 1b - Error injection is enabled on Error Injection Channel 27
3 EICH28EN	Error Injection Channel 28 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 28 1b - Error injection is enabled on Error Injection Channel 28
2 EICH29EN	Error Injection Channel 29 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 29 1b - Error injection is enabled on Error Injection Channel 29
1 EICH30EN	Error Injection Channel 30 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 30 1b - Error injection is enabled on Error Injection Channel 30
0 EICH31EN	Error Injection Channel 31 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 31 1b - Error injection is enabled on Error Injection Channel 31

73.13.4 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD3_WORD1)

Offset

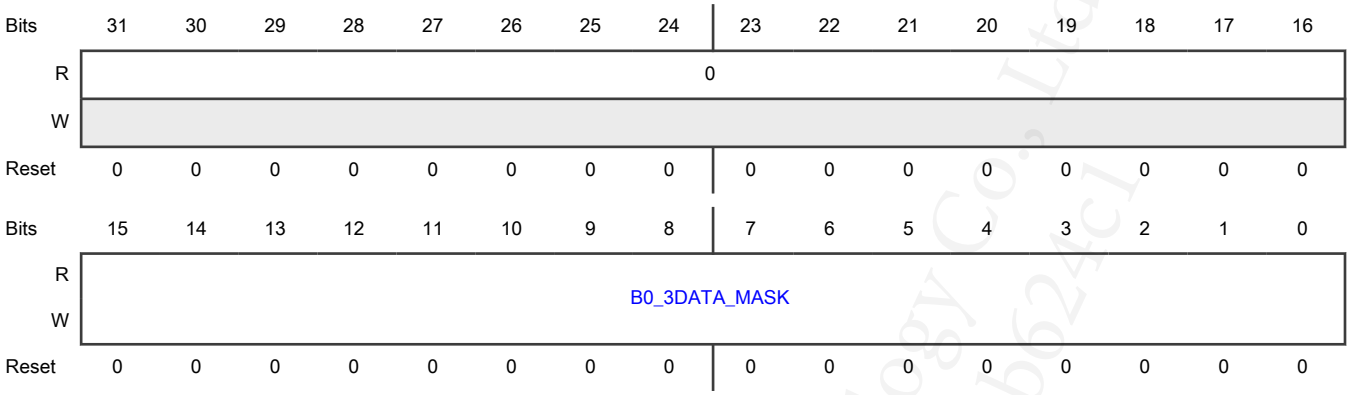
Register	Offset
EICHD0_WORD1	104h
EICHD1_WORD1	144h
EICHD2_WORD1	184h
EICHD3_WORD1	1C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target

RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.13.5 Error Injection Channel Descriptor n, Word1 (EICHD4_WORD1 - EICHD5_WORD1)

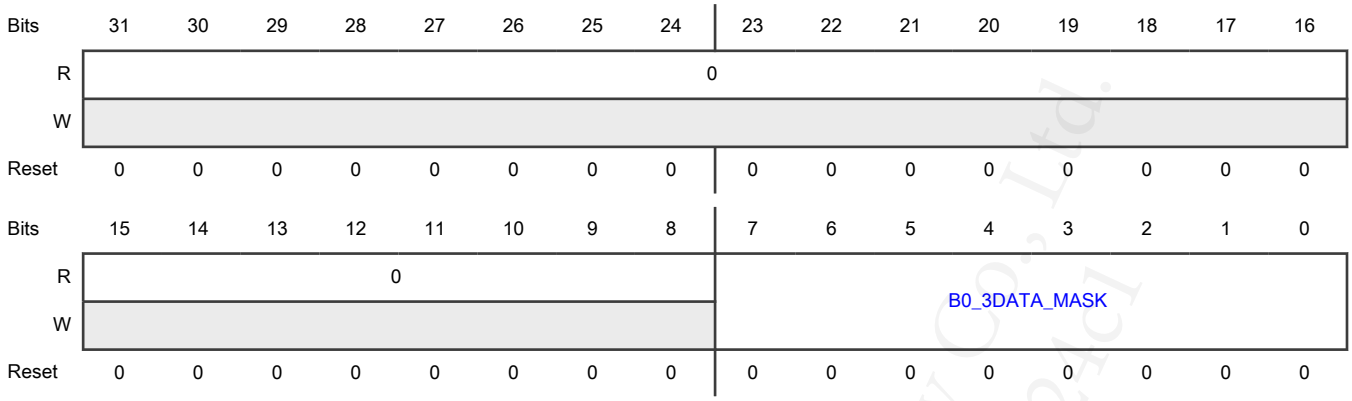
Offset

Register	Offset
EICHD4_WORD1	204h
EICHD5_WORD1	244h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.13.6 Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)

Offset

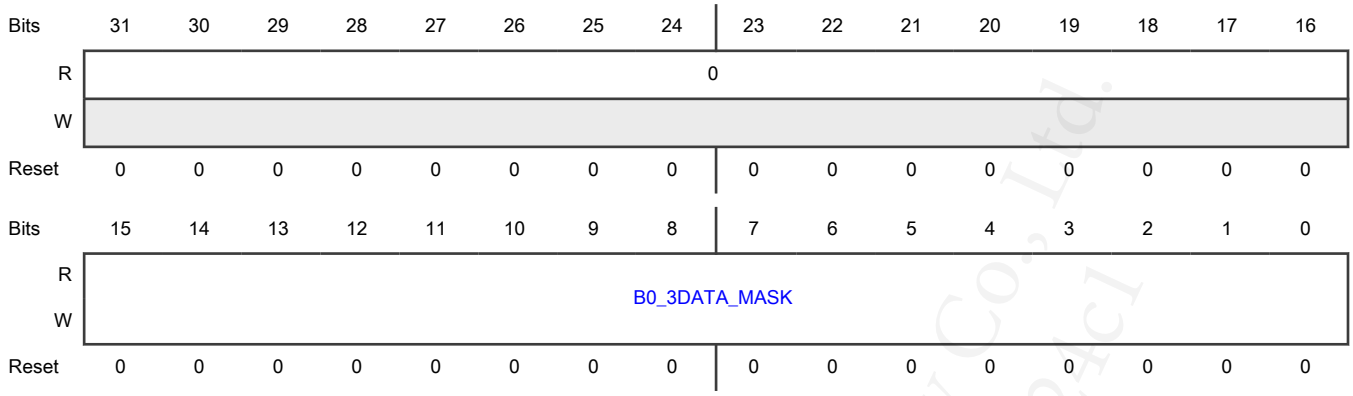
Register	Offset
EICHD6_WORD1	284h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-45a8-b251-390d3ab621c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.13.7 Error Injection Channel Descriptor n, Word1 (EICHD7_WORD1 - EICHD8_WORD1)

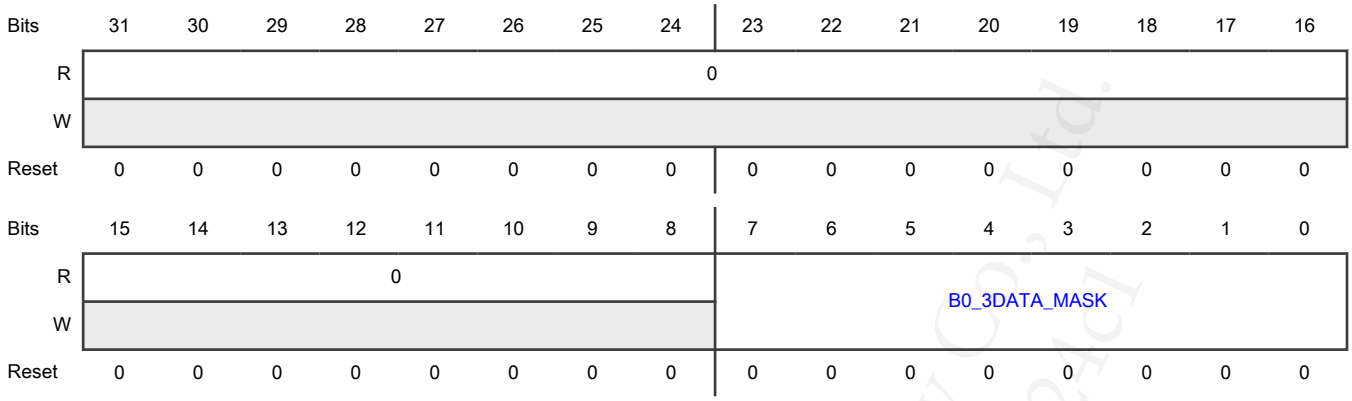
Offset

Register	Offset
EICHD7_WORD1	2C4h
EICHD8_WORD1	304h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.13.8 Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)

Offset

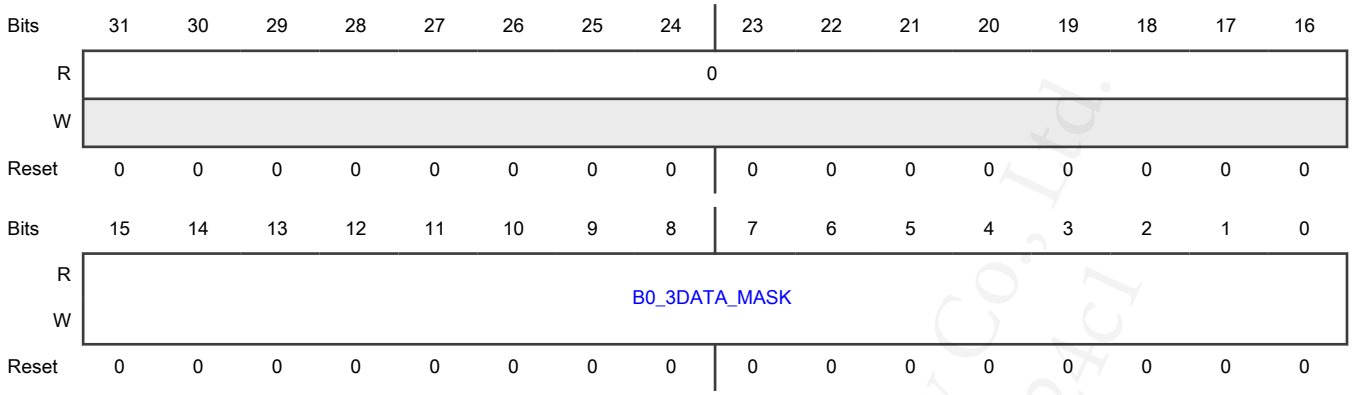
Register	Offset
EICHD9_WORD1	344h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-45a8-b251-390d3ab621c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.13.9 Error Injection Channel Descriptor n, Word1 (EICHD10_WORD1 - EICHD11_WORD1)

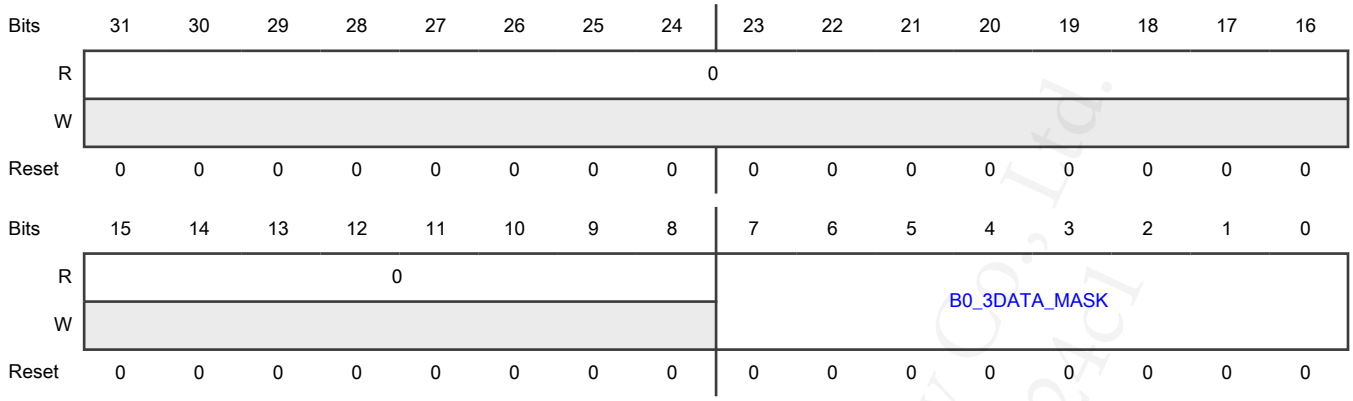
Offset

Register	Offset
EICHD10_WORD1	384h
EICHD11_WORD1	3C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.13.10 Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)

Offset

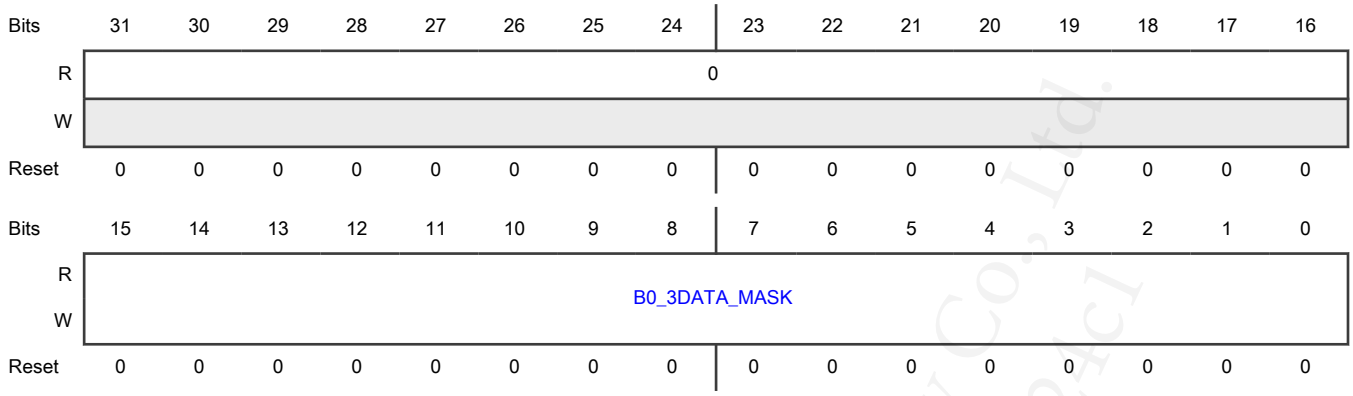
Register	Offset
EICHD12_WORD1	404h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b251-390d3ab621c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.13.11 Error Injection Channel Descriptor n, Word1 (EICHD13_WORD1 - EICHD14_WORD1)

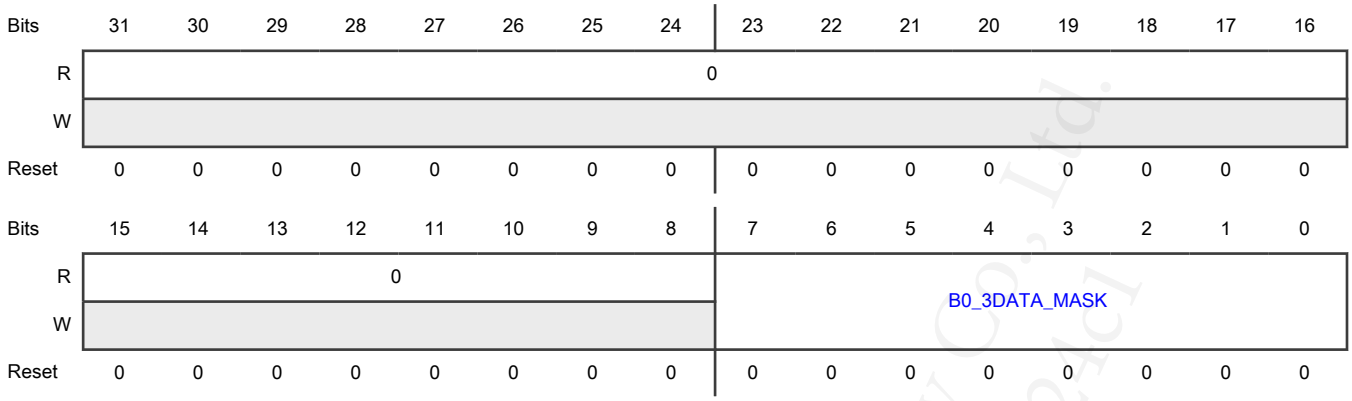
Offset

Register	Offset
EICHD13_WORD1	444h
EICHD14_WORD1	484h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.13.12 Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)

Offset

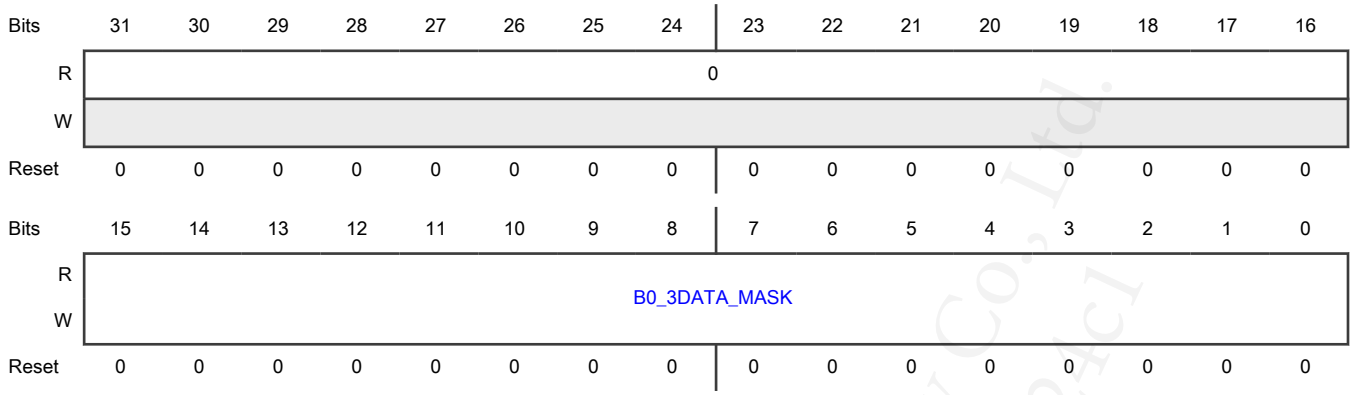
Register	Offset
EICHD15_WORD1	4C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b251-390d3ab621c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.13.13 Error Injection Channel Descriptor n, Word0 (EICHD16_WORD0 - EICHD21_WORD0)

Offset

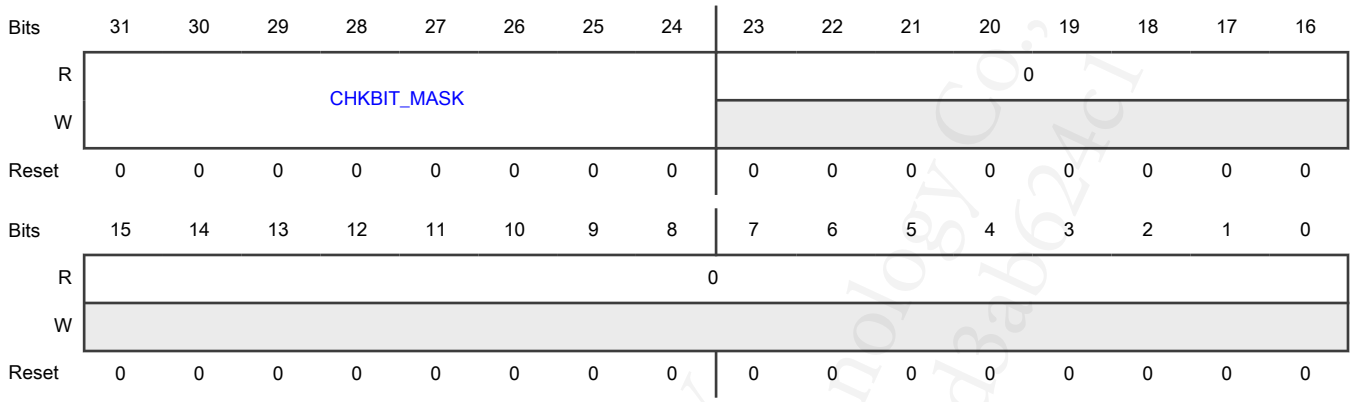
Register	Offset
EICHD16_WORD0	500h
EICHD17_WORD0	540h
EICHD18_WORD0	580h
EICHD19_WORD0	5C0h
EICHD20_WORD0	600h
EICHD21_WORD0	640h

Beijing CHJ Automotive Technology Co., Ltd. COMPANY PROPRIETARY Provided under NDA Only c8c462e5-4821-48a8-b251-390d3ab624c1

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.13.14 Error Injection Channel Descriptor n, Word1 (EICHD16_WORD1 - EICHD30_WORD1)

Offset

Register	Offset
EICHD16_WORD1	504h

Table continues on the next page...

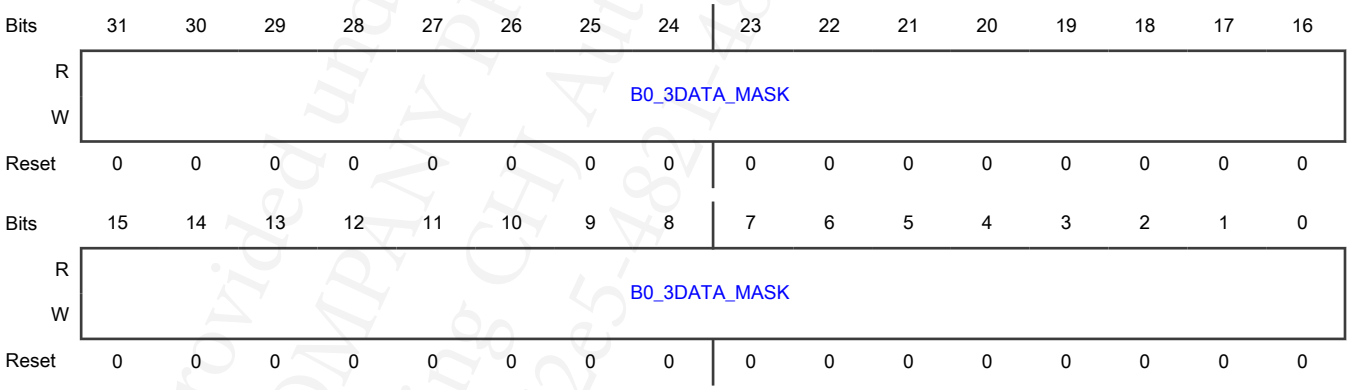
Table continued from the previous page...

Register	Offset
EICHD17_WORD1	544h
EICHD18_WORD1	584h
EICHD19_WORD1	5C4h
EICHD20_WORD1	604h
EICHD21_WORD1	644h
EICHD22_WORD1	684h
EICHD23_WORD1	6C4h
EICHD24_WORD1	704h
EICHD25_WORD1	744h
EICHD26_WORD1	784h
EICHD27_WORD1	7C4h
EICHD28_WORD1	804h
EICHD29_WORD1	844h
EICHD30_WORD1	884h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-9251-390d3ab624c1

Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.13.15 Error Injection Channel Descriptor n, Word2 (EICHD16_WORD2 - EICHD30_WORD2)

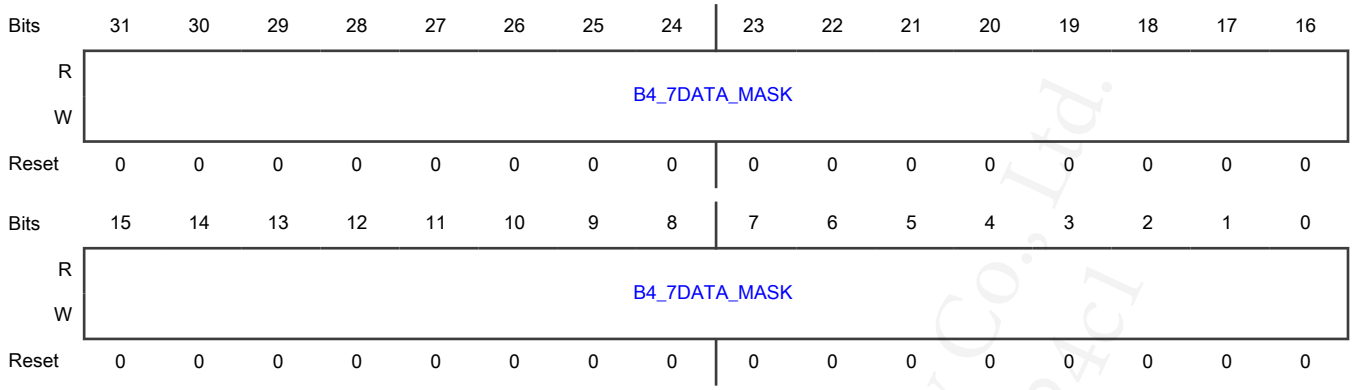
Offset

Register	Offset
EICHD16_WORD2	508h
EICHD17_WORD2	548h
EICHD18_WORD2	588h
EICHD19_WORD2	5C8h
EICHD20_WORD2	608h
EICHD21_WORD2	648h
EICHD22_WORD2	688h
EICHD23_WORD2	6C8h
EICHD24_WORD2	708h
EICHD25_WORD2	748h
EICHD26_WORD2	788h
EICHD27_WORD2	7C8h
EICHD28_WORD2	808h
EICHD29_WORD2	848h
EICHD30_WORD2	888h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Bytes 4-7 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

73.13.16 Error Injection Channel Descriptor 22, Word0 (EICHD22_WORD0)

Offset

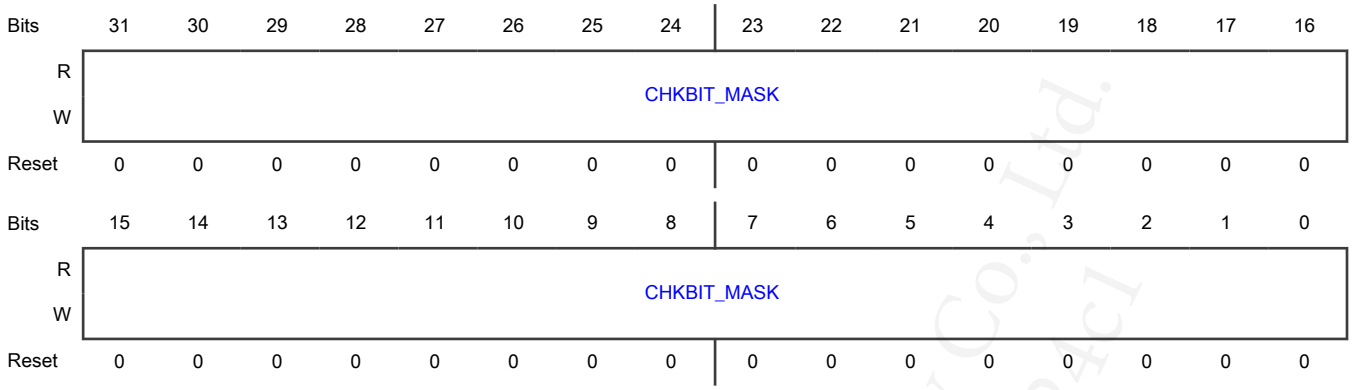
Register	Offset
EICHD22_WORD0	680h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing HHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b2f3-390d3ab624c1

Diagram



Fields

Field	Function
31-0 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[31:0] (32 bits wide), CHKBIT_MASK[31] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.

73.13.17 Error Injection Channel Descriptor 22, Word3 (EICHD22_WORD3)

Offset

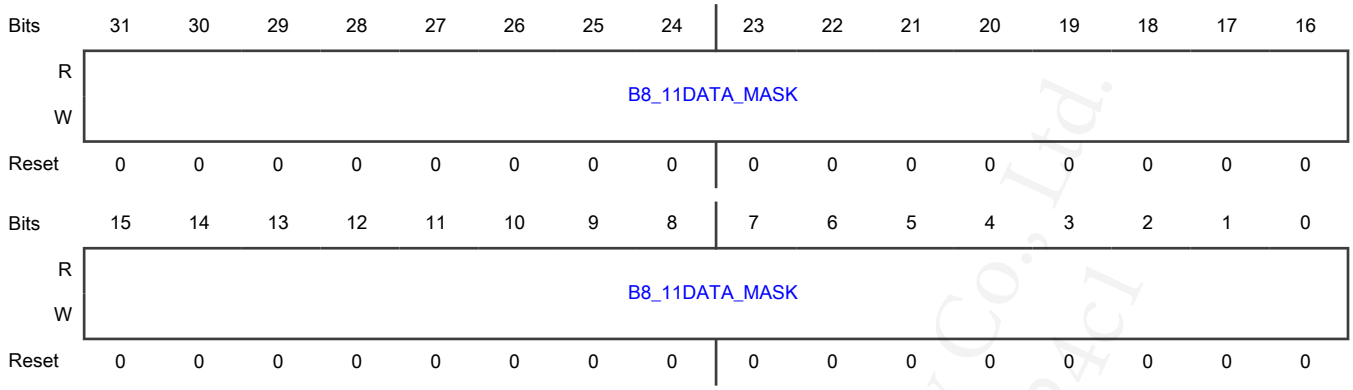
Register	Offset
EICHD22_WORD3	68Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to bytes 8–11 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NXP COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co.,Ltd. c8c0e25-4821-4ac1-b61-59d3ab624c1

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Bytes 8-11 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 8-11 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 8-11 on the read data bus is inverted.

73.13.18 Error Injection Channel Descriptor 22, Word4 (EICHD22_WORD4)

Offset

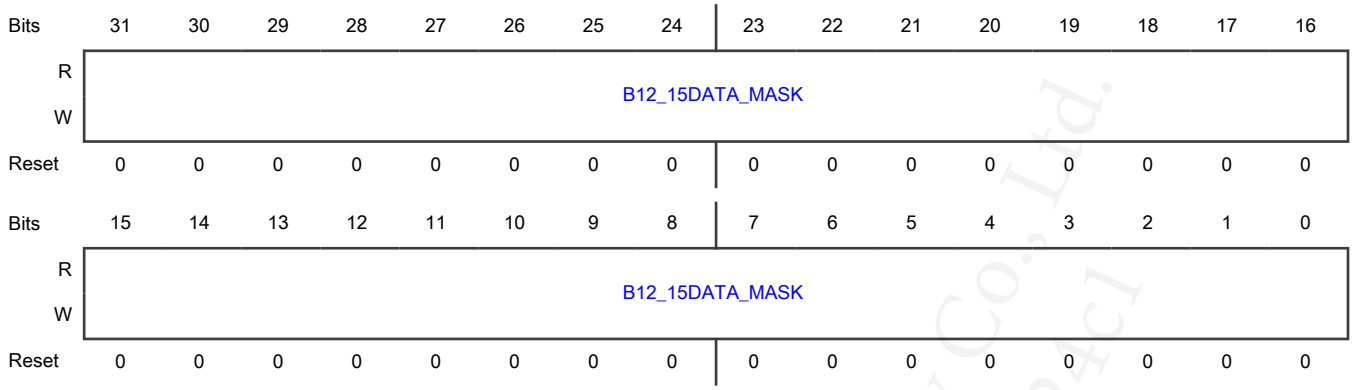
Register	Offset
EICHD22_WORD4	690h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to bytes 12–15 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46220-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Bytes 12-15 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 12-15 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 12-15 on the read data bus is inverted.

73.13.19 Error Injection Channel Descriptor 22, Word5 (EICHD22_WORD5)

Offset

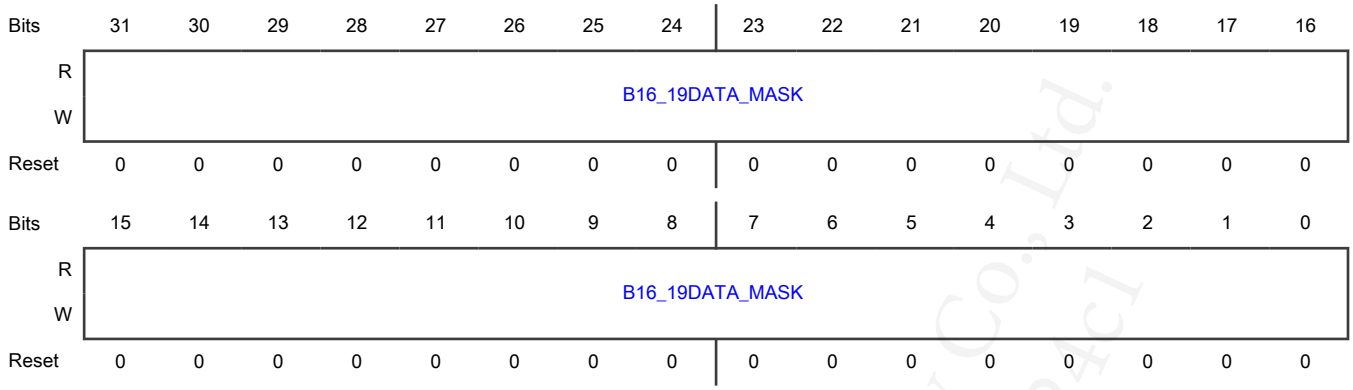
Register	Offset
EICHD22_WORD5	694h

Function

The sixth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B16_19DATA_MASK correspond to bytes 16–19 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing HHJ Automotive Technology Co., Ltd.
c8c46250-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B16_19DATA_MASK	Data Mask Bytes 16-19 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B16_19DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 16-19 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 16-19 on the read data bus is inverted.

73.13.20 Error Injection Channel Descriptor 22, Word6 (EICHD22_WORD6)

Offset

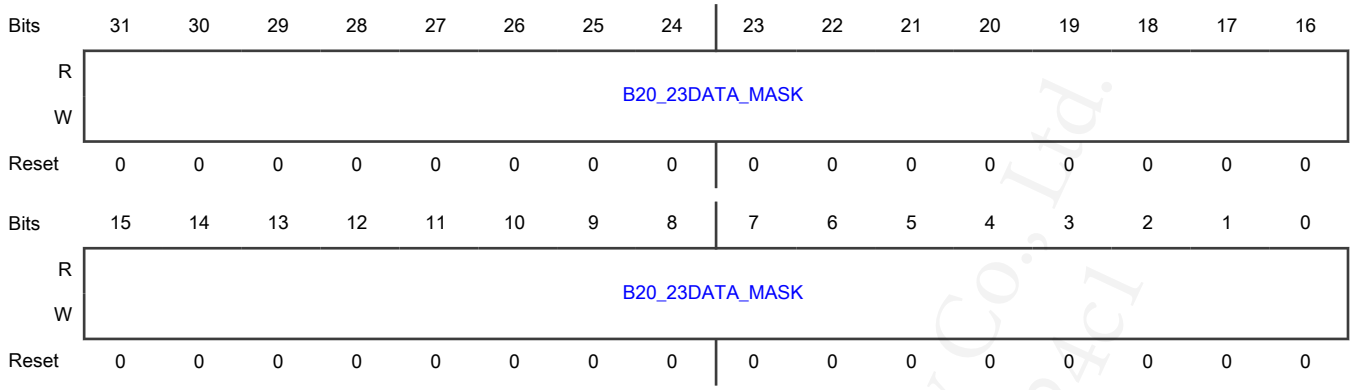
Register	Offset
EICHD22_WORD6	698h

Function

The seventh word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B20_23DATA_MASK correspond to bytes 20–23 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing HHJ Automotive Technology Co., Ltd.
c8c46205-4821-48a6-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B20_23DATA_MASK	Data Mask Bytes 20-23 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B20_23DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 20-23 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 20-23 on the read data bus is inverted.

73.13.21 Error Injection Channel Descriptor 22, Word7 (EICHD22_WORD7)

Offset

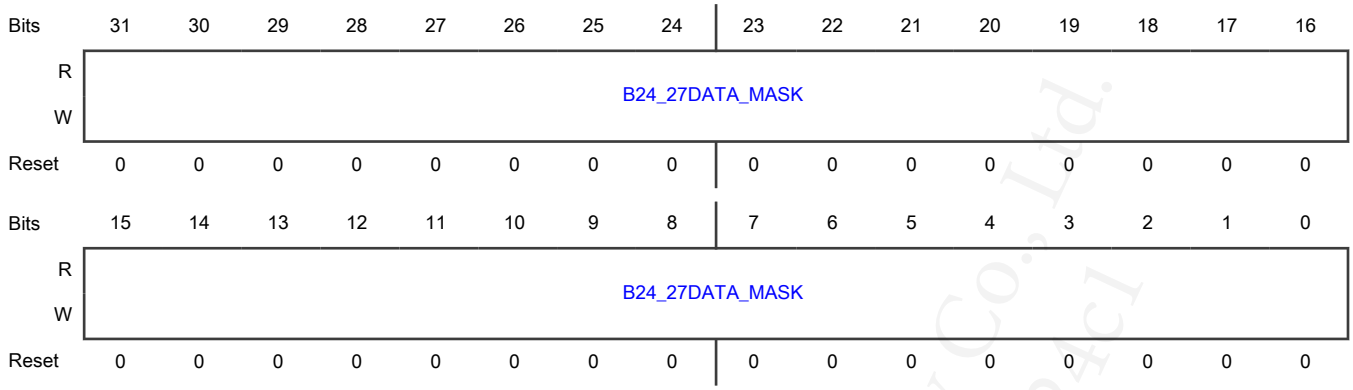
Register	Offset
EICHD22_WORD7	69Ch

Function

The eighth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B24_27DATA_MASK correspond to bytes 24–27 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing HHJ Automotive Technology Co., Ltd.
c8c46250-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B24_27DATA_MASK	Data Mask Bytes 24-27 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B24_27DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 24-27 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 24-27 on the read data bus is inverted.

73.13.22 Error Injection Channel Descriptor 22, Word8 (EICHD22_WORD8)

Offset

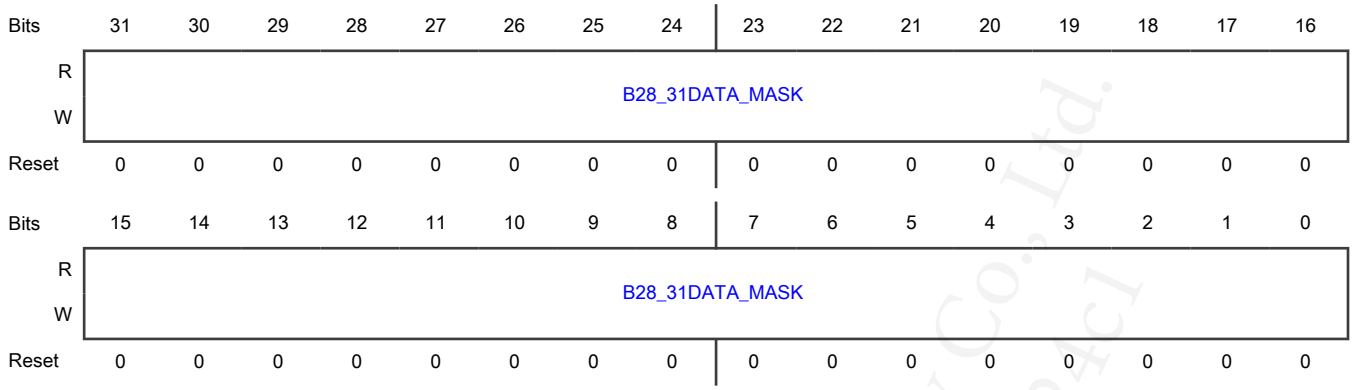
Register	Offset
EICHD22_WORD8	6A0h

Function

The ninth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B28_31DATA_MASK correspond to bytes 28–31 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46205-4821-48a6-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B28_31DATA_MASK	Data Mask Bytes 28-31 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B28_31DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 28-31 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 28-31 on the read data bus is inverted.

73.13.23 Error Injection Channel Descriptor n, Word0 (EICHD23_WORD0 - EICHD30_WORD0)

Offset

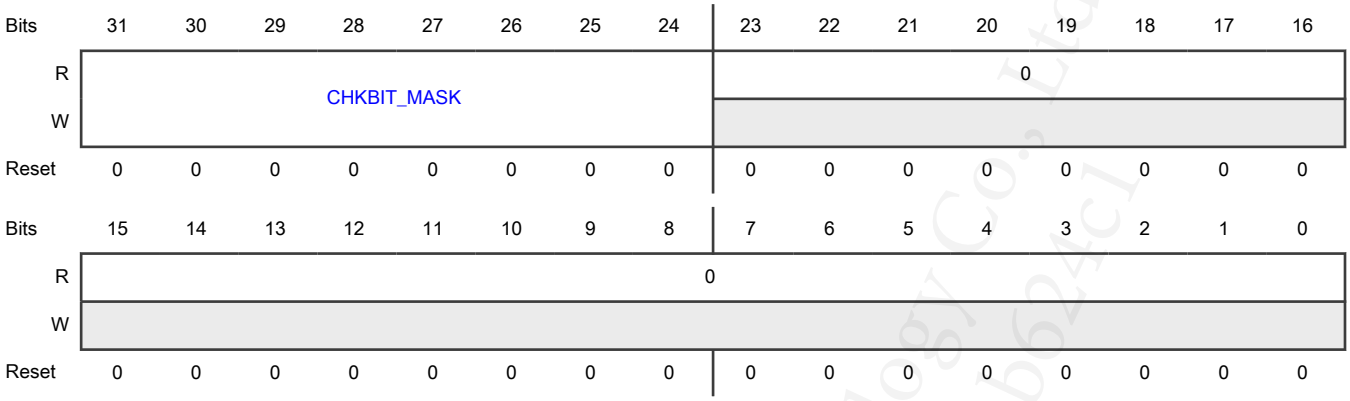
Register	Offset
EICHD23_WORD0	6C0h
EICHD24_WORD0	700h
EICHD25_WORD0	740h
EICHD26_WORD0	780h
EICHD27_WORD0	7C0h
EICHD28_WORD0	800h
EICHD29_WORD0	840h
EICHD30_WORD0	880h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or

remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.13.24 Error Injection Channel Descriptor 31, Word0 (EICHD31_WORD0)

Offset

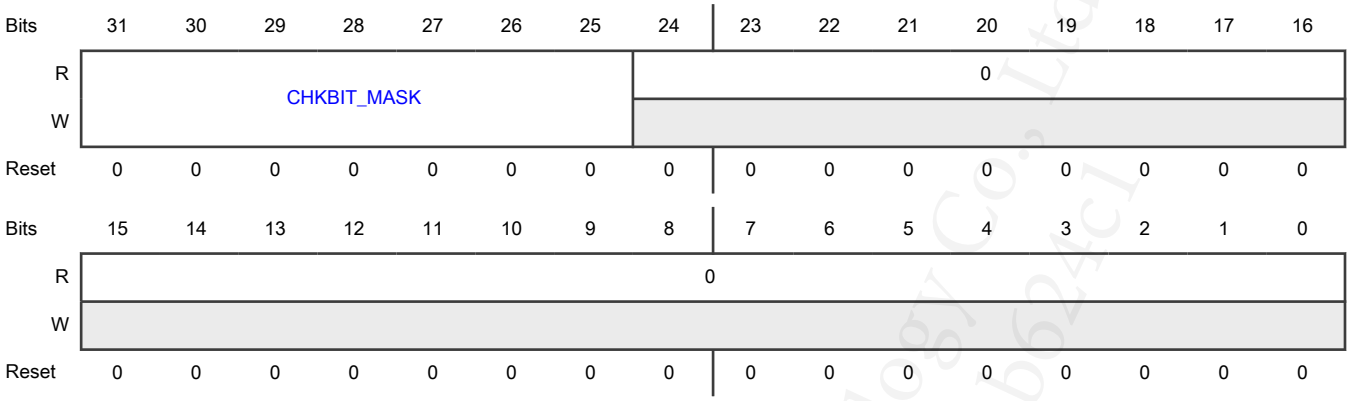
Register	Offset
EICHD31_WORD0	8C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or

remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.13.25 Error Injection Channel Descriptor 31, Word1 (EICHD31_WORD1)

Offset

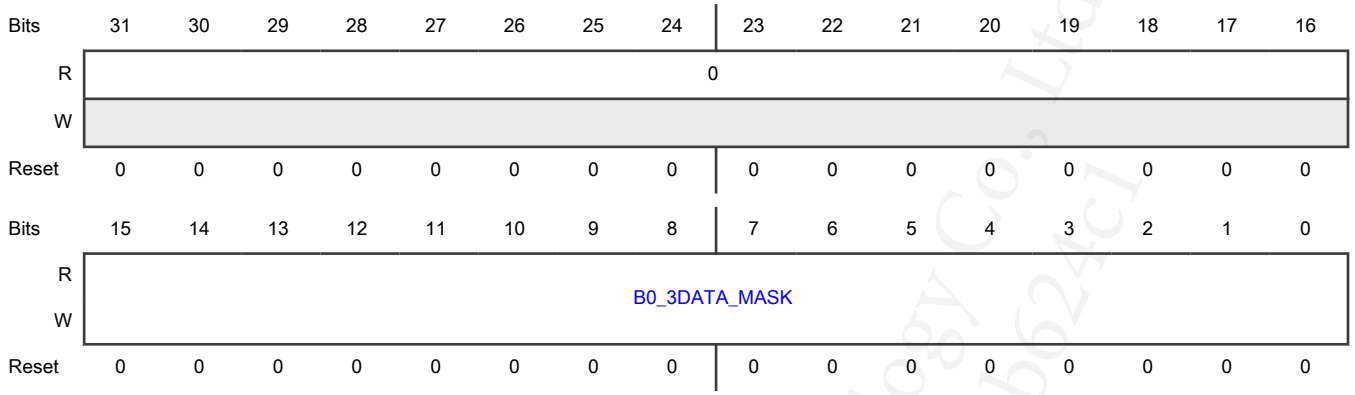
Register	Offset
EICHD31_WORD1	8C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target

RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.14 EIM_PFE3 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.
  - Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 595. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	256	255-224	223-192	191-160
1	64	63-32	31-0	—
2	64	63-32	31-0	—
3	64	63-32	31-0	—
4	64	63-32	31-0	—
5	64	63-32	31-0	—
6	64	63-32	31-0	—
7	64	63-32	31-0	—
8	64	63-32	31-0	—
9	16	15-0	—	—
10	256	255-224	223-192	191-160
11	64	63-32	31-0	—
12	64	63-32	31-0	—
13	64	63-32	31-0	—
14	64	63-32	31-0	—

Table continues on the next page...

Table 595. Error injection channel descriptor: DATA_MASK details (continued)

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
15	64	63-32	31-0	—
16	64	63-32	31-0	—
17	64	63-32	31-0	—
18	64	63-32	31-0	—
19	16	15-0	—	—
20	64	63-32	31-0	—
21	64	63-32	31-0	—
22	64	63-32	31-0	—
23	64	63-32	31-0	—
24	64	63-32	31-0	—
25	64	63-32	31-0	—
26	64	63-32	31-0	—
27	64	63-32	31-0	—
28	64	63-32	31-0	—
29	64	63-32	31-0	—
30	64	63-32	31-0	—
31	64	63-32	31-0	—

Table 596. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
0	159-128	127-96	95-64	63-32	31-0
10	159-128	127-96	95-64	63-32	31-0

73.14.1 EIM_PFE3 memory map

EIM_PFE3 base address: 4405_7000h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c46e2348214a8a1251a30d3ab624c1

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
100h	Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
10Ch	Error Injection Channel Descriptor 0, Word3 (EICHD0_WORD3)	32	RW	0000_0000h
110h	Error Injection Channel Descriptor 0, Word4 (EICHD0_WORD4)	32	RW	0000_0000h
114h	Error Injection Channel Descriptor 0, Word5 (EICHD0_WORD5)	32	RW	0000_0000h
118h	Error Injection Channel Descriptor 0, Word6 (EICHD0_WORD6)	32	RW	0000_0000h
11Ch	Error Injection Channel Descriptor 0, Word7 (EICHD0_WORD7)	32	RW	0000_0000h
120h	Error Injection Channel Descriptor 0, Word8 (EICHD0_WORD8)	32	RW	0000_0000h
140h	Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
148h	Error Injection Channel Descriptor 1, Word2 (EICHD1_WORD2)	32	RW	0000_0000h
180h	Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
1C0h	Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
200h	Error Injection Channel Descriptor 4, Word0 (EICHD4_WORD0)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
208h	Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)	32	RW	0000_0000h
240h	Error Injection Channel Descriptor 5, Word0 (EICHD5_WORD0)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
248h	Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)	32	RW	0000_0000h
280h	Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0)	32	RW	0000_0000h
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h
288h	Error Injection Channel Descriptor 6, Word2 (EICHD6_WORD2)	32	RW	0000_0000h
2C0h	Error Injection Channel Descriptor 7, Word0 (EICHD7_WORD0)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
2C8h	Error Injection Channel Descriptor 7, Word2 (EICHD7_WORD2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
300h	Error Injection Channel Descriptor 8, Word0 (EICHD8_WORD0)	32	RW	0000_0000h
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
308h	Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)	32	RW	0000_0000h
340h	Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
380h	Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0)	32	RW	0000_0000h
384h	Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)	32	RW	0000_0000h
388h	Error Injection Channel Descriptor 10, Word2 (EICHD10_WORD2)	32	RW	0000_0000h
38Ch	Error Injection Channel Descriptor 10, Word3 (EICHD10_WORD3)	32	RW	0000_0000h
390h	Error Injection Channel Descriptor 10, Word4 (EICHD10_WORD4)	32	RW	0000_0000h
394h	Error Injection Channel Descriptor 10, Word5 (EICHD10_WORD5)	32	RW	0000_0000h
398h	Error Injection Channel Descriptor 10, Word6 (EICHD10_WORD6)	32	RW	0000_0000h
39Ch	Error Injection Channel Descriptor 10, Word7 (EICHD10_WORD7)	32	RW	0000_0000h
3A0h	Error Injection Channel Descriptor 10, Word8 (EICHD10_WORD8)	32	RW	0000_0000h
3C0h	Error Injection Channel Descriptor 11, Word0 (EICHD11_WORD0)	32	RW	0000_0000h
3C4h	Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)	32	RW	0000_0000h
3C8h	Error Injection Channel Descriptor 11, Word2 (EICHD11_WORD2)	32	RW	0000_0000h
400h	Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0)	32	RW	0000_0000h
404h	Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)	32	RW	0000_0000h
408h	Error Injection Channel Descriptor 12, Word2 (EICHD12_WORD2)	32	RW	0000_0000h
440h	Error Injection Channel Descriptor 13, Word0 (EICHD13_WORD0)	32	RW	0000_0000h
444h	Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)	32	RW	0000_0000h
448h	Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)	32	RW	0000_0000h
480h	Error Injection Channel Descriptor 14, Word0 (EICHD14_WORD0)	32	RW	0000_0000h
484h	Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)	32	RW	0000_0000h
488h	Error Injection Channel Descriptor 14, Word2 (EICHD14_WORD2)	32	RW	0000_0000h
4C0h	Error Injection Channel Descriptor 15, Word0 (EICHD15_WORD0)	32	RW	0000_0000h
4C4h	Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)	32	RW	0000_0000h
4C8h	Error Injection Channel Descriptor 15, Word2 (EICHD15_WORD2)	32	RW	0000_0000h
500h	Error Injection Channel Descriptor 16, Word0 (EICHD16_WORD0)	32	RW	0000_0000h
504h	Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
508h	Error Injection Channel Descriptor 16, Word2 (EICHD16_WORD2)	32	RW	0000_0000h
540h	Error Injection Channel Descriptor 17, Word0 (EICHD17_WORD0)	32	RW	0000_0000h
544h	Error Injection Channel Descriptor 17, Word1 (EICHD17_WORD1)	32	RW	0000_0000h
548h	Error Injection Channel Descriptor 17, Word2 (EICHD17_WORD2)	32	RW	0000_0000h
580h	Error Injection Channel Descriptor 18, Word0 (EICHD18_WORD0)	32	RW	0000_0000h
584h	Error Injection Channel Descriptor 18, Word1 (EICHD18_WORD1)	32	RW	0000_0000h
588h	Error Injection Channel Descriptor 18, Word2 (EICHD18_WORD2)	32	RW	0000_0000h
5C0h	Error Injection Channel Descriptor 19, Word0 (EICHD19_WORD0)	32	RW	0000_0000h
5C4h	Error Injection Channel Descriptor 19, Word1 (EICHD19_WORD1)	32	RW	0000_0000h
600h	Error Injection Channel Descriptor 20, Word0 (EICHD20_WORD0)	32	RW	0000_0000h
604h	Error Injection Channel Descriptor 20, Word1 (EICHD20_WORD1)	32	RW	0000_0000h
608h	Error Injection Channel Descriptor 20, Word2 (EICHD20_WORD2)	32	RW	0000_0000h
640h	Error Injection Channel Descriptor 21, Word0 (EICHD21_WORD0)	32	RW	0000_0000h
644h	Error Injection Channel Descriptor 21, Word1 (EICHD21_WORD1)	32	RW	0000_0000h
648h	Error Injection Channel Descriptor 21, Word2 (EICHD21_WORD2)	32	RW	0000_0000h
680h	Error Injection Channel Descriptor 22, Word0 (EICHD22_WORD0)	32	RW	0000_0000h
684h	Error Injection Channel Descriptor 22, Word1 (EICHD22_WORD1)	32	RW	0000_0000h
688h	Error Injection Channel Descriptor 22, Word2 (EICHD22_WORD2)	32	RW	0000_0000h
6C0h	Error Injection Channel Descriptor 23, Word0 (EICHD23_WORD0)	32	RW	0000_0000h
6C4h	Error Injection Channel Descriptor 23, Word1 (EICHD23_WORD1)	32	RW	0000_0000h
6C8h	Error Injection Channel Descriptor 23, Word2 (EICHD23_WORD2)	32	RW	0000_0000h
700h	Error Injection Channel Descriptor 24, Word0 (EICHD24_WORD0)	32	RW	0000_0000h
704h	Error Injection Channel Descriptor 24, Word1 (EICHD24_WORD1)	32	RW	0000_0000h
708h	Error Injection Channel Descriptor 24, Word2 (EICHD24_WORD2)	32	RW	0000_0000h
740h	Error Injection Channel Descriptor 25, Word0 (EICHD25_WORD0)	32	RW	0000_0000h
744h	Error Injection Channel Descriptor 25, Word1 (EICHD25_WORD1)	32	RW	0000_0000h
748h	Error Injection Channel Descriptor 25, Word2 (EICHD25_WORD2)	32	RW	0000_0000h
780h	Error Injection Channel Descriptor 26, Word0 (EICHD26_WORD0)	32	RW	0000_0000h
784h	Error Injection Channel Descriptor 26, Word1 (EICHD26_WORD1)	32	RW	0000_0000h
788h	Error Injection Channel Descriptor 26, Word2 (EICHD26_WORD2)	32	RW	0000_0000h
7C0h	Error Injection Channel Descriptor 27, Word0 (EICHD27_WORD0)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
7C4h	Error Injection Channel Descriptor 27, Word1 (EICHD27_WORD1)	32	RW	0000_0000h
7C8h	Error Injection Channel Descriptor 27, Word2 (EICHD27_WORD2)	32	RW	0000_0000h
800h	Error Injection Channel Descriptor 28, Word0 (EICHD28_WORD0)	32	RW	0000_0000h
804h	Error Injection Channel Descriptor 28, Word1 (EICHD28_WORD1)	32	RW	0000_0000h
808h	Error Injection Channel Descriptor 28, Word2 (EICHD28_WORD2)	32	RW	0000_0000h
840h	Error Injection Channel Descriptor 29, Word0 (EICHD29_WORD0)	32	RW	0000_0000h
844h	Error Injection Channel Descriptor 29, Word1 (EICHD29_WORD1)	32	RW	0000_0000h
848h	Error Injection Channel Descriptor 29, Word2 (EICHD29_WORD2)	32	RW	0000_0000h
880h	Error Injection Channel Descriptor 30, Word0 (EICHD30_WORD0)	32	RW	0000_0000h
884h	Error Injection Channel Descriptor 30, Word1 (EICHD30_WORD1)	32	RW	0000_0000h
888h	Error Injection Channel Descriptor 30, Word2 (EICHD30_WORD2)	32	RW	0000_0000h
8C0h	Error Injection Channel Descriptor 31, Word0 (EICHD31_WORD0)	32	RW	0000_0000h
8C4h	Error Injection Channel Descriptor 31, Word1 (EICHD31_WORD1)	32	RW	0000_0000h
8C8h	Error Injection Channel Descriptor 31, Word2 (EICHD31_WORD2)	32	RW	0000_0000h

73.14.2 Error Injection Module Configuration Register (EIMCR)

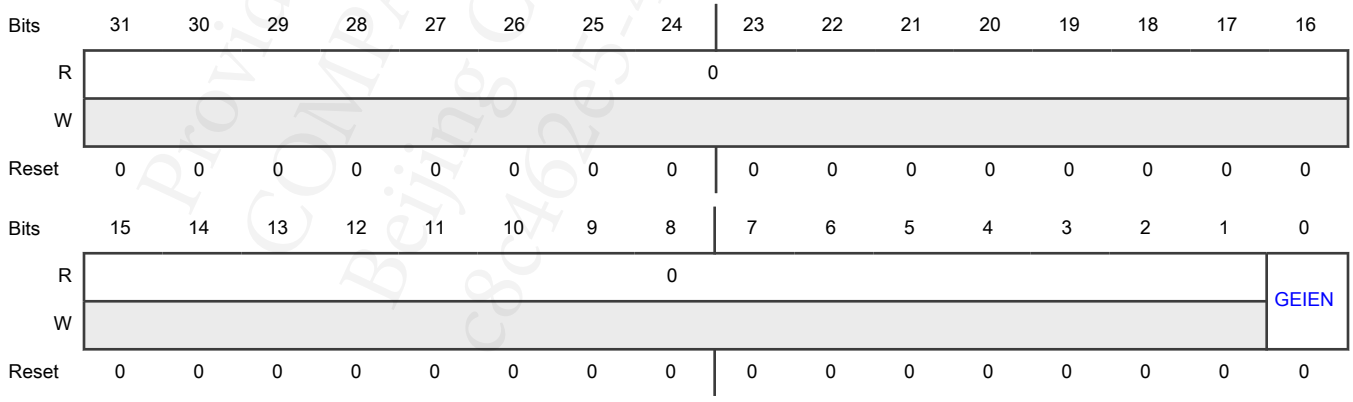
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

73.14.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EICH0EN	EICH1EN	EICH2EN	EICH3EN	EICH4EN	EICH5EN	EICH6EN	EICH7EN	EICH8EN	EICH9EN	EICH10EN	EICH11EN	EICH12EN	EICH13EN	EICH14EN	EICH15EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EICH16EN	EICH17EN	EICH18EN	EICH19EN	EICH20EN	EICH21EN	EICH22EN	EICH23EN	EICH24EN	EICH25EN	EICH26EN	EICH27EN	EICH28EN	EICH29EN	EICH30EN	EICH31EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c46254-8218a8-b251-390d3ab624c1

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9
21 EICH10EN	Error Injection Channel 10 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 10 1b - Error injection is enabled on Error Injection Channel 10

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 EICH11EN	Error Injection Channel 11 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 11 1b - Error injection is enabled on Error Injection Channel 11
19 EICH12EN	Error Injection Channel 12 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 12 1b - Error injection is enabled on Error Injection Channel 12
18 EICH13EN	Error Injection Channel 13 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 13 1b - Error injection is enabled on Error Injection Channel 13
17 EICH14EN	Error Injection Channel 14 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 14 1b - Error injection is enabled on Error Injection Channel 14
16 EICH15EN	Error Injection Channel 15 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 15 1b - Error injection is enabled on Error Injection Channel 15
15 EICH16EN	Error Injection Channel 16 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 16 1b - Error injection is enabled on Error Injection Channel 16
14 EICH17EN	Error Injection Channel 17 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 17 1b - Error injection is enabled on Error Injection Channel 17
13 EICH18EN	Error Injection Channel 18 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 18 1b - Error injection is enabled on Error Injection Channel 18
12 EICH19EN	Error Injection Channel 19 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 19 1b - Error injection is enabled on Error Injection Channel 19
11 EICH20EN	Error Injection Channel 20 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 20 1b - Error injection is enabled on Error Injection Channel 20
10 EICH21EN	Error Injection Channel 21 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 21 1b - Error injection is enabled on Error Injection Channel 21

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 EICH22EN	Error Injection Channel 22 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 22 1b - Error injection is enabled on Error Injection Channel 22
8 EICH23EN	Error Injection Channel 23 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 23 1b - Error injection is enabled on Error Injection Channel 23
7 EICH24EN	Error Injection Channel 24 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 24 1b - Error injection is enabled on Error Injection Channel 24
6 EICH25EN	Error Injection Channel 25 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 25 1b - Error injection is enabled on Error Injection Channel 25
5 EICH26EN	Error Injection Channel 26 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 26 1b - Error injection is enabled on Error Injection Channel 26
4 EICH27EN	Error Injection Channel 27 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 27 1b - Error injection is enabled on Error Injection Channel 27
3 EICH28EN	Error Injection Channel 28 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 28 1b - Error injection is enabled on Error Injection Channel 28
2 EICH29EN	Error Injection Channel 29 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 29 1b - Error injection is enabled on Error Injection Channel 29
1 EICH30EN	Error Injection Channel 30 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 30 1b - Error injection is enabled on Error Injection Channel 30
0 EICH31EN	Error Injection Channel 31 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 31 1b - Error injection is enabled on Error Injection Channel 31

73.14.4 Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)

Offset

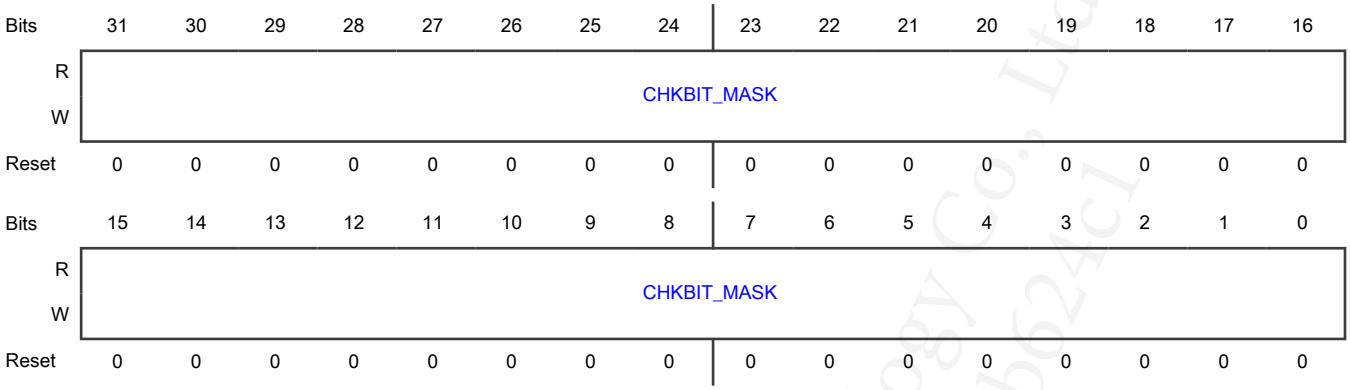
Register	Offset
EICHD0_WORD0	100h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or

remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[31:0] (32 bits wide), CHKBIT_MASK[31] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.

73.14.5 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD8_WORD1)

Offset

Register	Offset
EICHD0_WORD1	104h
EICHD1_WORD1	144h
EICHD2_WORD1	184h
EICHD3_WORD1	1C4h
EICHD4_WORD1	204h
EICHD5_WORD1	244h

Table continues on the next page...

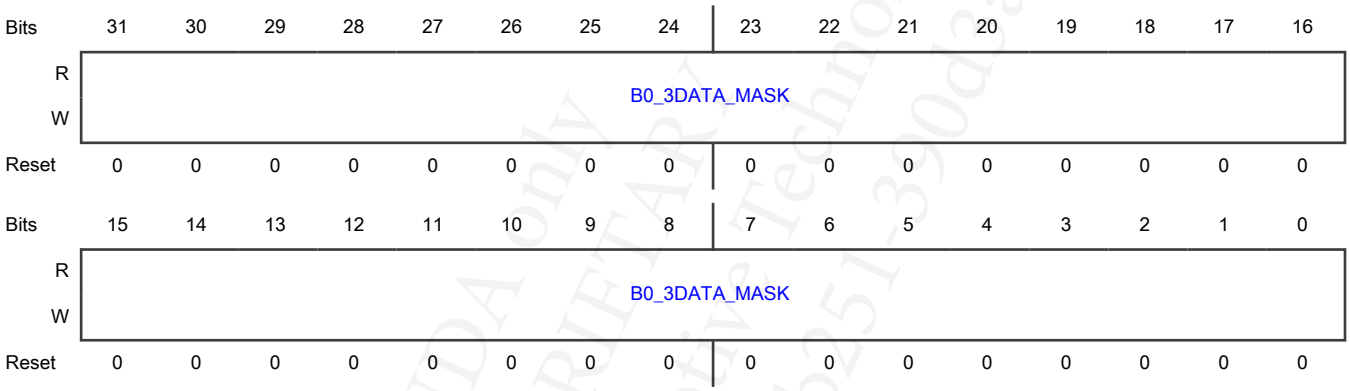
Table continued from the previous page...

Register	Offset
EICHD6_WORD1	284h
EICHD7_WORD1	2C4h
EICHD8_WORD1	304h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing BCAV Automotive Technology Co., Ltd.
c8c462e-1482-48a8-b251-390a3ab6fcc1

73.14.6 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD31_WORD2)

Offset

Register	Offset
EICHD0_WORD2	108h
EICHD1_WORD2	148h
EICHD2_WORD2	188h
EICHD3_WORD2	1C8h
EICHD4_WORD2	208h
EICHD5_WORD2	248h
EICHD6_WORD2	288h
EICHD7_WORD2	2C8h
EICHD8_WORD2	308h
EICHD10_WORD2	388h
EICHD11_WORD2	3C8h
EICHD12_WORD2	408h
EICHD13_WORD2	448h
EICHD14_WORD2	488h
EICHD15_WORD2	4C8h
EICHD16_WORD2	508h
EICHD17_WORD2	548h
EICHD18_WORD2	588h
EICHD20_WORD2	608h
EICHD21_WORD2	648h
EICHD22_WORD2	688h
EICHD23_WORD2	6C8h
EICHD24_WORD2	708h
EICHD25_WORD2	748h
EICHD26_WORD2	788h
EICHD27_WORD2	7C8h
EICHD28_WORD2	808h
EICHD29_WORD2	848h
EICHD30_WORD2	888h
EICHD31_WORD2	8C8h

Confidential under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4a8-b251-390d3ab624c1

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	B4_7DATA_MASK															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	B4_7DATA_MASK															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Bytes 4-7 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

73.14.7 Error Injection Channel Descriptor n, Word3 (EICHD0_WORD3 - EICHD10_WORD3)

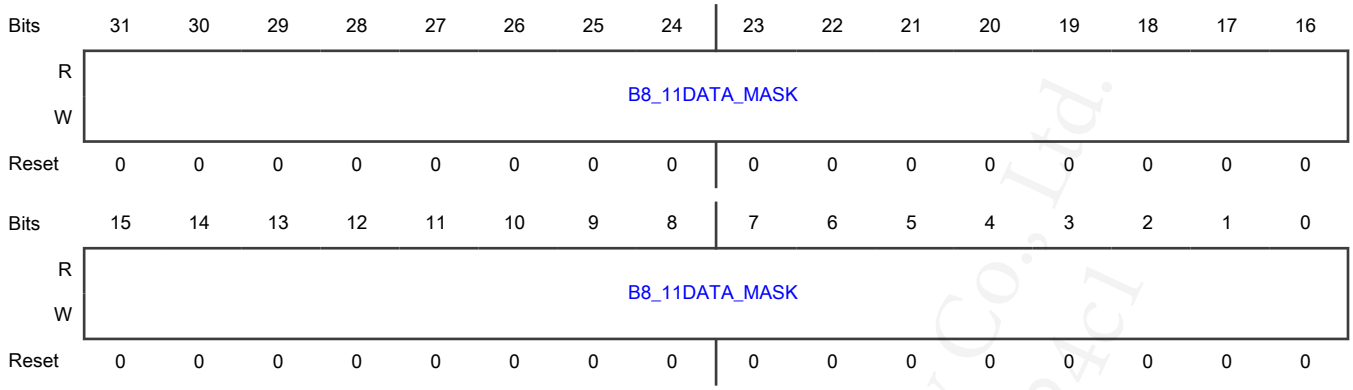
Offset

Register	Offset
EICHD0_WORD3	10Ch
EICHD10_WORD3	38Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to bytes 8–11 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Bytes 8-11 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 8-11 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 8-11 on the read data bus is inverted.

73.14.8 Error Injection Channel Descriptor n, Word4 (EICHD0_WORD4 - EICHD10_WORD4)

Offset

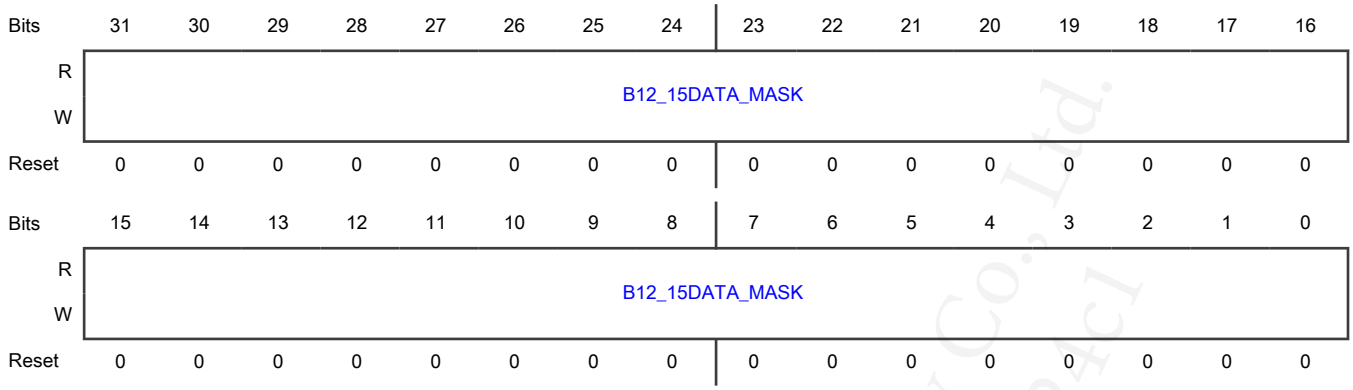
Register	Offset
EICHD0_WORD4	110h
EICHD10_WORD4	390h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to bytes 12–15 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co.,Ltd. c8c46a2e-4821-48a8-b2e1-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Bytes 12-15 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 12-15 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 12-15 on the read data bus is inverted.

73.14.9 Error Injection Channel Descriptor n, Word5 (EICHD0_WORD5 - EICHD10_WORD5)

Offset

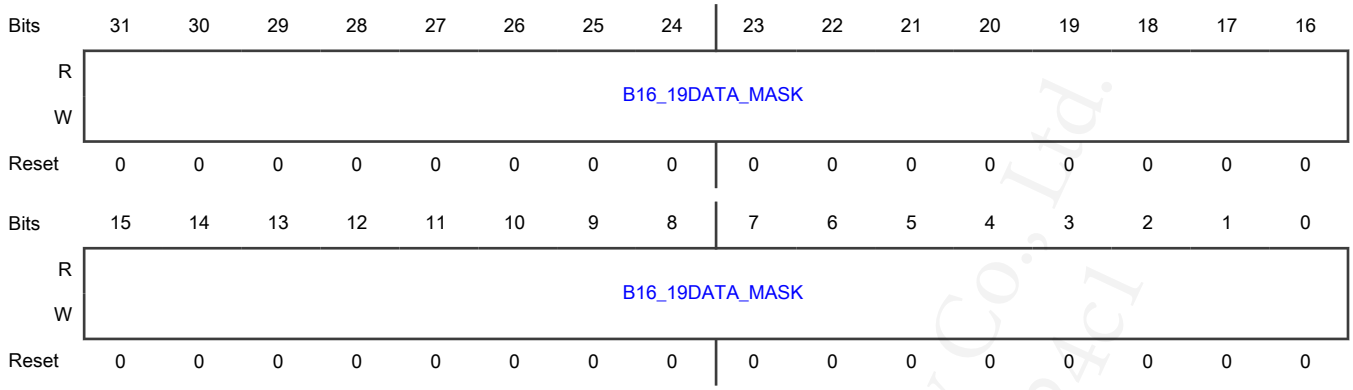
Register	Offset
EICHD0_WORD5	114h
EICHD10_WORD5	394h

Function

The sixth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B16_19DATA_MASK correspond to bytes 16–19 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c467c4-4821-48a8-b2f3-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B16_19DATA_MASK	Data Mask Bytes 16-19 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B16_19DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 16-19 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 16-19 on the read data bus is inverted.

73.14.10 Error Injection Channel Descriptor n, Word6 (EICHD0_WORD6 - EICHD10_WORD6)

Offset

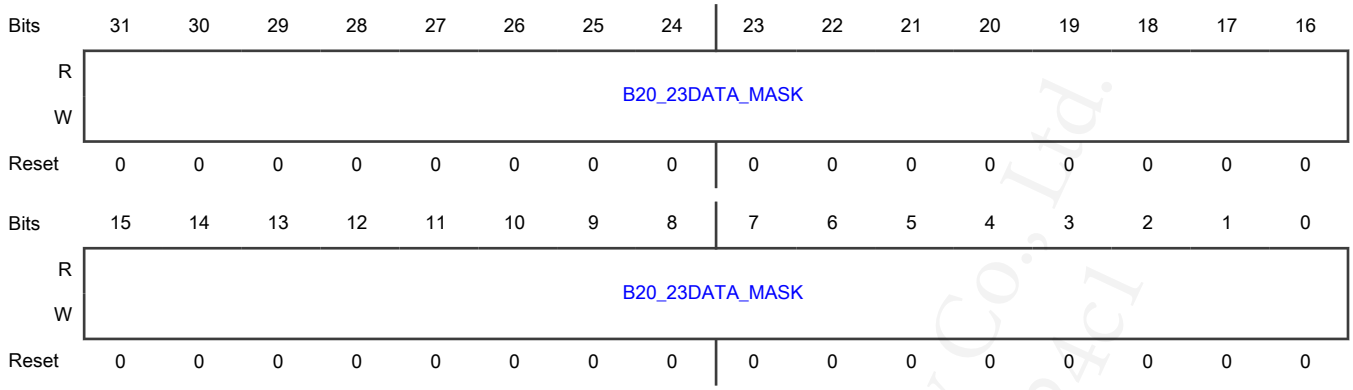
Register	Offset
EICHD0_WORD6	118h
EICHD10_WORD6	398h

Function

The seventh word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B20_23DATA_MASK correspond to bytes 20–23 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA COMPANY PROPRIETARY Beijing CHJ Automotive Technology Co.,Ltd. c8c467_4821-48a6-b2...390d3ab624c1

Diagram



Fields

Field	Function
31-0 B20_23DATA_MASK	Data Mask Bytes 20-23 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B20_23DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 20-23 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 20-23 on the read data bus is inverted.

73.14.11 Error Injection Channel Descriptor n, Word7 (EICHD0_WORD7 - EICHD10_WORD7)

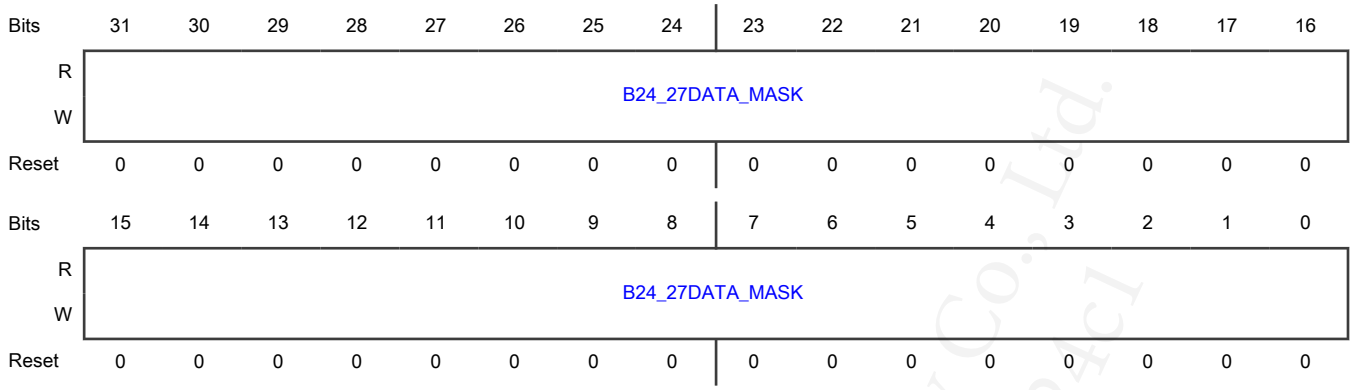
Offset

Register	Offset
EICHD0_WORD7	11Ch
EICHD10_WORD7	39Ch

Function

The eighth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B24_27DATA_MASK correspond to bytes 24–27 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B24_27DATA_MASK	Data Mask Bytes 24-27 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B24_27DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 24-27 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 24-27 on the read data bus is inverted.

73.14.12 Error Injection Channel Descriptor n, Word8 (EICHD0_WORD8 - EICHD10_WORD8)

Offset

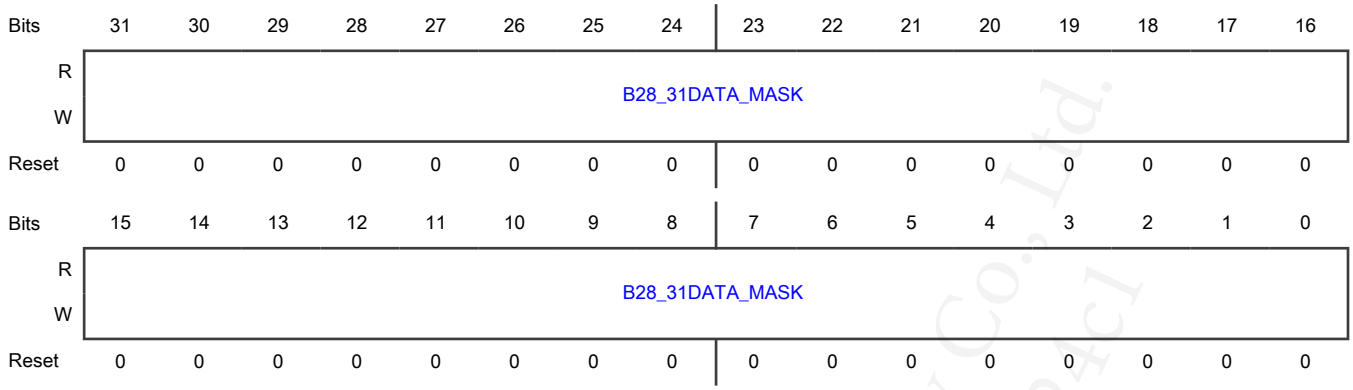
Register	Offset
EICHD0_WORD8	120h
EICHD10_WORD8	3A0h

Function

The ninth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B28_31DATA_MASK correspond to bytes 28–31 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c467c_4821-48a8-b2f3-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B28_31DATA_MASK	Data Mask Bytes 28-31 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B28_31DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 28-31 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 28-31 on the read data bus is inverted.

73.14.13 Error Injection Channel Descriptor n, Word0 (EICHD1_WORD0 - EICHD8_WORD0)

Offset

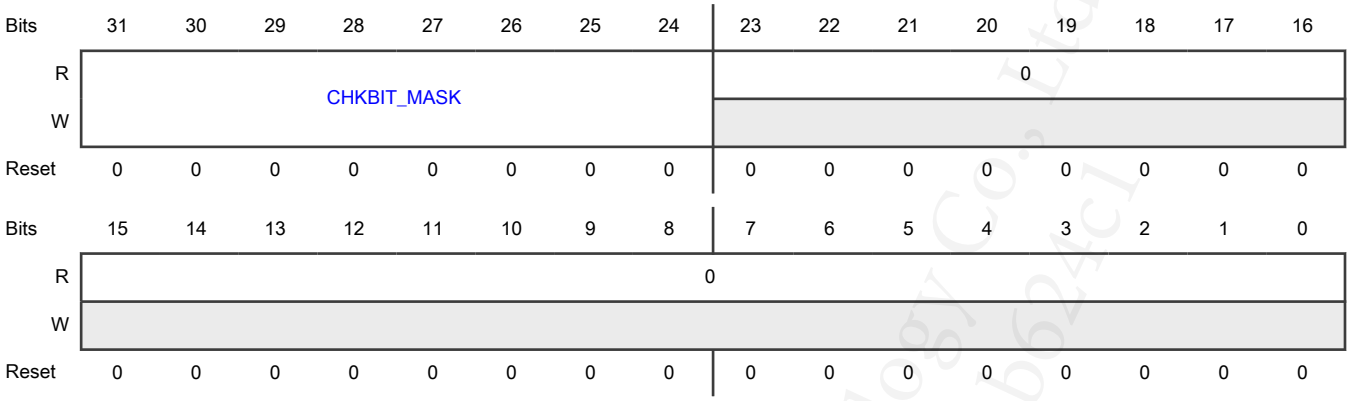
Register	Offset
EICHD1_WORD0	140h
EICHD2_WORD0	180h
EICHD3_WORD0	1C0h
EICHD4_WORD0	200h
EICHD5_WORD0	240h
EICHD6_WORD0	280h
EICHD7_WORD0	2C0h
EICHD8_WORD0	300h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or

remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.14.14 Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0)

Offset

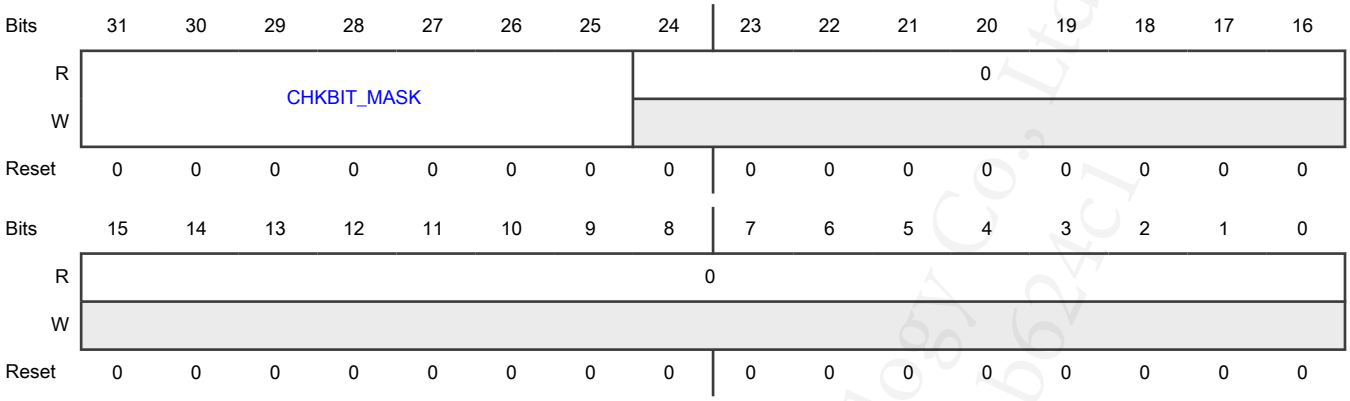
Register	Offset
EICHD9_WORD0	340h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or

remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.14.15 Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)

Offset

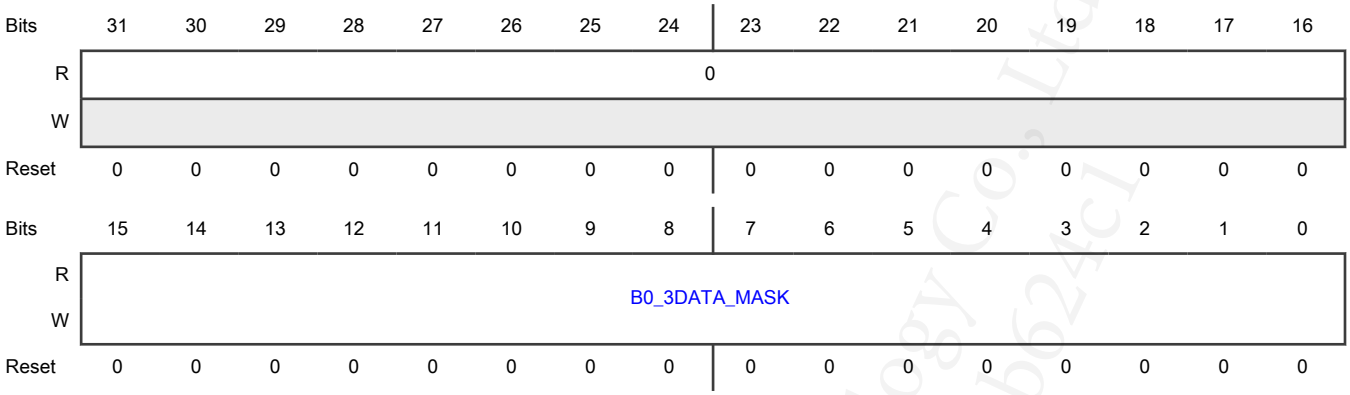
Register	Offset
EICHD9_WORD1	344h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target

RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.14.16 Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0)

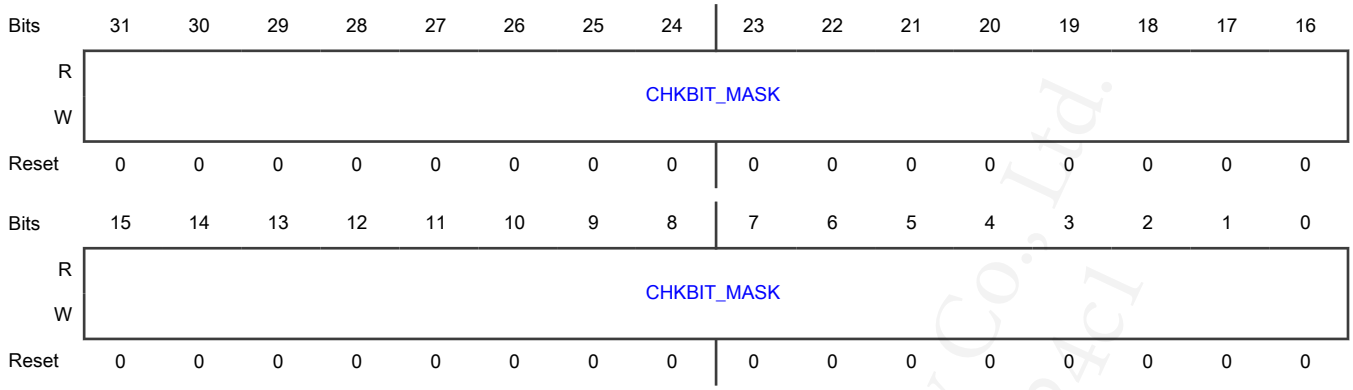
Offset

Register	Offset
EICHD10_WORD0	380h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[31:0] (32 bits wide), CHKBIT_MASK[31] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.

73.14.17 Error Injection Channel Descriptor n, Word1 (EICHD10_WORD1 - EICHD18_WORD1)

Offset

Register	Offset
EICHD10_WORD1	384h
EICHD11_WORD1	3C4h
EICHD12_WORD1	404h
EICHD13_WORD1	444h
EICHD14_WORD1	484h
EICHD15_WORD1	4C4h
EICHD16_WORD1	504h
EICHD17_WORD1	544h
EICHD18_WORD1	584h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B0_3DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.14.18 Error Injection Channel Descriptor n, Word0 (EICHD11_WORD0 - EICHD18_WORD0)

Offset

Register	Offset
EICHD11_WORD0	3C0h
EICHD12_WORD0	400h
EICHD13_WORD0	440h
EICHD14_WORD0	480h
EICHD15_WORD0	4C0h
EICHD16_WORD0	500h

Table continues on the next page...

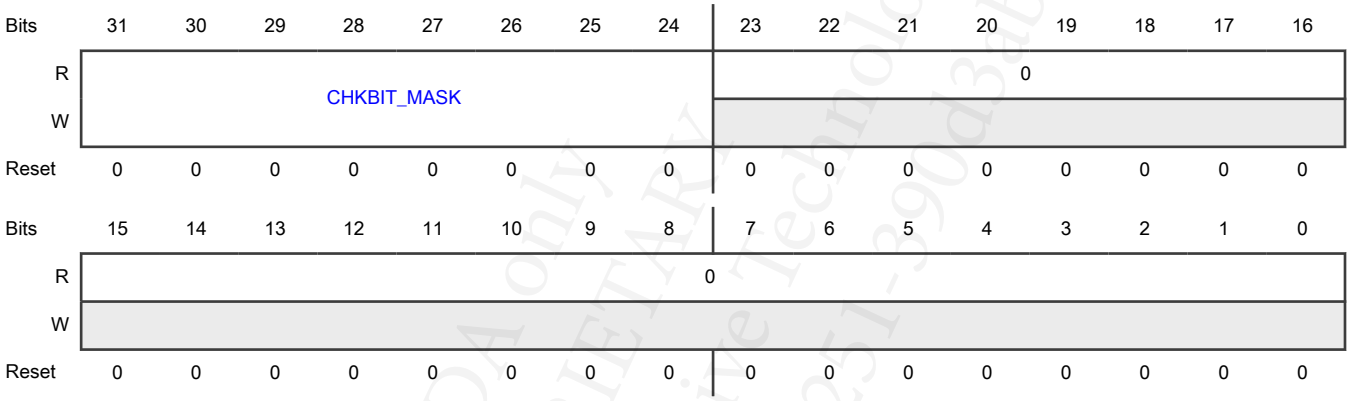
Table continued from the previous page...

Register	Offset
EICHD17_WORD0	540h
EICHD18_WORD0	580h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
8cA2e4824-1848-b251-590a3ab623

73.14.19 Error Injection Channel Descriptor 19, Word0 (EICHD19_WORD0)

Offset

Register	Offset
EICHD19_WORD0	5C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CHKBIT_MASK								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.14.20 Error Injection Channel Descriptor 19, Word1 (EICHD19_WORD1)

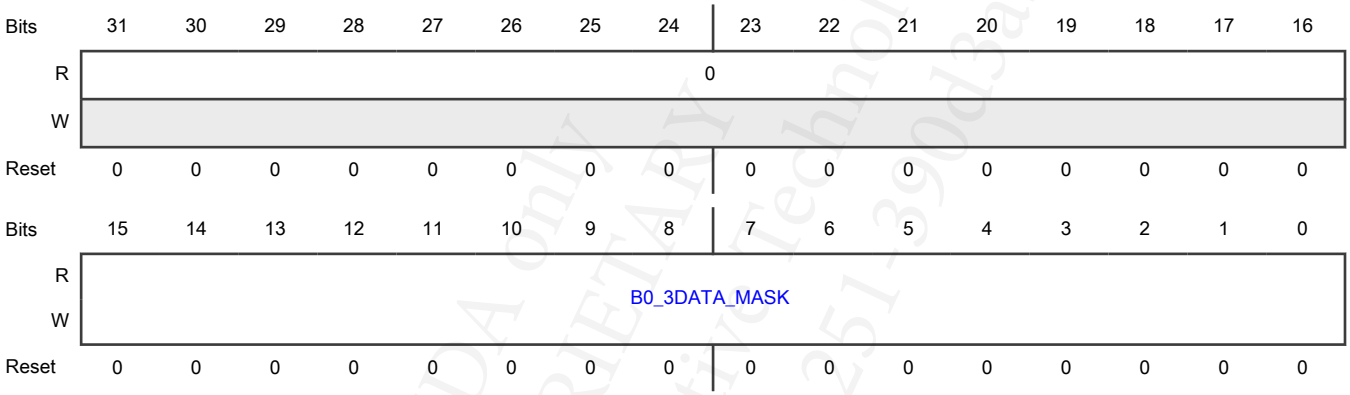
Offset

Register	Offset
EICHD19_WORD1	5C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing C&I Automotive Technology Co., Ltd.
c8a6e421-48a8-b251-390d3ab62e1

73.14.21 Error Injection Channel Descriptor n, Word0 (EICHD20_WORD0 - EICHD31_WORD0)

Offset

Register	Offset
EICHD20_WORD0	600h
EICHD21_WORD0	640h
EICHD22_WORD0	680h
EICHD23_WORD0	6C0h
EICHD24_WORD0	700h
EICHD25_WORD0	740h
EICHD26_WORD0	780h
EICHD27_WORD0	7C0h
EICHD28_WORD0	800h
EICHD29_WORD0	840h
EICHD30_WORD0	880h
EICHD31_WORD0	8C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CHKBIT_MASK								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-24	Checkbit Mask

Table continues on the next page...

Table continued from the previous page...

Field	Function
CHKBIT_MASK	This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.14.22 Error Injection Channel Descriptor n, Word1 (EICHD20_WORD1 - EICHD31_WORD1)

Offset

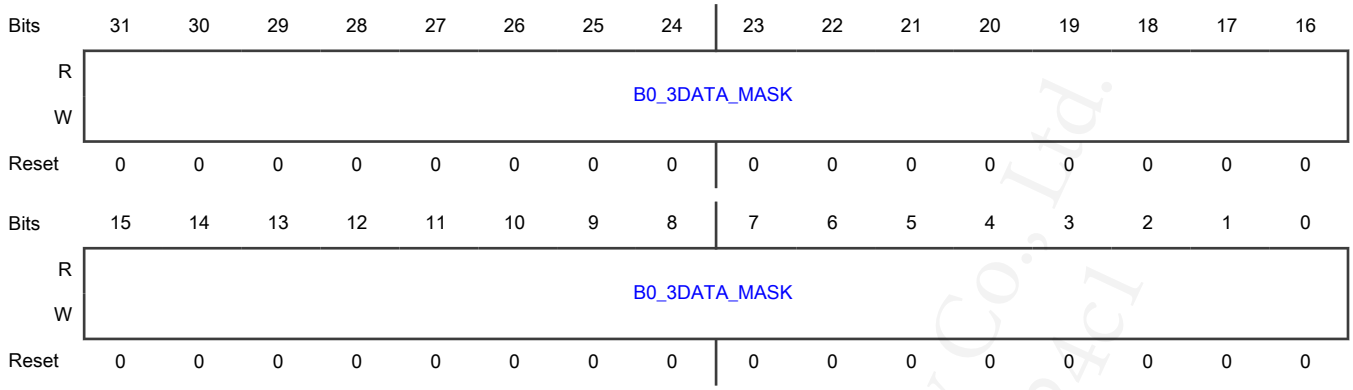
Register	Offset
EICHD20_WORD1	604h
EICHD21_WORD1	644h
EICHD22_WORD1	684h
EICHD23_WORD1	6C4h
EICHD24_WORD1	704h
EICHD25_WORD1	744h
EICHD26_WORD1	784h
EICHD27_WORD1	7C4h
EICHD28_WORD1	804h
EICHD29_WORD1	844h
EICHD30_WORD1	884h
EICHD31_WORD1	8C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
462e5-4821-48a8-b251-3a0d3ab624a1

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.15 EIM_PFE4 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 128-bit (16-byte) structure, composed of four 32-bit words, in the EIM programming model. Unused words are not documented.

— Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.

— Word1-3 (EICHD*n*_WORD1-3), if present, define the data mask. Word2 and Word3 are present only when required by the total width of the channel's data mask. See Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following table shows the total width of DATA_MASK and the distribution of its bits across the WORD registers.

Table 597. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	64	63-32	31-0	—
1	32	31-0	—	—
2	64	63-32	31-0	—
3	64	63-32	31-0	—
4	64	63-32	31-0	—
5	64	63-32	31-0	—
6	64	63-32	31-0	—
7	64	63-32	31-0	—
8	64	63-32	31-0	—
9	64	63-32	31-0	—

73.15.1 EIM_PFE4 memory map

EIM_PFE4 base address: 4405_8000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
100h	Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
108h	Error Injection Channel Descriptor 0, Word2 (EICHD0_WORD2)	32	RW	0000_0000h
140h	Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
180h	Error Injection Channel Descriptor 2, Word0 (EICHD2_WORD0)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
188h	Error Injection Channel Descriptor 2, Word2 (EICHD2_WORD2)	32	RW	0000_0000h
1C0h	Error Injection Channel Descriptor 3, Word0 (EICHD3_WORD0)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
1C8h	Error Injection Channel Descriptor 3, Word2 (EICHD3_WORD2)	32	RW	0000_0000h
200h	Error Injection Channel Descriptor 4, Word0 (EICHD4_WORD0)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
208h	Error Injection Channel Descriptor 4, Word2 (EICHD4_WORD2)	32	RW	0000_0000h
240h	Error Injection Channel Descriptor 5, Word0 (EICHD5_WORD0)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
248h	Error Injection Channel Descriptor 5, Word2 (EICHD5_WORD2)	32	RW	0000_0000h
280h	Error Injection Channel Descriptor 6, Word0 (EICHD6_WORD0)	32	RW	0000_0000h
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h
288h	Error Injection Channel Descriptor 6, Word2 (EICHD6_WORD2)	32	RW	0000_0000h
2C0h	Error Injection Channel Descriptor 7, Word0 (EICHD7_WORD0)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
2C8h	Error Injection Channel Descriptor 7, Word2 (EICHD7_WORD2)	32	RW	0000_0000h
300h	Error Injection Channel Descriptor 8, Word0 (EICHD8_WORD0)	32	RW	0000_0000h
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
308h	Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)	32	RW	0000_0000h
340h	Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
348h	Error Injection Channel Descriptor 9, Word2 (EICHD9_WORD2)	32	RW	0000_0000h

73.15.2 Error Injection Module Configuration Register (EIMCR)

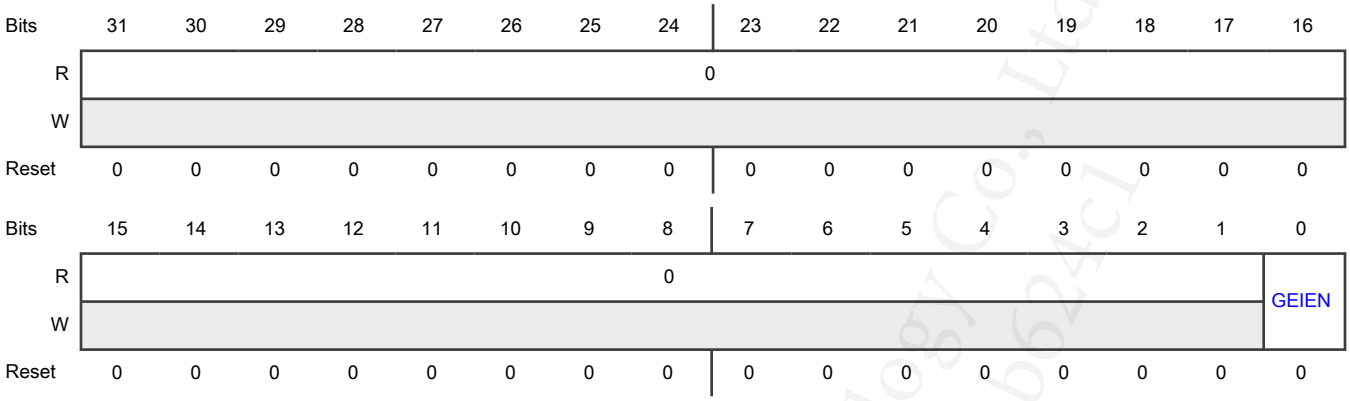
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

73.15.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-4a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0 EN	EICH1 EN	EICH2 EN	EICH3 EN	EICH4 EN	EICH5 EN	EICH6 EN	EICH7 EN	EICH8 EN	EICH9 EN	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9

Table continues on the next page...

Table continued from the previous page...

Field	Function
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13 —	Reserved
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continues on the next page...

Table continued from the previous page...

Field	Function
—	
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

73.15.4 Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)

Offset

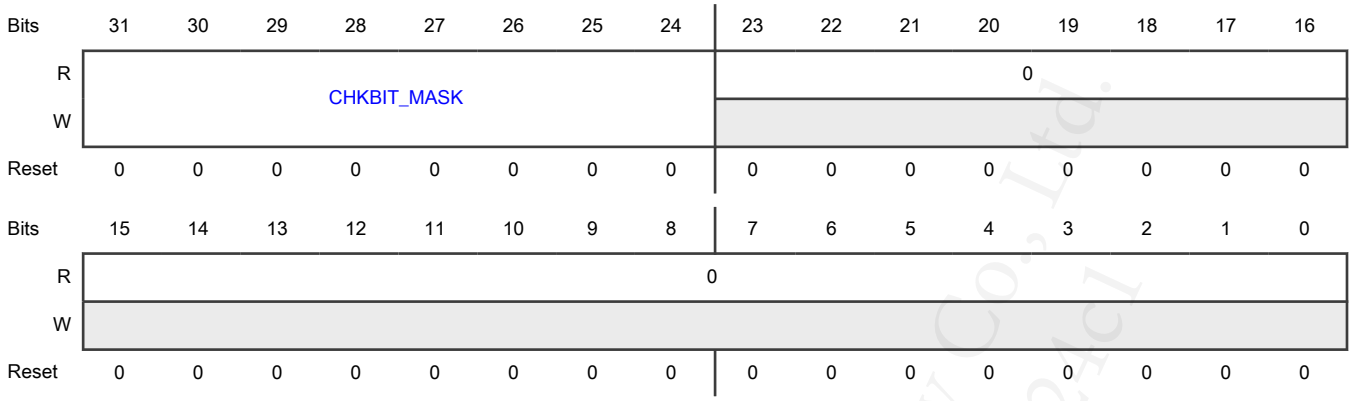
Register	Offset
EICHD0_WORD0	100h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e3-821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.15.5 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD9_WORD1)

Offset

Register	Offset
EICHD0_WORD1	104h
EICHD1_WORD1	144h
EICHD2_WORD1	184h
EICHD3_WORD1	1C4h
EICHD4_WORD1	204h

Table continues on the next page...

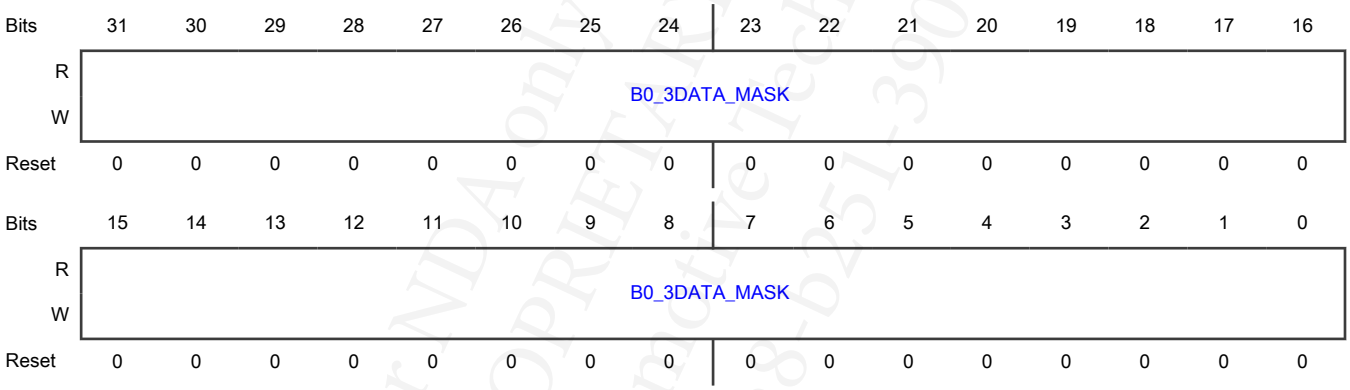
Table continued from the previous page...

Register	Offset
EICHD5_WORD1	244h
EICHD6_WORD1	284h
EICHD7_WORD1	2C4h
EICHD8_WORD1	304h
EICHD9_WORD1	344h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJ Automotive Technology Co., Ltd.
c8c8e2c3-421-48a8-b251-390d3a624c1

73.15.6 Error Injection Channel Descriptor n, Word2 (EICHD0_WORD2 - EICHD9_WORD2)

Offset

Register	Offset
EICHD0_WORD2	108h
EICHD2_WORD2	188h
EICHD3_WORD2	1C8h
EICHD4_WORD2	208h
EICHD5_WORD2	248h
EICHD6_WORD2	288h
EICHD7_WORD2	2C8h
EICHD8_WORD2	308h
EICHD9_WORD2	348h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	B4_7DATA_MASK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Bytes 4-7 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified.

Table continues on the next page...

Field	Function
	NOTE For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

73.15.7 Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)

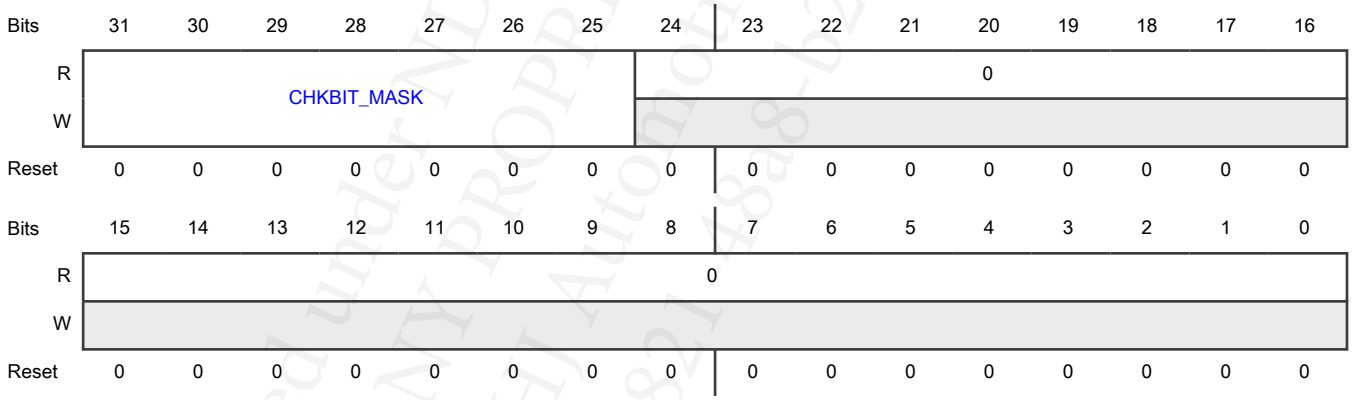
Offset

Register	Offset
EICHD1_WORD0	140h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.15.8 Error Injection Channel Descriptor n, Word0 (EICHD2_WORD0 - EICHD9_WORD0)

Offset

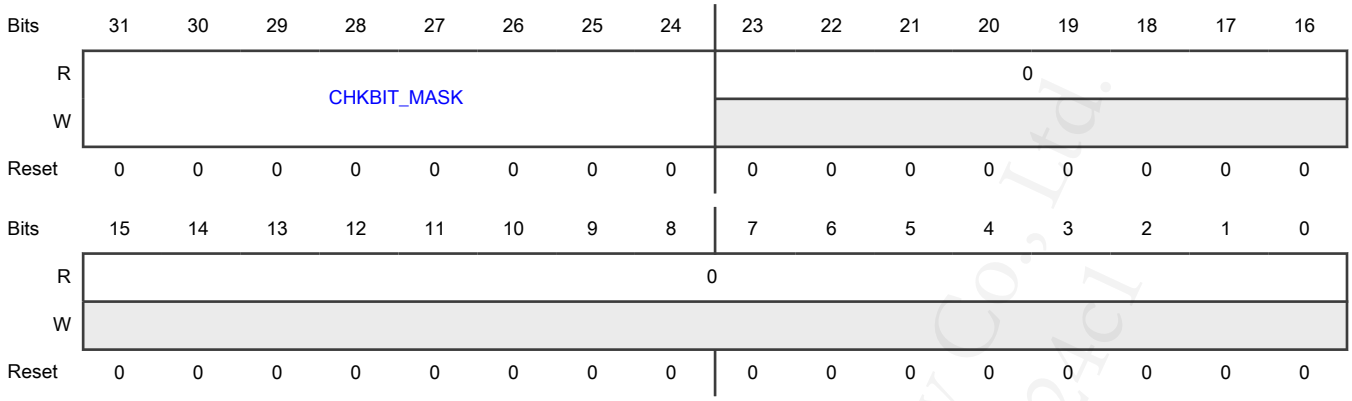
Register	Offset
EICHD2_WORD0	180h
EICHD3_WORD0	1C0h
EICHD4_WORD0	200h
EICHD5_WORD0	240h
EICHD6_WORD0	280h
EICHD7_WORD0	2C0h
EICHD8_WORD0	300h
EICHD9_WORD0	340h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing ESUN Automotive Technology Co., Ltd.
c8c462e5-821-48a8-b251-3903ab624c1

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.16 EIM_PFE5 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 288-bit (36-byte) structure, composed of nine 32-bit words, in the EIM programming model. Unused words are not documented.
  - Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.
  - Word1 (EICHD*n*_WORD1) and additional words, if present, define the data mask. Word registers subsequent to Word1 are present only when required by the total width of the channel's data mask. Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following tables show the distribution of DATA_MASK's bits across the WORD registers. The first table shows the total width of DATA_MASK and the distribution of its bits across WORD1, WORD2, and WORD3. The second table shows the distribution of DATA_MASK's bits across WORD4 and subsequent registers.

Table 598. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	16	15-0	—	—
1	16	15-0	—	—
2	8	7-0	—	—
3	8	7-0	—	—
4	16	15-0	—	—
5	8	7-0	—	—
6	8	7-0	—	—
7	16	15-0	—	—
8	64	63-32	31-0	—
9	64	63-32	31-0	—
10	64	63-32	31-0	—
11	64	63-32	31-0	—
12	64	63-32	31-0	—
13	64	63-32	31-0	—
14	32	31-0	—	—

Table continues on the next page...

Table 598. Error injection channel descriptor: DATA_MASK details (continued)

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
15	64	63-32	31-0	—
16	32	31-0	—	—
17	64	63-32	31-0	—
18	32	31-0	—	—
19	64	63-32	31-0	—
20	32	31-0	—	—
21	64	63-32	31-0	—
22	32	31-0	—	—
23	64	63-32	31-0	—
24	32	31-0	—	—
25	128	127-96	95-64	63-32
26	64	63-32	31-0	—
27	64	63-32	31-0	—
28	48	47-32	31-0	—
29	16	15-0	—	—
30	32	31-0	—	—
31	32	31-0	—	—

Table 599. DATA_MASK bit: Channel-word mapping

Channel	Specific bits of DATA_MASK in				
	WORD4	WORD5	WORD6	WORD7	WORD8
25	31-0	—	—	—	—

73.16.1 EIM_PFE5 memory map

EIM_PFE5 base address: 4405_9000h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c462e5-4821-4888-1251-890d3ab624c1

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h
184h	Error Injection Channel Descriptor 2, Word1 (EICHD2_WORD1)	32	RW	0000_0000h
1C4h	Error Injection Channel Descriptor 3, Word1 (EICHD3_WORD1)	32	RW	0000_0000h
204h	Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)	32	RW	0000_0000h
244h	Error Injection Channel Descriptor 5, Word1 (EICHD5_WORD1)	32	RW	0000_0000h
284h	Error Injection Channel Descriptor 6, Word1 (EICHD6_WORD1)	32	RW	0000_0000h
2C4h	Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)	32	RW	0000_0000h
300h	Error Injection Channel Descriptor 8, Word0 (EICHD8_WORD0)	32	RW	0000_0000h
304h	Error Injection Channel Descriptor 8, Word1 (EICHD8_WORD1)	32	RW	0000_0000h
308h	Error Injection Channel Descriptor 8, Word2 (EICHD8_WORD2)	32	RW	0000_0000h
340h	Error Injection Channel Descriptor 9, Word0 (EICHD9_WORD0)	32	RW	0000_0000h
344h	Error Injection Channel Descriptor 9, Word1 (EICHD9_WORD1)	32	RW	0000_0000h
348h	Error Injection Channel Descriptor 9, Word2 (EICHD9_WORD2)	32	RW	0000_0000h
380h	Error Injection Channel Descriptor 10, Word0 (EICHD10_WORD0)	32	RW	0000_0000h
384h	Error Injection Channel Descriptor 10, Word1 (EICHD10_WORD1)	32	RW	0000_0000h
388h	Error Injection Channel Descriptor 10, Word2 (EICHD10_WORD2)	32	RW	0000_0000h
3C0h	Error Injection Channel Descriptor 11, Word0 (EICHD11_WORD0)	32	RW	0000_0000h
3C4h	Error Injection Channel Descriptor 11, Word1 (EICHD11_WORD1)	32	RW	0000_0000h
3C8h	Error Injection Channel Descriptor 11, Word2 (EICHD11_WORD2)	32	RW	0000_0000h
400h	Error Injection Channel Descriptor 12, Word0 (EICHD12_WORD0)	32	RW	0000_0000h
404h	Error Injection Channel Descriptor 12, Word1 (EICHD12_WORD1)	32	RW	0000_0000h
408h	Error Injection Channel Descriptor 12, Word2 (EICHD12_WORD2)	32	RW	0000_0000h
440h	Error Injection Channel Descriptor 13, Word0 (EICHD13_WORD0)	32	RW	0000_0000h
444h	Error Injection Channel Descriptor 13, Word1 (EICHD13_WORD1)	32	RW	0000_0000h
448h	Error Injection Channel Descriptor 13, Word2 (EICHD13_WORD2)	32	RW	0000_0000h
480h	Error Injection Channel Descriptor 14, Word0 (EICHD14_WORD0)	32	RW	0000_0000h
484h	Error Injection Channel Descriptor 14, Word1 (EICHD14_WORD1)	32	RW	0000_0000h
4C0h	Error Injection Channel Descriptor 15, Word0 (EICHD15_WORD0)	32	RW	0000_0000h
4C4h	Error Injection Channel Descriptor 15, Word1 (EICHD15_WORD1)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
4C8h	Error Injection Channel Descriptor 15, Word2 (EICHD15_WORD2)	32	RW	0000_0000h
500h	Error Injection Channel Descriptor 16, Word0 (EICHD16_WORD0)	32	RW	0000_0000h
504h	Error Injection Channel Descriptor 16, Word1 (EICHD16_WORD1)	32	RW	0000_0000h
540h	Error Injection Channel Descriptor 17, Word0 (EICHD17_WORD0)	32	RW	0000_0000h
544h	Error Injection Channel Descriptor 17, Word1 (EICHD17_WORD1)	32	RW	0000_0000h
548h	Error Injection Channel Descriptor 17, Word2 (EICHD17_WORD2)	32	RW	0000_0000h
580h	Error Injection Channel Descriptor 18, Word0 (EICHD18_WORD0)	32	RW	0000_0000h
584h	Error Injection Channel Descriptor 18, Word1 (EICHD18_WORD1)	32	RW	0000_0000h
5C0h	Error Injection Channel Descriptor 19, Word0 (EICHD19_WORD0)	32	RW	0000_0000h
5C4h	Error Injection Channel Descriptor 19, Word1 (EICHD19_WORD1)	32	RW	0000_0000h
5C8h	Error Injection Channel Descriptor 19, Word2 (EICHD19_WORD2)	32	RW	0000_0000h
600h	Error Injection Channel Descriptor 20, Word0 (EICHD20_WORD0)	32	RW	0000_0000h
604h	Error Injection Channel Descriptor 20, Word1 (EICHD20_WORD1)	32	RW	0000_0000h
640h	Error Injection Channel Descriptor 21, Word0 (EICHD21_WORD0)	32	RW	0000_0000h
644h	Error Injection Channel Descriptor 21, Word1 (EICHD21_WORD1)	32	RW	0000_0000h
648h	Error Injection Channel Descriptor 21, Word2 (EICHD21_WORD2)	32	RW	0000_0000h
680h	Error Injection Channel Descriptor 22, Word0 (EICHD22_WORD0)	32	RW	0000_0000h
684h	Error Injection Channel Descriptor 22, Word1 (EICHD22_WORD1)	32	RW	0000_0000h
6C0h	Error Injection Channel Descriptor 23, Word0 (EICHD23_WORD0)	32	RW	0000_0000h
6C4h	Error Injection Channel Descriptor 23, Word1 (EICHD23_WORD1)	32	RW	0000_0000h
6C8h	Error Injection Channel Descriptor 23, Word2 (EICHD23_WORD2)	32	RW	0000_0000h
700h	Error Injection Channel Descriptor 24, Word0 (EICHD24_WORD0)	32	RW	0000_0000h
704h	Error Injection Channel Descriptor 24, Word1 (EICHD24_WORD1)	32	RW	0000_0000h
740h	Error Injection Channel Descriptor 25, Word0 (EICHD25_WORD0)	32	RW	0000_0000h
744h	Error Injection Channel Descriptor 25, Word1 (EICHD25_WORD1)	32	RW	0000_0000h
748h	Error Injection Channel Descriptor 25, Word2 (EICHD25_WORD2)	32	RW	0000_0000h
74Ch	Error Injection Channel Descriptor 25, Word3 (EICHD25_WORD3)	32	RW	0000_0000h
750h	Error Injection Channel Descriptor 25, Word4 (EICHD25_WORD4)	32	RW	0000_0000h
780h	Error Injection Channel Descriptor 26, Word0 (EICHD26_WORD0)	32	RW	0000_0000h
784h	Error Injection Channel Descriptor 26, Word1 (EICHD26_WORD1)	32	RW	0000_0000h
788h	Error Injection Channel Descriptor 26, Word2 (EICHD26_WORD2)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
7C0h	Error Injection Channel Descriptor 27, Word0 (EICHD27_WORD0)	32	RW	0000_0000h
7C4h	Error Injection Channel Descriptor 27, Word1 (EICHD27_WORD1)	32	RW	0000_0000h
7C8h	Error Injection Channel Descriptor 27, Word2 (EICHD27_WORD2)	32	RW	0000_0000h
800h	Error Injection Channel Descriptor 28, Word0 (EICHD28_WORD0)	32	RW	0000_0000h
804h	Error Injection Channel Descriptor 28, Word1 (EICHD28_WORD1)	32	RW	0000_0000h
808h	Error Injection Channel Descriptor 28, Word2 (EICHD28_WORD2)	32	RW	0000_0000h
840h	Error Injection Channel Descriptor 29, Word0 (EICHD29_WORD0)	32	RW	0000_0000h
844h	Error Injection Channel Descriptor 29, Word1 (EICHD29_WORD1)	32	RW	0000_0000h
880h	Error Injection Channel Descriptor 30, Word0 (EICHD30_WORD0)	32	RW	0000_0000h
884h	Error Injection Channel Descriptor 30, Word1 (EICHD30_WORD1)	32	RW	0000_0000h
8C0h	Error Injection Channel Descriptor 31, Word0 (EICHD31_WORD0)	32	RW	0000_0000h
8C4h	Error Injection Channel Descriptor 31, Word1 (EICHD31_WORD1)	32	RW	0000_0000h

73.16.2 Error Injection Module Configuration Register (EIMCR)

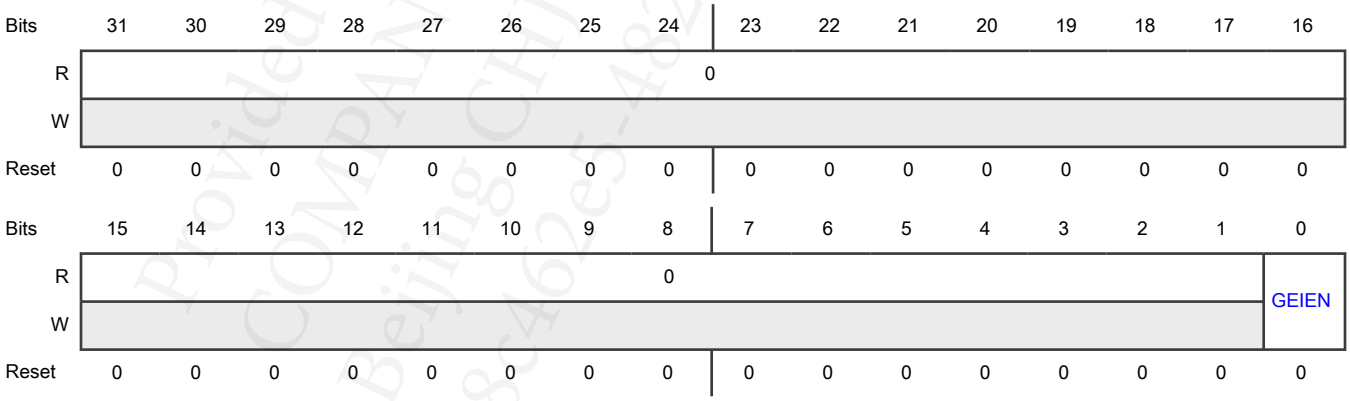
Offset

Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Diagram



Provided under NDA
COMPANY PROPRIETARY
Beijing CHI Automotive Technology Co.,Ltd.
c8462e5-4821-48a8-b251-5900cbb6a1a1

Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

73.16.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	EICH0EN	EICH1EN	EICH2EN	EICH3EN	EICH4EN	EICH5EN	EICH6EN	EICH7EN	EICH8EN	EICH9EN	EICH10EN	EICH11EN	EICH12EN	EICH13EN	EICH14EN	EICH15EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	EICH16EN	EICH17EN	EICH18EN	EICH19EN	EICH20EN	EICH21EN	EICH22EN	EICH23EN	EICH24EN	EICH25EN	EICH26EN	EICH27EN	EICH28EN	EICH29EN	EICH30EN	EICH31EN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHY Automotive Technology Co., Ltd.
c8c46245-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 EICH2EN	Error Injection Channel 2 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 2 1b - Error injection is enabled on Error Injection Channel 2
28 EICH3EN	Error Injection Channel 3 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 3 1b - Error injection is enabled on Error Injection Channel 3
27 EICH4EN	Error Injection Channel 4 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 4 1b - Error injection is enabled on Error Injection Channel 4
26 EICH5EN	Error Injection Channel 5 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 5 1b - Error injection is enabled on Error Injection Channel 5
25 EICH6EN	Error Injection Channel 6 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 6 1b - Error injection is enabled on Error Injection Channel 6
24 EICH7EN	Error Injection Channel 7 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 7 1b - Error injection is enabled on Error Injection Channel 7
23 EICH8EN	Error Injection Channel 8 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 8 1b - Error injection is enabled on Error Injection Channel 8
22 EICH9EN	Error Injection Channel 9 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 9 1b - Error injection is enabled on Error Injection Channel 9
21 EICH10EN	Error Injection Channel 10 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 10 1b - Error injection is enabled on Error Injection Channel 10

Table continues on the next page...

Table continued from the previous page...

Field	Function
20 EICH11EN	Error Injection Channel 11 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 11 1b - Error injection is enabled on Error Injection Channel 11
19 EICH12EN	Error Injection Channel 12 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 12 1b - Error injection is enabled on Error Injection Channel 12
18 EICH13EN	Error Injection Channel 13 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 13 1b - Error injection is enabled on Error Injection Channel 13
17 EICH14EN	Error Injection Channel 14 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 14 1b - Error injection is enabled on Error Injection Channel 14
16 EICH15EN	Error Injection Channel 15 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 15 1b - Error injection is enabled on Error Injection Channel 15
15 EICH16EN	Error Injection Channel 16 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 16 1b - Error injection is enabled on Error Injection Channel 16
14 EICH17EN	Error Injection Channel 17 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 17 1b - Error injection is enabled on Error Injection Channel 17
13 EICH18EN	Error Injection Channel 18 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 18 1b - Error injection is enabled on Error Injection Channel 18
12 EICH19EN	Error Injection Channel 19 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 19 1b - Error injection is enabled on Error Injection Channel 19
11 EICH20EN	Error Injection Channel 20 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 20 1b - Error injection is enabled on Error Injection Channel 20
10 EICH21EN	Error Injection Channel 21 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 21 1b - Error injection is enabled on Error Injection Channel 21

Table continues on the next page...

Table continued from the previous page...

Field	Function
9 EICH22EN	Error Injection Channel 22 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 22 1b - Error injection is enabled on Error Injection Channel 22
8 EICH23EN	Error Injection Channel 23 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 23 1b - Error injection is enabled on Error Injection Channel 23
7 EICH24EN	Error Injection Channel 24 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 24 1b - Error injection is enabled on Error Injection Channel 24
6 EICH25EN	Error Injection Channel 25 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 25 1b - Error injection is enabled on Error Injection Channel 25
5 EICH26EN	Error Injection Channel 26 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 26 1b - Error injection is enabled on Error Injection Channel 26
4 EICH27EN	Error Injection Channel 27 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 27 1b - Error injection is enabled on Error Injection Channel 27
3 EICH28EN	Error Injection Channel 28 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 28 1b - Error injection is enabled on Error Injection Channel 28
2 EICH29EN	Error Injection Channel 29 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 29 1b - Error injection is enabled on Error Injection Channel 29
1 EICH30EN	Error Injection Channel 30 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 30 1b - Error injection is enabled on Error Injection Channel 30
0 EICH31EN	Error Injection Channel 31 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 31 1b - Error injection is enabled on Error Injection Channel 31

73.16.4 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD1_WORD1)

Offset

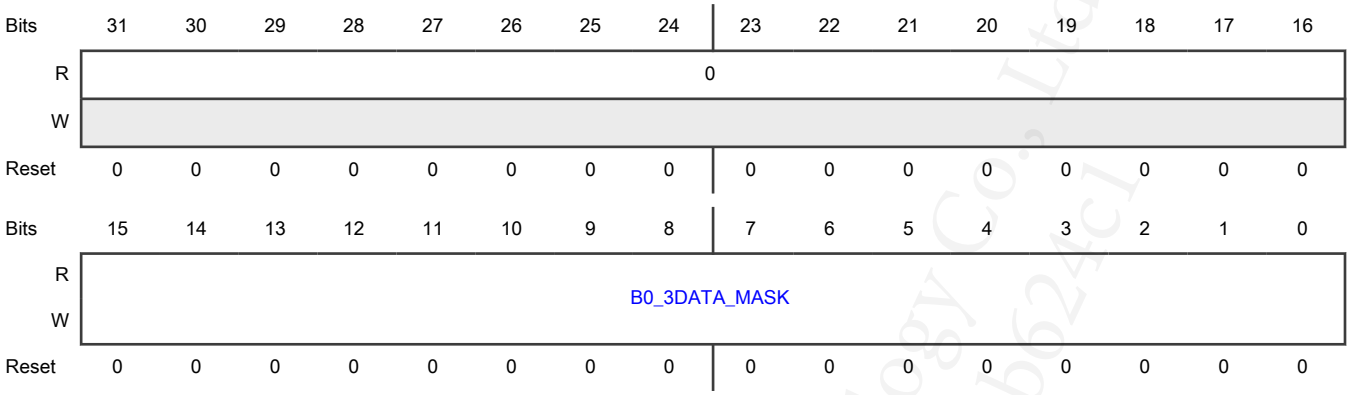
Register	Offset
EICHD0_WORD1	104h
EICHD1_WORD1	144h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target

RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.16.5 Error Injection Channel Descriptor n, Word1 (EICHD2_WORD1 - EICHD3_WORD1)

Offset

Register	Offset
EICHD2_WORD1	184h
EICHD3_WORD1	1C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								B0_3DATA_MASK							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.16.6 Error Injection Channel Descriptor 4, Word1 (EICHD4_WORD1)

Offset

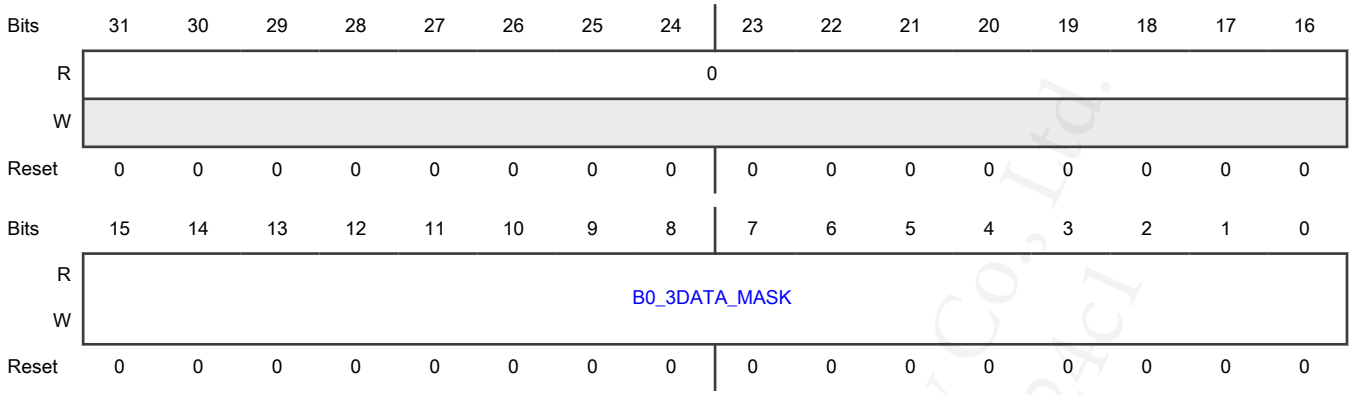
Register	Offset
EICHD4_WORD1	204h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-45a8-b251-390d3ab621c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.16.7 Error Injection Channel Descriptor n, Word1 (EICHD5_WORD1 - EICHD6_WORD1)

Offset

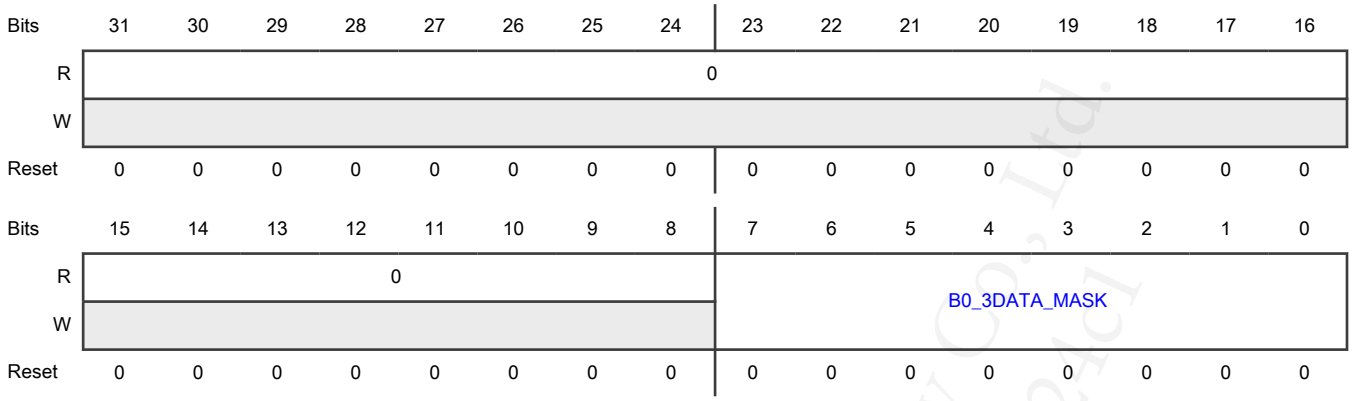
Register	Offset
EICHD5_WORD1	244h
EICHD6_WORD1	284h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c2e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-8 —	Reserved
7-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.16.8 Error Injection Channel Descriptor 7, Word1 (EICHD7_WORD1)

Offset

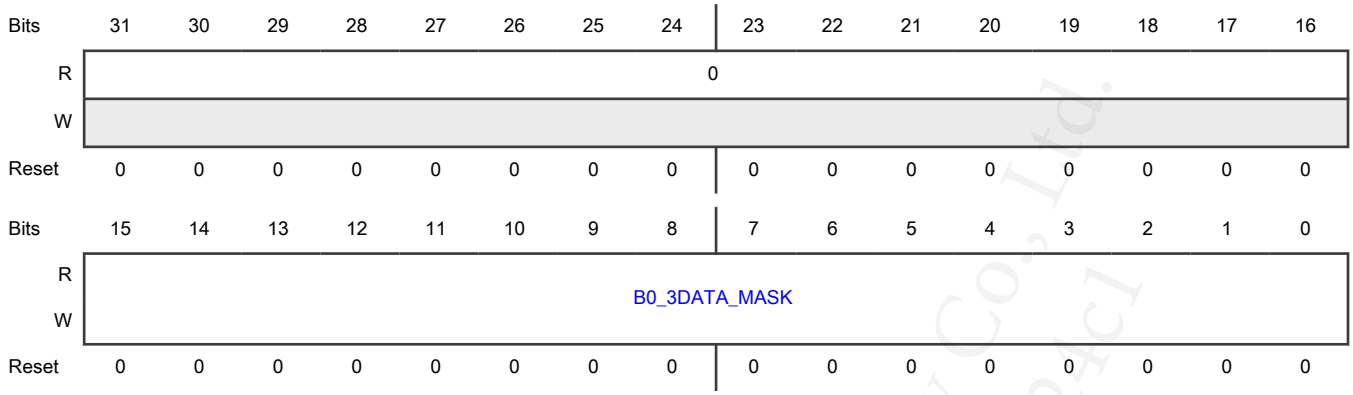
Register	Offset
EICHD7_WORD1	2C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-45a8-b251-390d3ab621c1

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.16.9 Error Injection Channel Descriptor n, Word0 (EICHD8_WORD0 - EICHD13_WORD0)

Offset

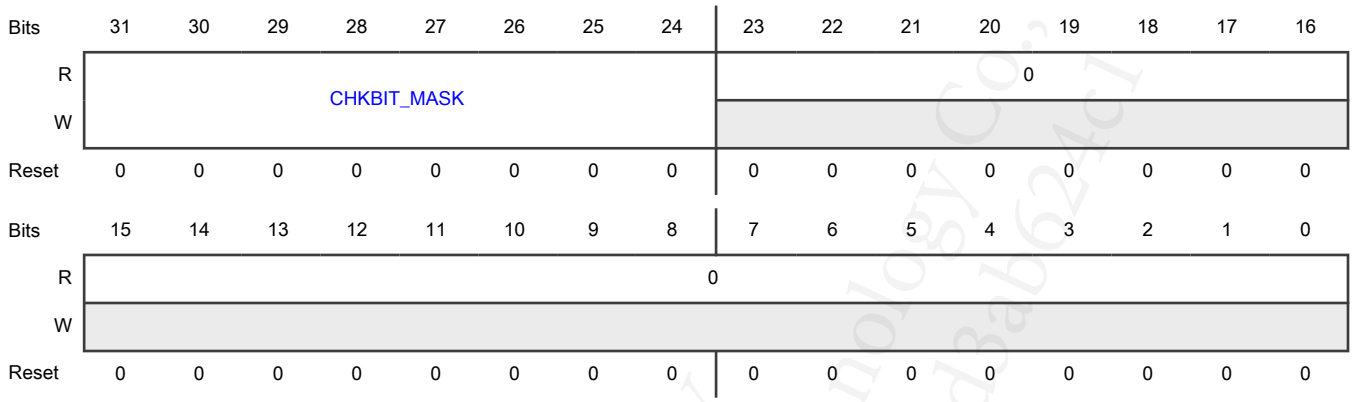
Register	Offset
EICHD8_WORD0	300h
EICHD9_WORD0	340h
EICHD10_WORD0	380h
EICHD11_WORD0	3C0h
EICHD12_WORD0	400h
EICHD13_WORD0	440h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-390d3ab624c1

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.16.10 Error Injection Channel Descriptor n, Word1 (EICHD8_WORD1 - EICHD27_WORD1)

Offset

Register	Offset
EICHD8_WORD1	304h

Table continues on the next page...

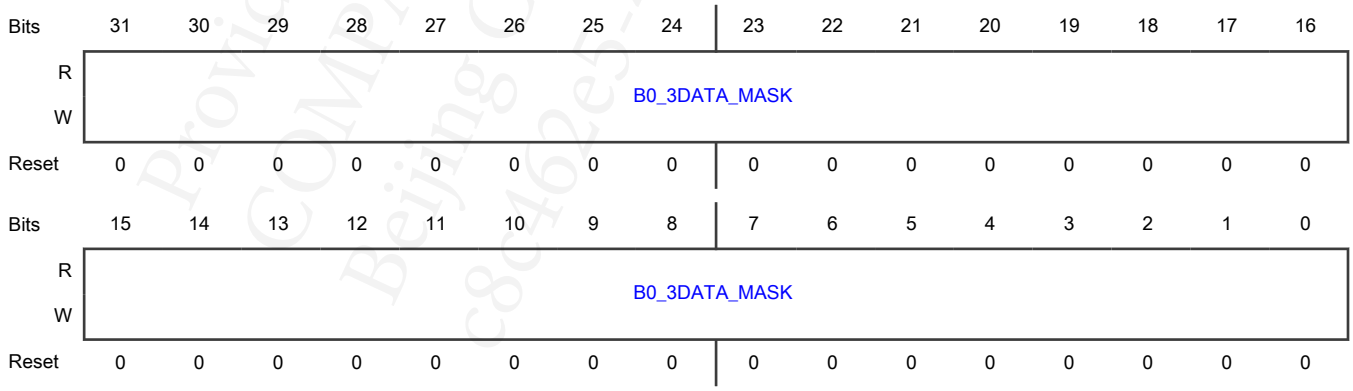
Table continued from the previous page...

Register	Offset
EICHD9_WORD1	344h
EICHD10_WORD1	384h
EICHD11_WORD1	3C4h
EICHD12_WORD1	404h
EICHD13_WORD1	444h
EICHD14_WORD1	484h
EICHD15_WORD1	4C4h
EICHD16_WORD1	504h
EICHD17_WORD1	544h
EICHD18_WORD1	584h
EICHD19_WORD1	5C4h
EICHD20_WORD1	604h
EICHD21_WORD1	644h
EICHD22_WORD1	684h
EICHD23_WORD1	6C4h
EICHD24_WORD1	704h
EICHD25_WORD1	744h
EICHD26_WORD1	784h
EICHD27_WORD1	7C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.16.11 Error Injection Channel Descriptor n, Word2 (EICHD8_WORD2 - EICHD28_WORD2)

Offset

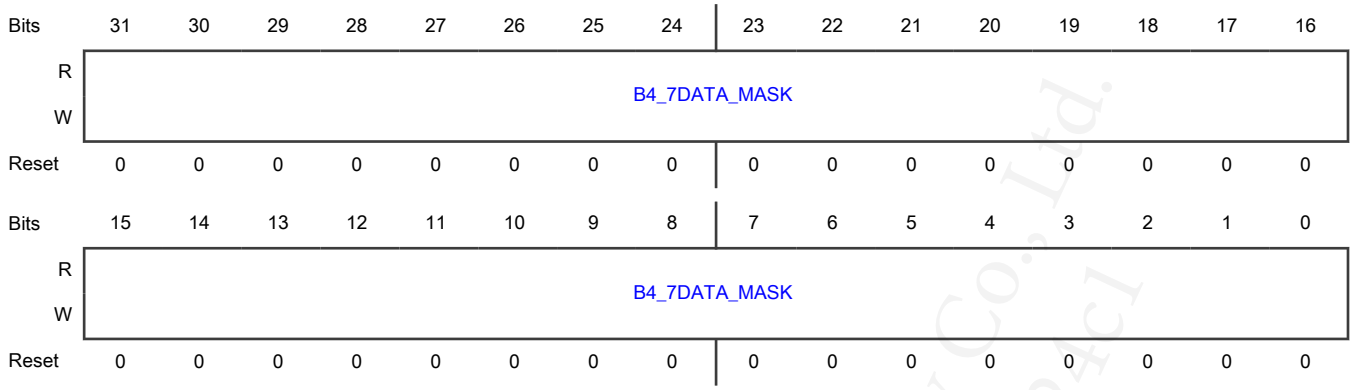
Register	Offset
EICHD8_WORD2	308h
EICHD9_WORD2	348h
EICHD10_WORD2	388h
EICHD11_WORD2	3C8h
EICHD12_WORD2	408h
EICHD13_WORD2	448h
EICHD15_WORD2	4C8h
EICHD17_WORD2	548h
EICHD19_WORD2	5C8h
EICHD21_WORD2	648h
EICHD23_WORD2	6C8h
EICHD25_WORD2	748h
EICHD26_WORD2	788h
EICHD27_WORD2	7C8h
EICHD28_WORD2	808h

Function

The third word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B4_7DATA_MASK correspond to bytes 4–7 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
d462e5-4821-48a8-b251-390d3b624c2

Diagram



Fields

Field	Function
31-0 B4_7DATA_MASK	Data Mask Bytes 4-7 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For each channel: For the specific DATA_MASK bits to which B4_7DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 4-7 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 4-7 on the read data bus is inverted.

73.16.12 Error Injection Channel Descriptor 14, Word0 (EICHD14_WORD0)

Offset

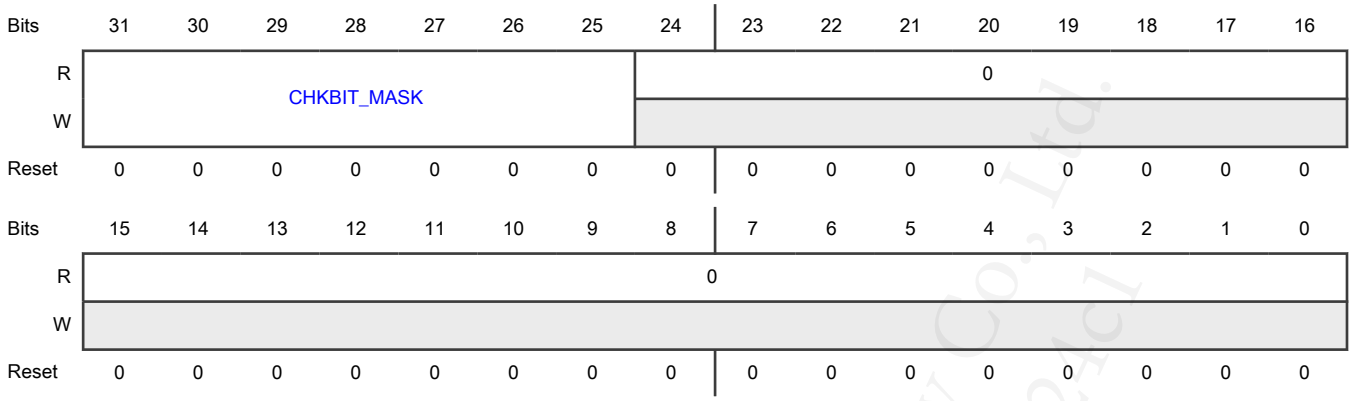
Register	Offset
EICHD14_WORD0	480h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing HHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b2f3-3903d3ab624c1

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.16.13 Error Injection Channel Descriptor 15, Word0 (EICHD15_WORD0)

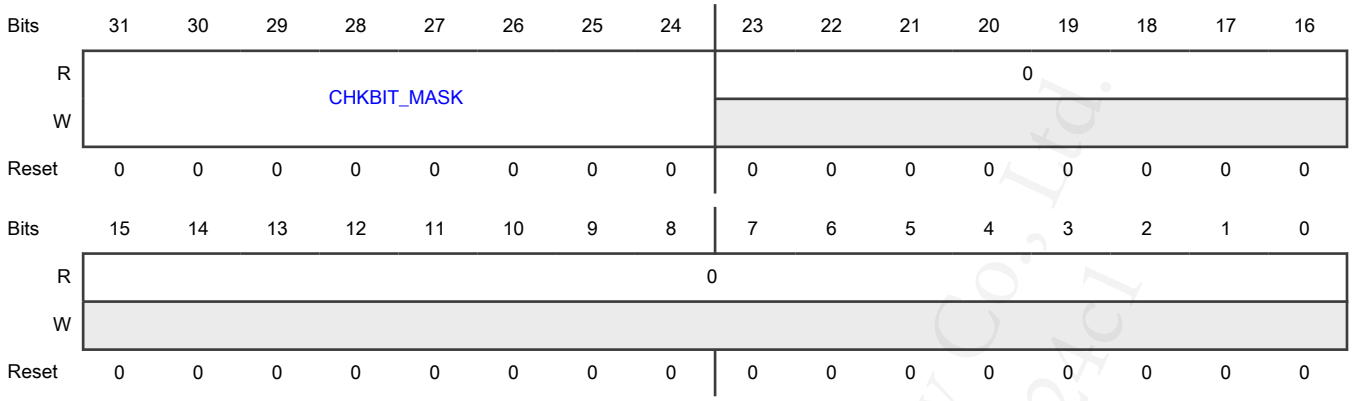
Offset

Register	Offset
EICHD15_WORD0	4C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.16.14 Error Injection Channel Descriptor 16, Word0 (EICHD16_WORD0)

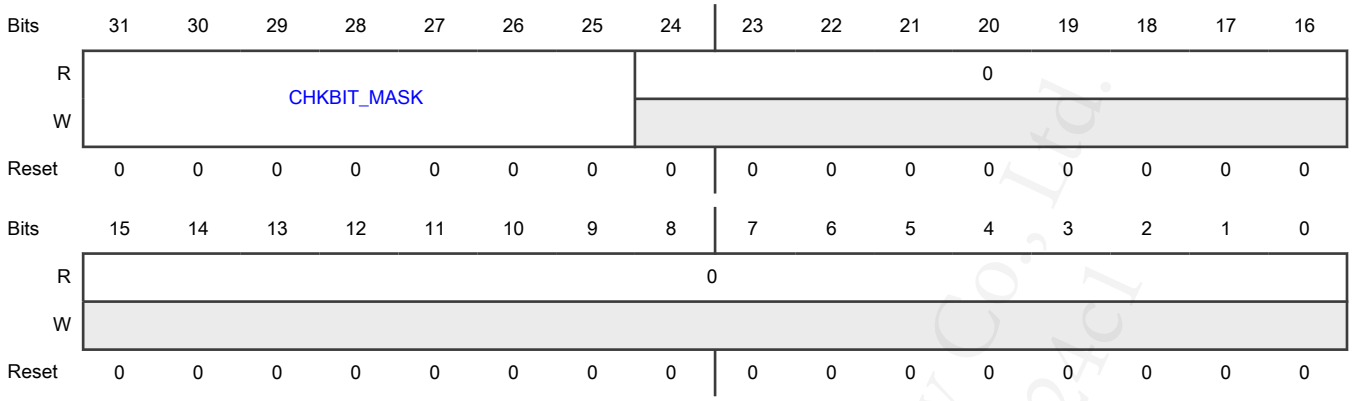
Offset

Register	Offset
EICHD16_WORD0	500h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.16.15 Error Injection Channel Descriptor 17, Word0 (EICHD17_WORD0)

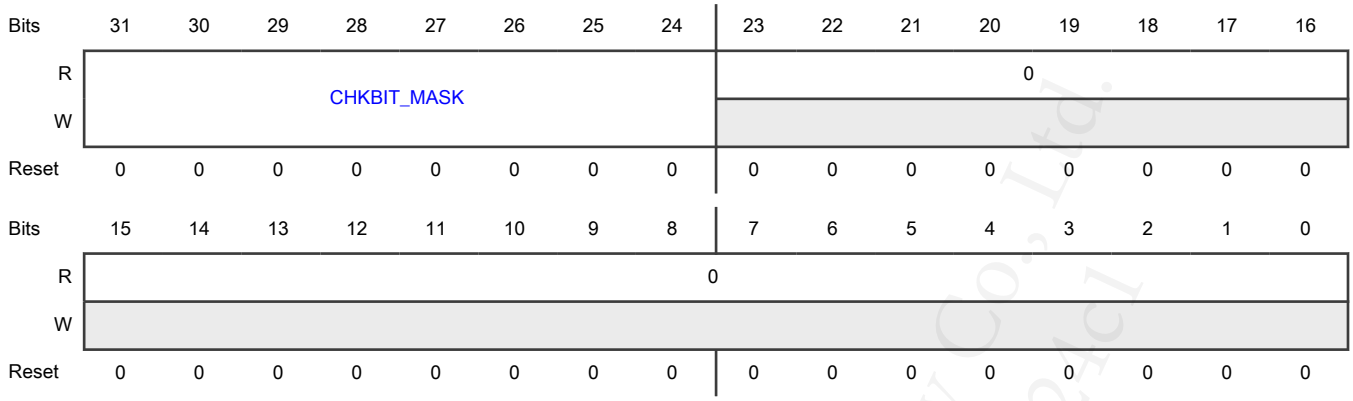
Offset

Register	Offset
EICHD17_WORD0	540h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.16.16 Error Injection Channel Descriptor 18, Word0 (EICHD18_WORD0)

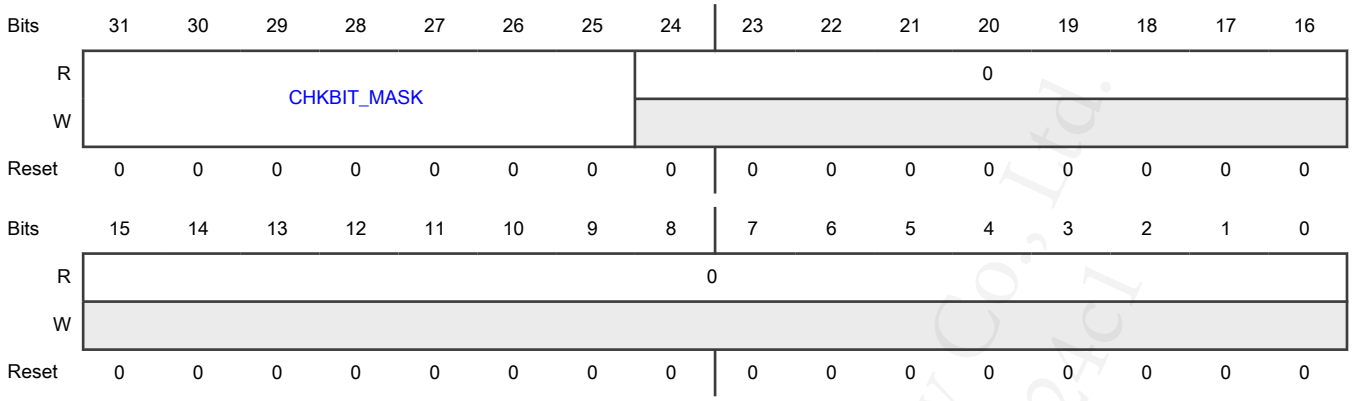
Offset

Register	Offset
EICHD18_WORD0	580h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.16.17 Error Injection Channel Descriptor 19, Word0 (EICHD19_WORD0)

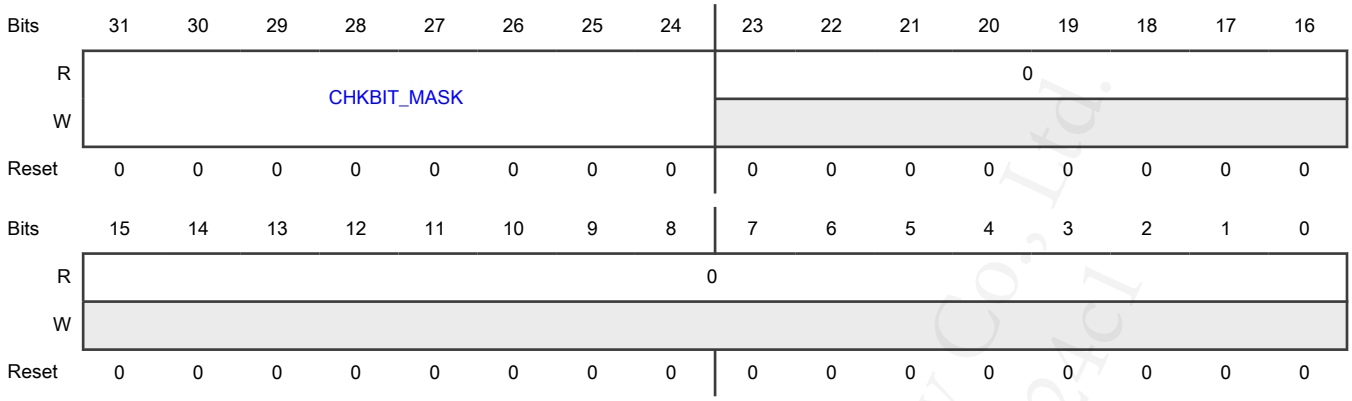
Offset

Register	Offset
EICHD19_WORD0	5C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.16.18 Error Injection Channel Descriptor 20, Word0 (EICHD20_WORD0)

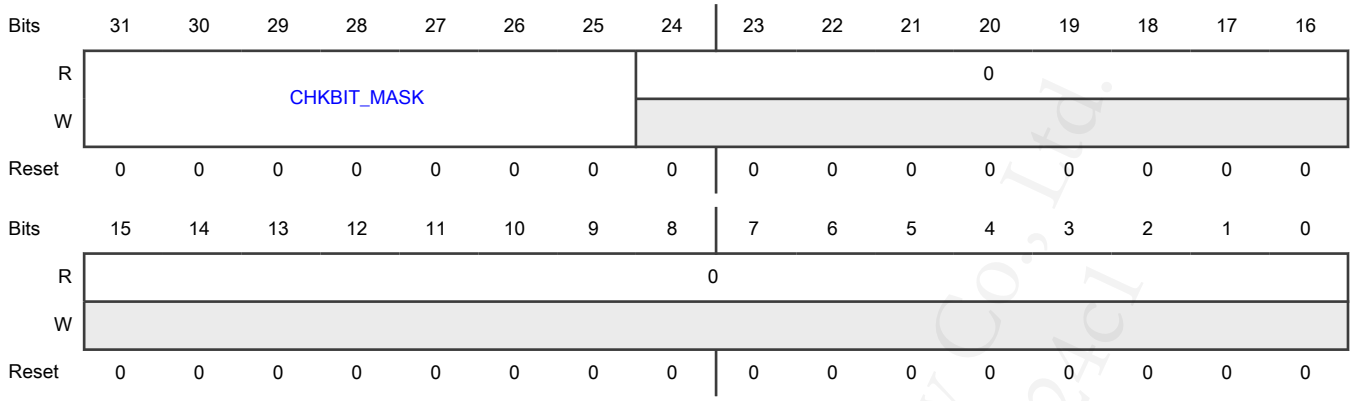
Offset

Register	Offset
EICHD20_WORD0	600h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.16.19 Error Injection Channel Descriptor 21, Word0 (EICHD21_WORD0)

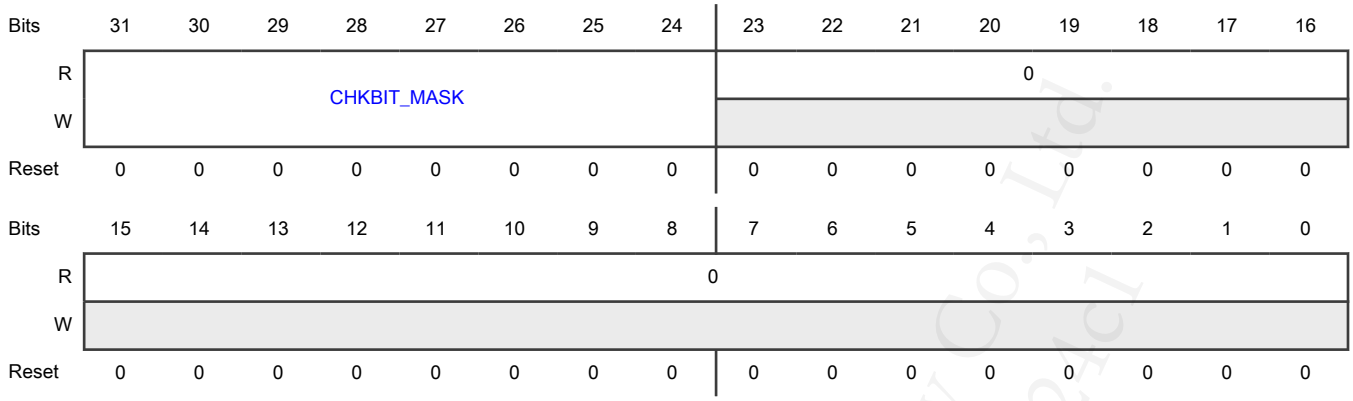
Offset

Register	Offset
EICHD21_WORD0	640h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.16.20 Error Injection Channel Descriptor 22, Word0 (EICHD22_WORD0)

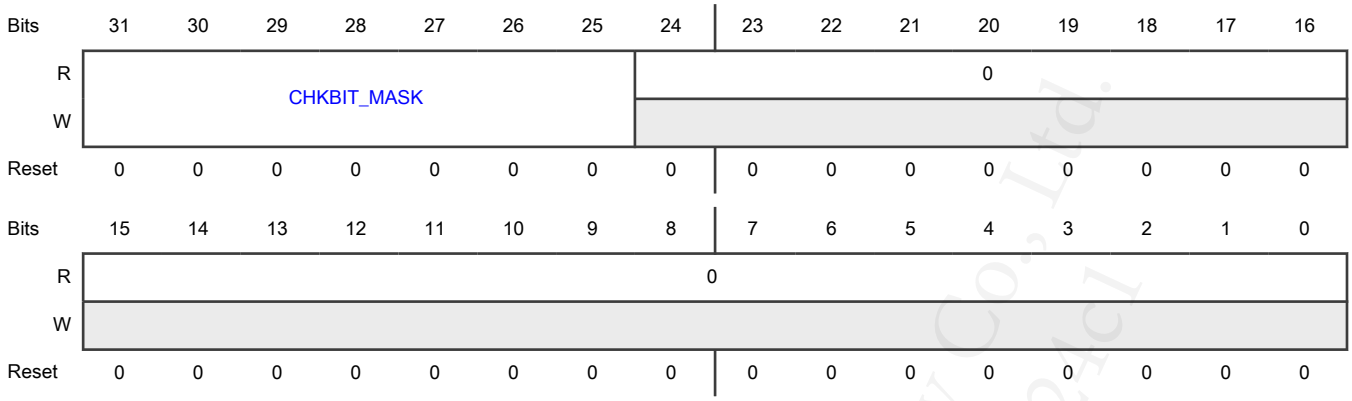
Offset

Register	Offset
EICHD22_WORD0	680h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.16.21 Error Injection Channel Descriptor 23, Word0 (EICHD23_WORD0)

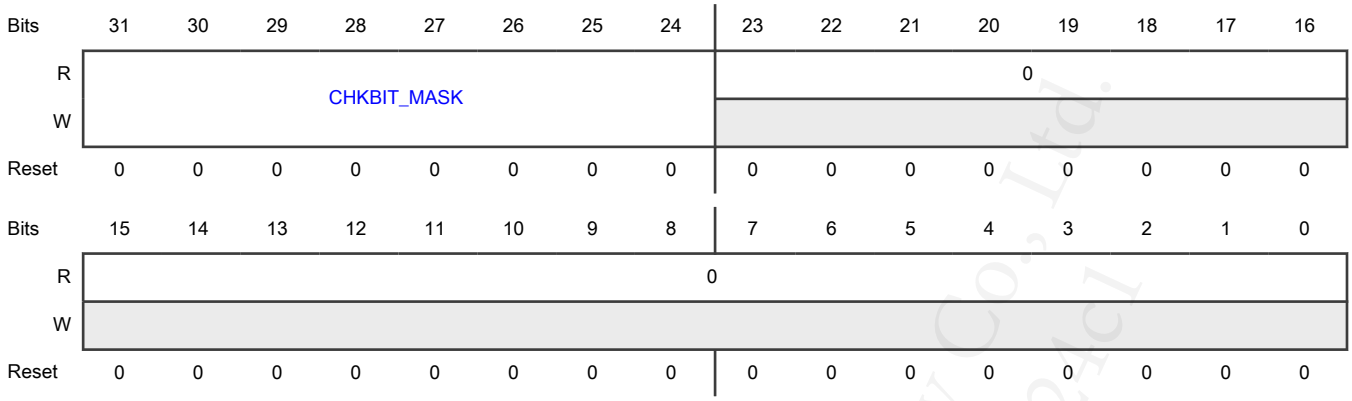
Offset

Register	Offset
EICHD23_WORD0	6C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.16.22 Error Injection Channel Descriptor 24, Word0 (EICHD24_WORD0)

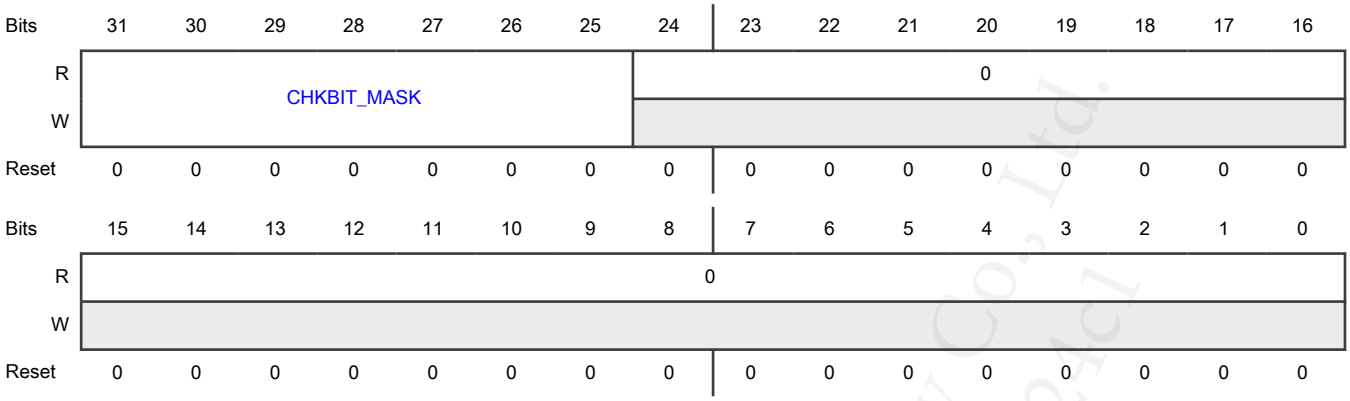
Offset

Register	Offset
EICHD24_WORD0	700h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.16.23 Error Injection Channel Descriptor 25, Word0 (EICHD25_WORD0)

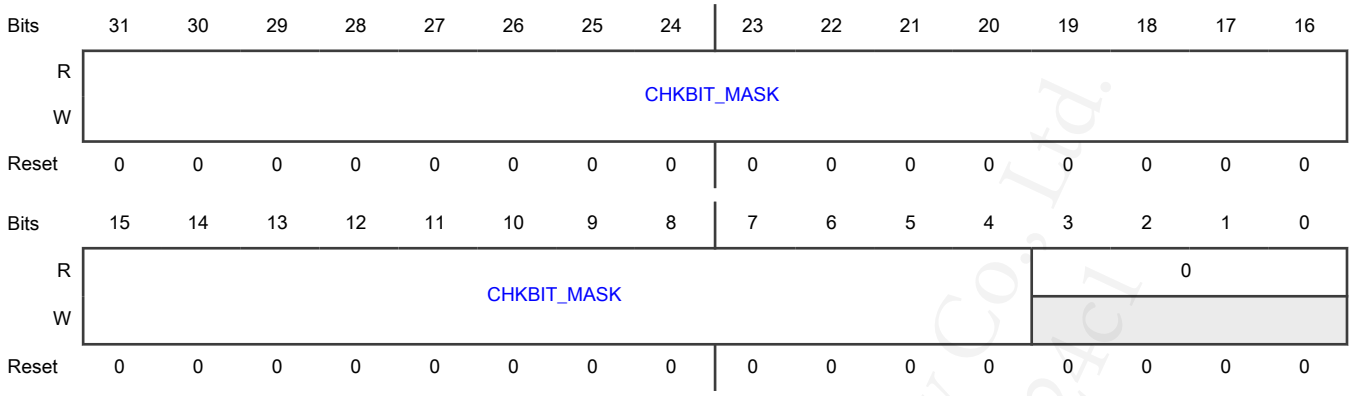
Offset

Register	Offset
EICHD25_WORD0	740h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-4 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[27:0] (28 bits wide), CHKBIT_MASK[27] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
3-0 —	Reserved

73.16.24 Error Injection Channel Descriptor 25, Word3 (EICHD25_WORD3)

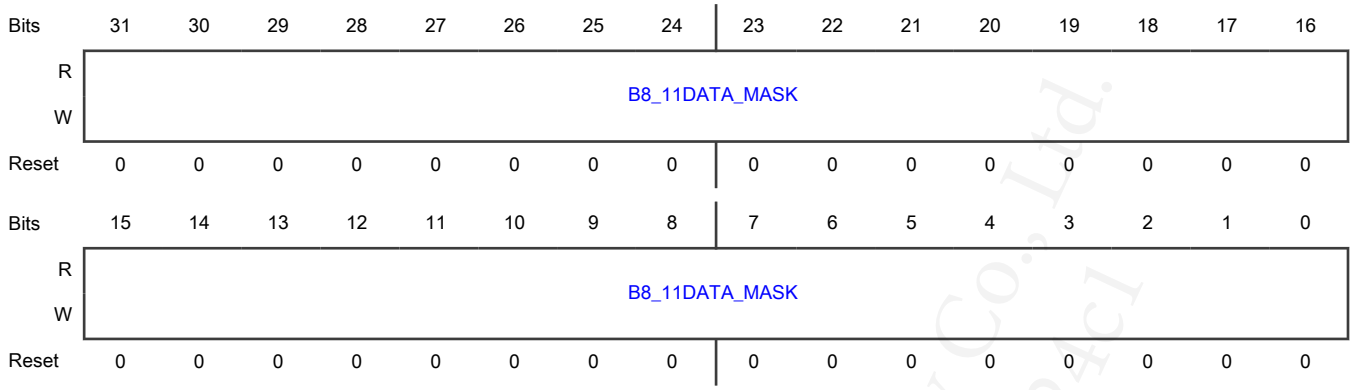
Offset

Register	Offset
EICHD25_WORD3	74Ch

Function

The fourth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B8_11DATA_MASK correspond to bytes 8–11 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B8_11DATA_MASK	Data Mask Bytes 8-11 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B8_11DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 8-11 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 8-11 on the read data bus is inverted.

73.16.25 Error Injection Channel Descriptor 25, Word4 (EICHD25_WORD4)

Offset

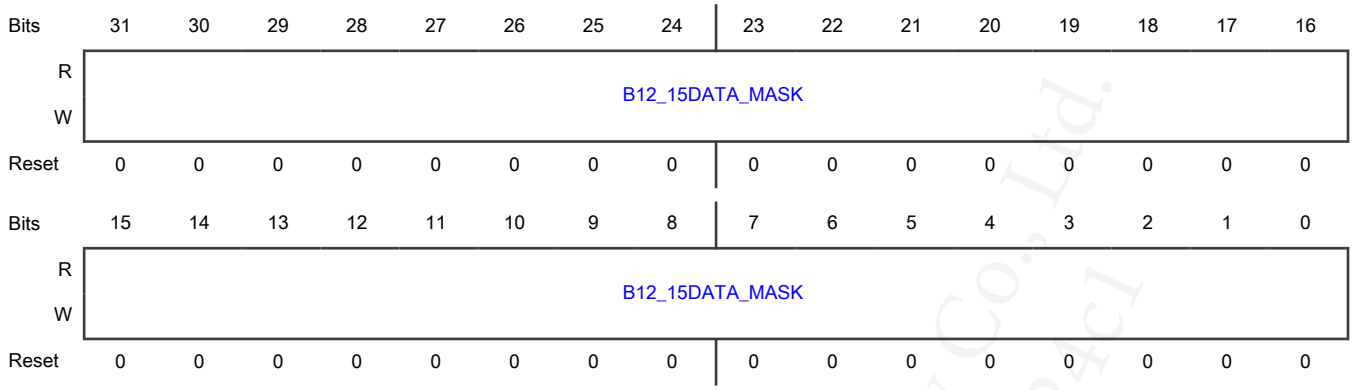
Register	Offset
EICHD25_WORD4	750h

Function

The fifth word of the Error Injection Channel Descriptor, when present, defines a right-justified mask field. The bits in B12_15DATA_MASK correspond to bytes 12–15 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46220-4821-48a6-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 B12_15DATA_MASK	Data Mask Bytes 12-15 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. **NOTE** For the specific DATA_MASK bits to which B12_15DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 12-15 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 12-15 on the read data bus is inverted.

73.16.26 Error Injection Channel Descriptor 26, Word0 (EICHD26_WORD0)

Offset

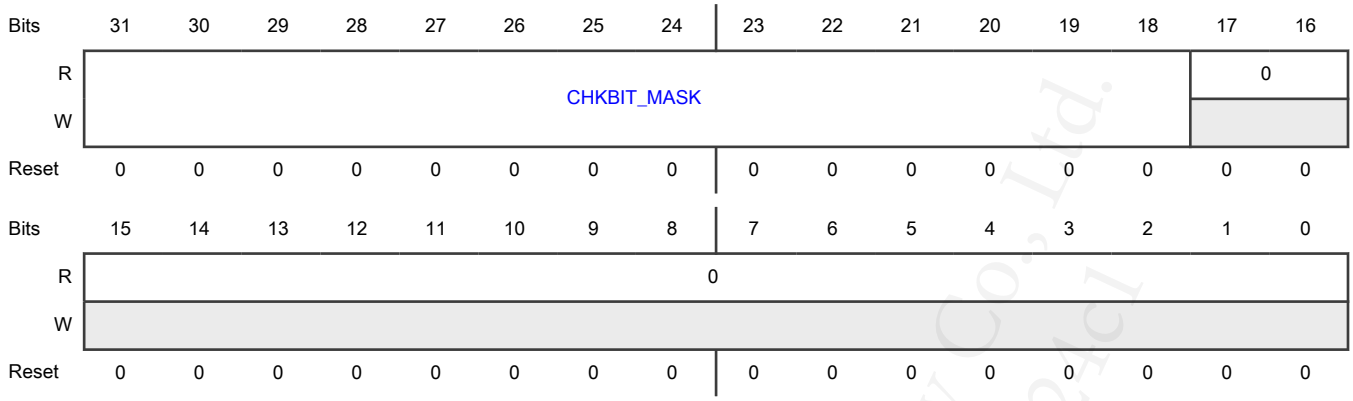
Register	Offset
EICHD26_WORD0	780h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing HHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a6-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-18 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[13:0] (14 bits wide), CHKBIT_MASK[13] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
17-0 —	Reserved

73.16.27 Error Injection Channel Descriptor 27, Word0 (EICHD27_WORD0)

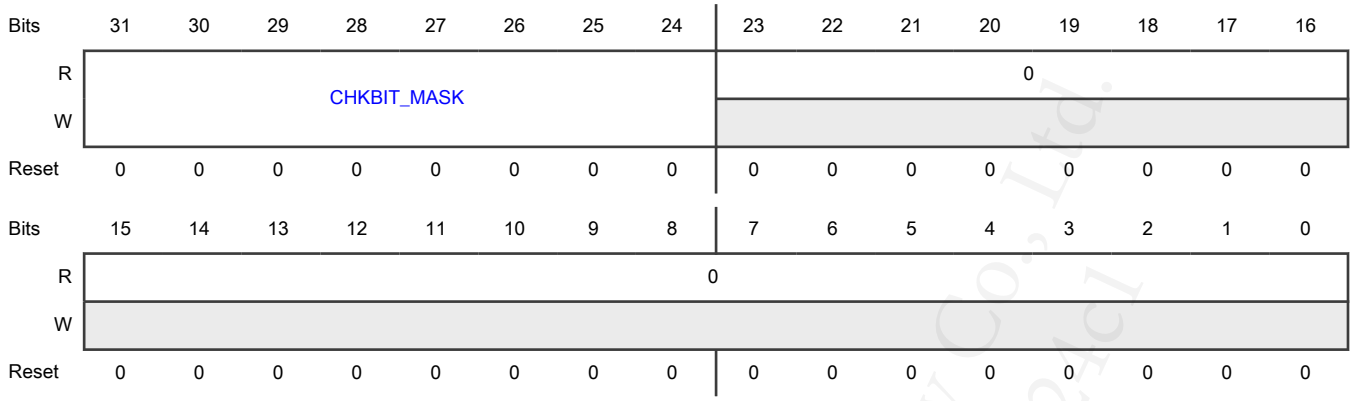
Offset

Register	Offset
EICHD27_WORD0	7C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-24 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[7:0] (8 bits wide), CHKBIT_MASK[7] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
23-0 —	Reserved

73.16.28 Error Injection Channel Descriptor 28, Word0 (EICHD28_WORD0)

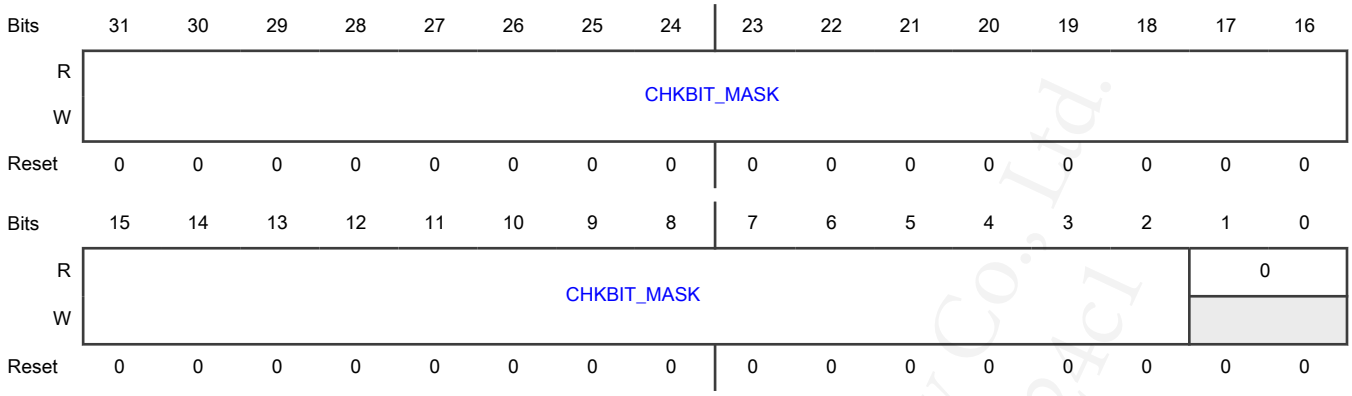
Offset

Register	Offset
EICHD28_WORD0	800h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-2 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[29:0] (30 bits wide), CHKBIT_MASK[29] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
1-0 —	Reserved

73.16.29 Error Injection Channel Descriptor n, Word1 (EICHD28_WORD1 - EICHD29_WORD1)

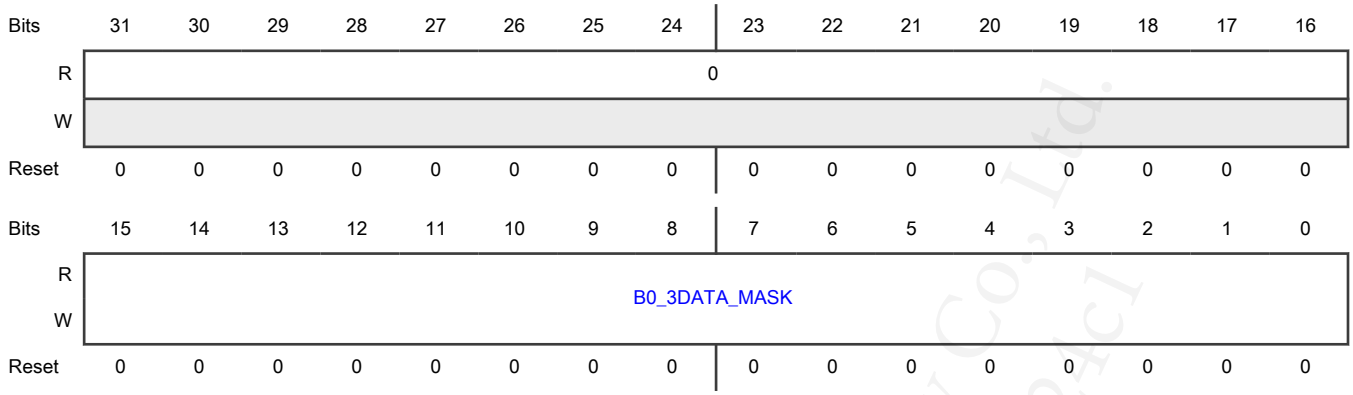
Offset

Register	Offset
EICHD28_WORD1	804h
EICHD29_WORD1	844h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-16 —	Reserved
15-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.16.30 Error Injection Channel Descriptor 29, Word0 (EICHD29_WORD0)

Offset

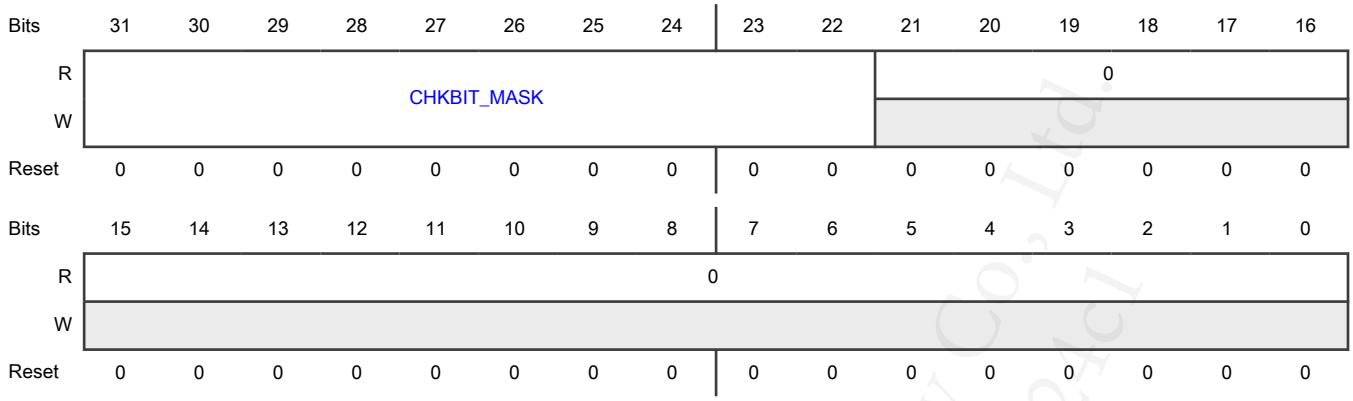
Register	Offset
EICHD29_WORD0	840h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4e5-4821-48a8-b513-90d3ab624c1

Diagram



Fields

Field	Function
31-22 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[9:0] (10 bits wide), CHKBIT_MASK[9] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
21-0 —	Reserved

73.16.31 Error Injection Channel Descriptor n, Word0 (EICHD30_WORD0 - EICHD31_WORD0)

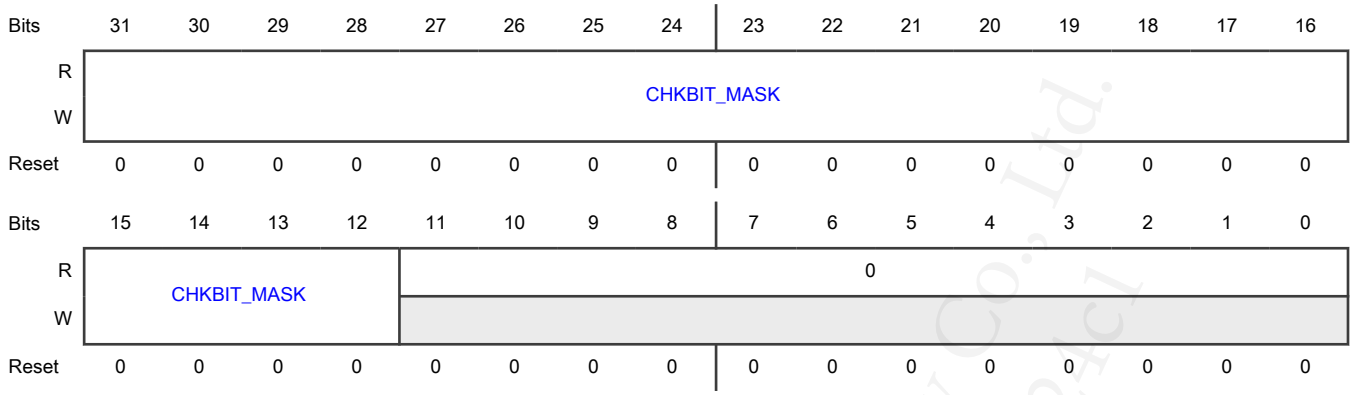
Offset

Register	Offset
EICHD30_WORD0	880h
EICHD31_WORD0	8C0h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-12 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[19:0] (20 bits wide), CHKBIT_MASK[19] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
11-0 —	Reserved

73.16.32 Error Injection Channel Descriptor n, Word1 (EICHD30_WORD1 - EICHD31_WORD1)

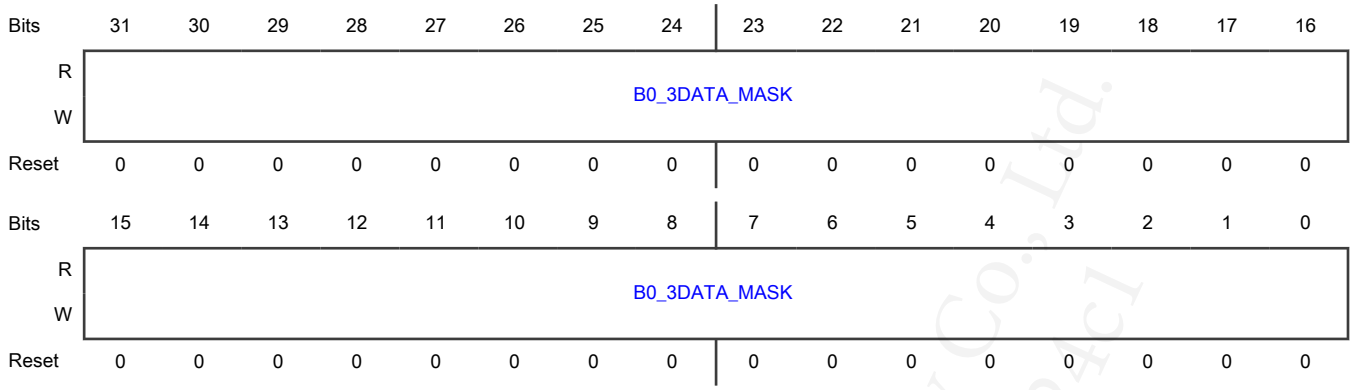
Offset

Register	Offset
EICHD30_WORD1	884h
EICHD31_WORD1	8C4h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.17 EIM_PFE6 register descriptions

The EIM provides a programming model mapped to an on-platform peripheral slot.

Programming model access

All system bus masters can access the programming model:

- Only in supervisor mode
- Using only 32-bit (word) accesses

Any of the following attempted references to the programming model generates an error termination:

- In user mode
- Using non-32-bit access sizes
- To undefined (reserved) addresses

Attempted updates to the programming model while the EIM is in the midst of an operation result in non-deterministic behavior.

Error injection channel descriptor: function and structure

Each error injection channel descriptor:

- Specifies a mask that defines which bits of the read data and/or checkbit bus from target RAM are inverted on a read access.
- Consists of a 128-bit (16-byte) structure, composed of four 32-bit words, in the EIM programming model. Unused words are not documented.

— Word0 (EICHD*n*_WORD0), if present, defines the checkbit mask.

— Word1-3 (EICHD*n*_WORD1-3), if present, define the data mask. Word2 and Word3 are present only when required by the total width of the channel's data mask. See Error injection channel descriptor: DATA_MASK details.

The multiple channel descriptors are organized sequentially.

Error injection channel descriptor: DATA_MASK details

For each channel: The following table shows the total width of DATA_MASK and the distribution of its bits across the WORD registers.

Table 600. Error injection channel descriptor: DATA_MASK details

Channel	DATA_MASK total width (bits)	Specific bits of DATA_MASK in		
		WORD1	WORD2	WORD3
0	32	31-0	—	—
1	32	31-0	—	—

73.17.1 EIM_PFE6 memory map

EIM_PFE6 base address: 4405_A000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Error Injection Module Configuration Register (EIMCR)	32	RW	0000_0000h
4h	Error Injection Channel Enable register (EICHEN)	32	RW	0000_0000h
100h	Error Injection Channel Descriptor 0, Word0 (EICHD0_WORD0)	32	RW	0000_0000h
104h	Error Injection Channel Descriptor 0, Word1 (EICHD0_WORD1)	32	RW	0000_0000h
140h	Error Injection Channel Descriptor 1, Word0 (EICHD1_WORD0)	32	RW	0000_0000h
144h	Error Injection Channel Descriptor 1, Word1 (EICHD1_WORD1)	32	RW	0000_0000h

73.17.2 Error Injection Module Configuration Register (EIMCR)

Offset

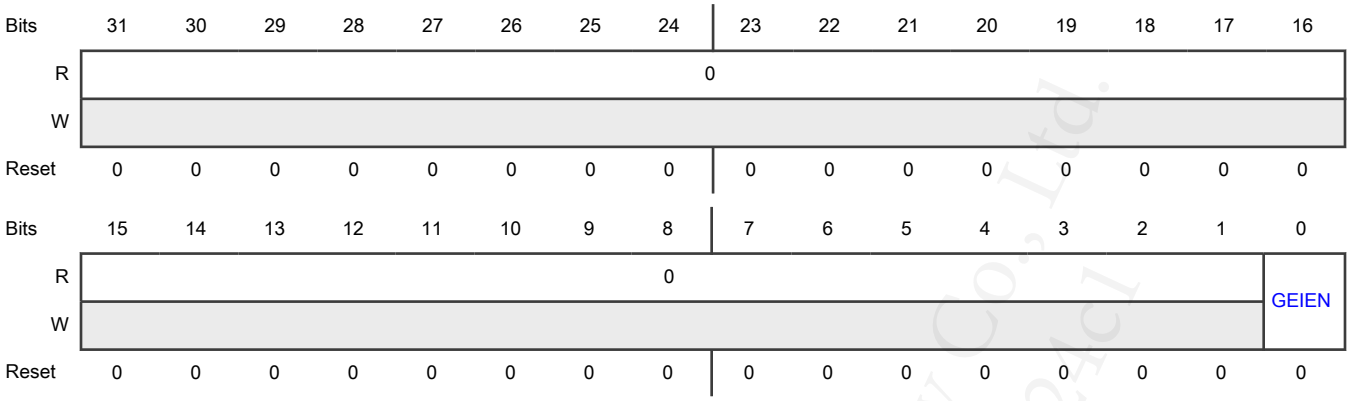
Register	Offset
EIMCR	0h

Function

The EIM Configuration Register is used to globally enable/disable the error injection function.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4e0e-9251-390d3ab62a11

Diagram



Fields

Field	Function
31-1 —	Reserved
0 GEIEN	Global Error Injection Enable This bit globally enables or disables the error injection function of the EIM. This field is initialized by hardware reset. 0b - Disabled 1b - Enabled

73.17.3 Error Injection Channel Enable register (EICHEN)

Offset

Register	Offset
EICHEN	4h

Function

Each field of the Error Injection Channel Enable register (EICHEN) is used to enable or disable the corresponding error injection channel.

NOTE

To enable an error injection channel, the Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-821-48a8-b251-390d8ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	EICH0EN	EICH1EN	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 EICH0EN	Error Injection Channel 0 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 0 1b - Error injection is enabled on Error Injection Channel 0
30 EICH1EN	Error Injection Channel 1 Enable This field enables the corresponding error injection channel. The Global Error Injection Enable (EIMCR[GEIEN]) field must also be asserted to enable error injection. After error injection is enabled, all subsequent read accesses incur one or more bit inversions as defined in the corresponding EICHD*n*_WORD registers. Error injection remains in effect until the error injection channel is manually disabled via software. Any write to the corresponding EICHD*n*_WORD registers clears the corresponding EICHEN[EICH*n*EN] field, disabling the error injection channel. 0b - Error injection is disabled on Error Injection Channel 1 1b - Error injection is enabled on Error Injection Channel 1
29 —	Reserved
28 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
27 —	Reserved
26 —	Reserved
25 —	Reserved
24 —	Reserved
23 —	Reserved
22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 —	Reserved
13	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continues on the next page...

Table continued from the previous page...

Field	Function
— —	
12 —	Reserved
11 —	Reserved
10 —	Reserved
9 —	Reserved
8 —	Reserved
7 —	Reserved
6 —	Reserved
5 —	Reserved
4 —	Reserved
3 —	Reserved
2 —	Reserved
1 —	Reserved
0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

73.17.4 Error Injection Channel Descriptor n, Word0 (EICHD0_WORD0 - EICHD1_WORD0)

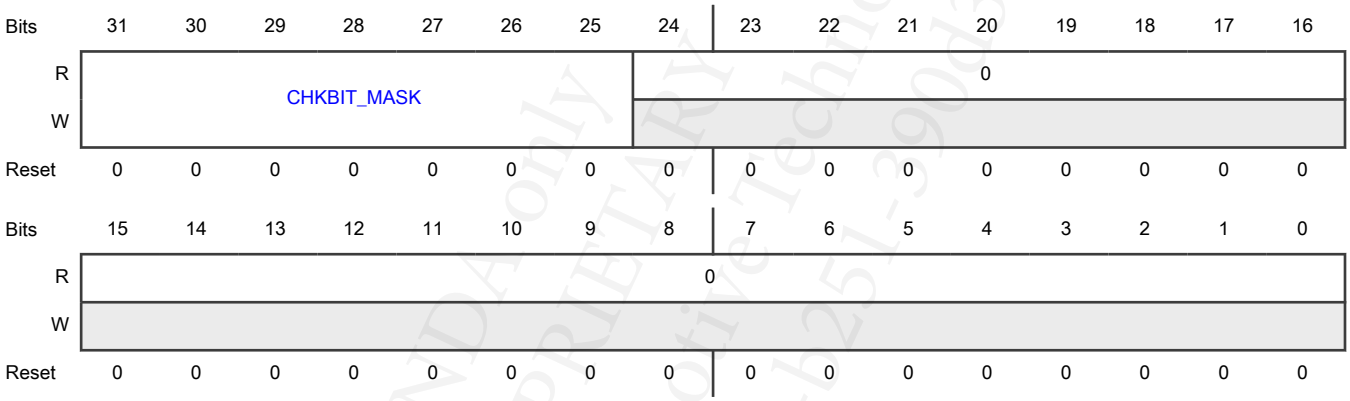
Offset

Register	Offset
EICHD0_WORD0	100h
EICHD1_WORD0	140h

Function

The first word of the Error Injection Channel Descriptor defines a left-justified mask field: CHKBIT_MASK. Each bit of CHKBIT_MASK specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified on read accesses. Successful write to this field clears the corresponding error injection channel valid bit, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-25 CHKBIT_MASK	Checkbit Mask This field defines a bit-mapped mask that specifies whether the corresponding bit of the checkbit bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. For any unique details about the mapping of CHKBIT_MASK's bits to a channel's target RAM, see the chip-specific EIM information. **NOTE** Because CHKBIT_MASK is left-justified, the highest bit in the bit range is always in the position of the most significant bit. For CHKBIT_MASK[6:0] (7 bits wide), CHKBIT_MASK[6] is in the position of the most significant bit. 0b - The corresponding bit of the checkbit bus remains unmodified. 1b - The corresponding bit of the checkbit bus is inverted.
24-0 —	Reserved

73.17.5 Error Injection Channel Descriptor n, Word1 (EICHD0_WORD1 - EICHD1_WORD1)

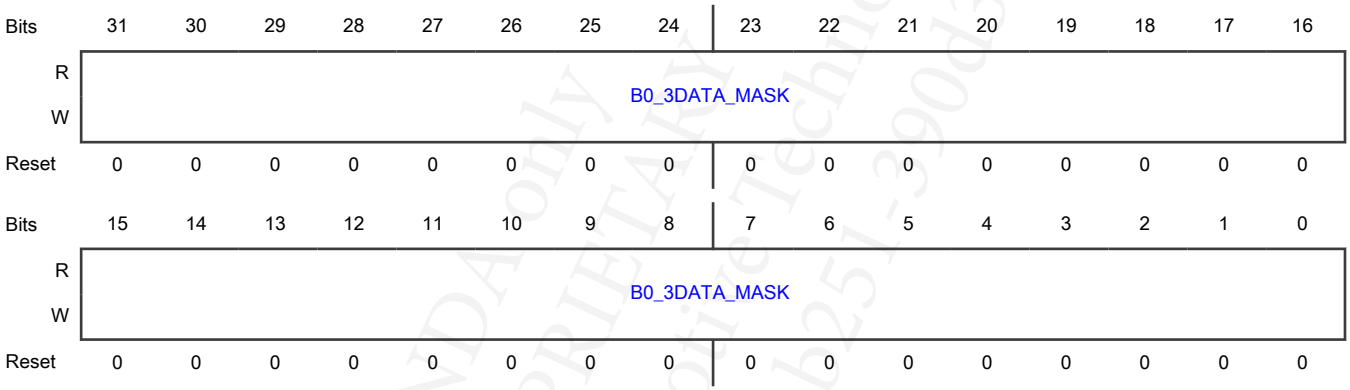
Offset

Register	Offset
EICHD0_WORD1	104h
EICHD1_WORD1	144h

Function

The second word of the Error Injection Channel Descriptor defines a right-justified mask field. The bits in B0_3DATA_MASK correspond to bytes 0–3 of the read data bus. Each bit specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified on read accesses. A successful write to this field clears the corresponding error injection channel valid field, EICHEN[EICH*n*EN].

Diagram



Fields

Field	Function
31-0 B0_3DATA_MASK	Data Mask Bytes 0-3 This field defines a bit-mapped mask that specifies whether the corresponding bit of the read data bus from the target RAM should be inverted or remain unmodified. Writes to unimplemented bits are ignored. **NOTE** For the specific DATA_MASK bits to which B0_3DATA_MASK corresponds, See Error injection channel descriptor: DATA_MASK details. 0b - The corresponding bit of bytes 0-3 on the read data bus remains unmodified. 1b - The corresponding bit of bytes 0-3 on the read data bus is inverted.

73.18 Functional description

The EIM provides protection against accidental enabling and reconfiguration of the error injection function by enforcing a two-stage enablement mechanism. To properly enable the error injection mechanism for a channel:

- Write 1 to the EICHEN[EICH*n*EN] field, where *n* denotes the channel number.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c5c46e25-8187-48a8-b251-390d3ab64c1

- Write 1 to EIMCR[GEIEN].

NOTE

When the use case for a channel requires writing any EICHD*n*_WORD register, write the EICHD*n*_WORD register before executing the two-stage enablement mechanism. A successful write to any EICHD*n*_WORD register clears the corresponding EICHEN[EICH*n*EN] field.

The EIM supports 31 error injection channels. Each channel:

- Can be assigned to a single memory array interface by intercepting the assigned memory read data bus and checkbit bus, and injects errors by inverting the value transmitted for selected bits on each bus line.
- Can be assigned to a redundant comparison unit by intercepting the signals being compared, and injecting errors by inverting the value transmitted for selected bits on each bus line.

On a memory read access, the applicable EICHD*n*_WORD registers define which bits of the read data and/or checkbit bus to invert.

Figure 575 depicts the interception and override of a 64-bit read data bus and an 8-bit checkbit data bus for an example memory array.

Error injection scenarios

The EIM supports these cases of error injection:

- To generate a single-bit error, invert only 1 bit of the CHKBIT_MASK or DATA_MASK in the EICHD*n*_WORD registers.
- To generate a multi-bit error, invert only 2 bits of the CHKBIT_MASK or DATA_MASK in the EICHD*n*_WORD registers.

NOTE

An attempt to invert more than 2 bits in one operation might result in undefined behavior.

73.19 Glossary

SEC-DED	Single error correct – dual error detect
ECC	Error correction code

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b25f-599d3aB62461

Chapter 74
Cyclic Redundancy Check (CRC) Unit

74.1 Introduction

The Cyclic Redundancy Check (CRC) computing unit is dedicated to the computation of CRC, off-loading the CPU. Each context has a separate CRC computation engine to allow the concurrent computation of the CRC of multiple data streams. The CRC computation is performed at speed without wait-state insertion. Bit-swap and bit-inversion operations can be applied to the final CRC signature. Each context can be configured with one of the four hard-wired polynomials, normally used for most of the standard communication protocols. The data stream supports multiple data width (byte/halfword/word) formats.

74.2 Main features

- Peripheral bus interface
- Three contexts for the concurrent CRC computation are supported
- Separate CRC engine for each context
- Zero-wait states during the CRC computation (pipeline scheme)
- Four hard-wired polynomials
  - CRC-8 VDA CAN
  - CRC-16-CCITT
  - CRC-32 ITU-T V.42 (see note below)
  - CRC-8-H2F AUTOSAR polynomial
- Support for byte, halfword, or word width of the input data stream

NOTE

Although the CRC-32 ITU-T V.42 polynomial is used for CRC generation, the generation algorithm MISR differs from the one used by the IEEE 802.3 standard.

74.3 Block diagram

See the following figure for an overview of the CRC module. See Main features for number of contexts in this module.

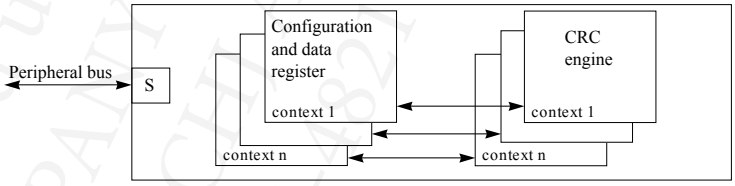


Figure 576. CRC top-level diagram

74.4 External signal description

The CRC module does not generate any external signals.

74.4.1 Peripheral bus interface

The peripheral bus interface is a slave bus used for configuration and data streaming (CRC computation) purposes via CPU or DMA. The following bus operations (contiguous byte enables) are supported:

- Word (32 bits) data write/read operations to any registers
- Low and high halfword (16 bits, data[31:16] or data[15:0]) data write/read operations to any registers
- Byte (8 bits, data[31:24], data[23:16], data[15:8], or data[7:0]) data write/read operations to any registers
- Any other operation (free byte enables or other operations) must be avoided

The CRC module generates a transfer error in the following cases:

- Any write/read access to the register addresses not mapped to the peripheral but included in the address space of the peripheral
- Any write/read operation different from byte/halfword/word (free byte enables or other operations) on each register

The registers of the CRC module are read/write accessible in each access mode:

- User
- Supervisor

The following summarizes bus operation performance:

- Zero wait states (single bus cycle) for each write/read operation to the CRC_CFG and CRC_INP registers
- Zero wait states (single bus cycle) for each write operation to the CRC_ CSTAT register
- Double wait states (3 bus cycles) for each read operation to the CRC_ CSTAT or CRC_OUTP registers immediately following (next clock cycle) a write operation to the CRC_CSTAT, CRC_INP, or CRC_CFG registers belonging to the same context; in all the other cases, no wait states are inserted

74.5 CRC register descriptions

74.5.1 CRC memory map

CRC_0 base address: 4019_0000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Configuration Register (CFG1)	32	RW	0000_0000h
4h	Input Register (INP1)	32	RW	0000_0000h
8h	Current Status Register (CSTAT1)	32	RW	FFFF_FFFFh
Ch	Output Register (OUTP1)	32	RO	FFFF_FFFFh
10h	Configuration Register (CFG2)	32	RW	0000_0004h
14h	Input Register (INP2)	32	RW	0000_0000h
18h	Current Status Register (CSTAT2)	32	RW	FFFF_FFFFh
1Ch	Output Register (OUTP2)	32	RO	FFFF_FFFFh
20h	Configuration Register (CFG3)	32	RW	0000_0000h
24h	Input Register (INP3)	32	RW	0000_0000h
28h	Current Status Register (CSTAT3)	32	RW	FFFF_FFFFh
2Ch	Output Register (OUTP3)	32	RO	FFFF_FFFFh

74.5.2 Configuration Register (CFG1 - CFG3)

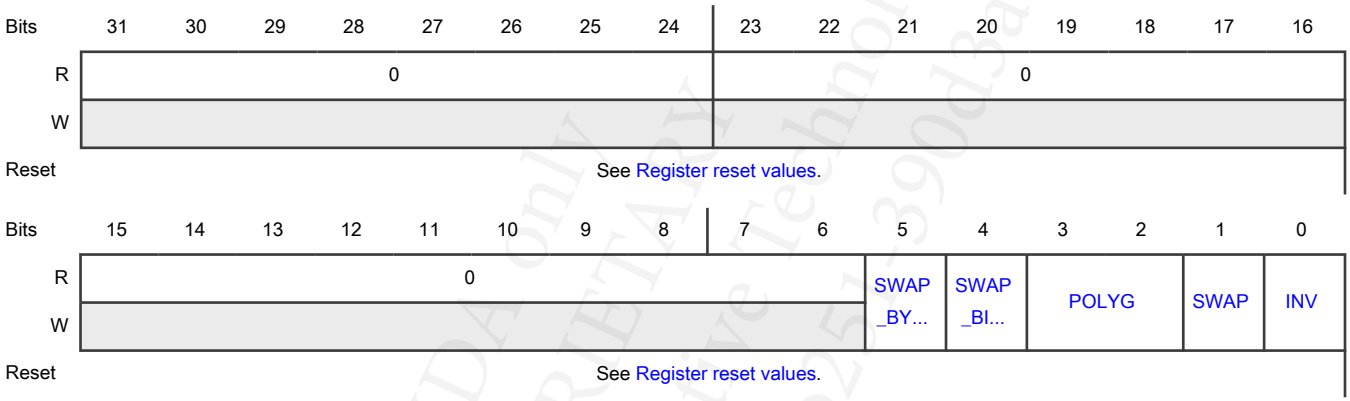
Offset

Register	Offset
CFG1	0h
CFG2	10h
CFG3	20h

Function

Access: User read/write.

Diagram



Register reset values

Register	Reset value
CFG1	0000_0000h
CFG2	0000_0004h
CFG3	0000_0000h

Fields

Field	Function
31-24 —	Reserved
23-6 —	Reserved
5	Swap CRC_INP byte-wise

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482a-4e8-b251-390d8ab624c1

Table continued from the previous page...

Field	Function
SWAP_BYTEWISE	**NOTE** INV and SWAP bits are set to one for CRC-32 polynomial calculations. 0b - Do not swap 1b - Perform byte-wise swap on CRC_INP input data internally for CRC-32 polynomial calculations
4 SWAP_BITWISE	Swap CRC_INP bit-wise 0b - Do not swap 1b - Perform bit-wise swap on CRC_INP input data internally for CRC-8 and CRC-16 polynomial calculations
3-2 POLYG	Polynomial selection This bit can be written only during the configuration phase. 00b - CRC-CCITT polynomial 01b - CRC-32 polynomial 10b - CRC-8 polynomial 11b - CRC-8-H2F AUTOSAR polynomial
1 SWAP	Swap selection This bit can be written only during the configuration phase. The swap operation provides a bit-by-bit swapping of the content. 0b - No swap selection applied on the CRC_OUTP content 1b - Swap selection (MSB to LSB, LSB to MSB) applied to the CRC_OUTP content
0 INV	Inversion selection This bit can be written only during the configuration phase. The inversion operation is a complement (or negation) of the content. 0b - No inversion selection applied on the CRC_OUTP content 1b - Inversion selection (bit x bit) applied on the CRC_OUTP content

74.5.3 Input Register (INP1 - INP3)

Offset

Register	Offset
INP1	4h
INP2	14h
INP3	24h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-46b1-48a3-b251-390d3ab624c1

Function

Access: User read/write.

NOTE

High-level code must be written in a way to realize byte-, halfword-, and word-wide writes on the assembler code level. Any endianness problem faced can be corrected by appropriate configuration of CRC_CFG[SWAP_BYTEWISE] and CRC_CFG[SWAP_BITWISE].

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	INP															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	INP															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 INP	Input data for the CRC computation This register can be written with word-, halfword- (high or low), or byte-wide writes in any sequence. Only the bits written are fed to the CRC engine. In case of halfword write operation, the bytes must be contiguous.

74.5.4 Current Status Register (CSTAT1 - CSTAT3)

Offset

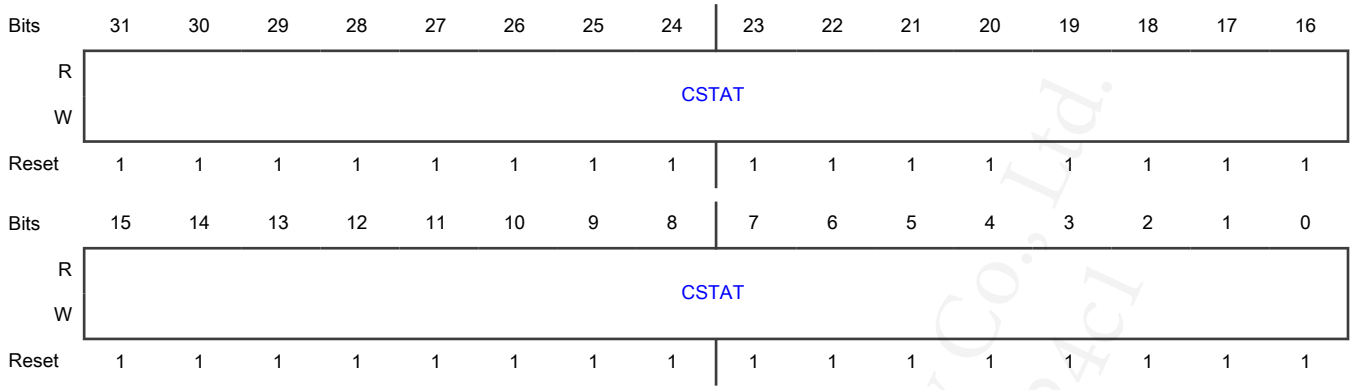
Register	Offset
CSTAT1	8h
CSTAT2	18h
CSTAT3	28h

Function

Access: User read/write.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-251-390d3ab624c1

Diagram



Fields

Field	Function
31-0 CSTAT	CRC signature status This register includes the current status of the CRC signature. No bit swap and inversion are applied to this register. In the case of the CRC-CCITT polynomial, only the 16 least-significant bits are significant. The 16 most-significant bits are set to zero during the computation. In the case of the CRC-8 polynomial, only the 8 least-significant bits are significant. The 24 most-significant bits are set to zero during the computation. The CSTAT register can be written at byte, halfword, or word. This register can be written only during the configuration phase.

74.5.5 Output Register (OUTP1 - OUTP3)

Offset

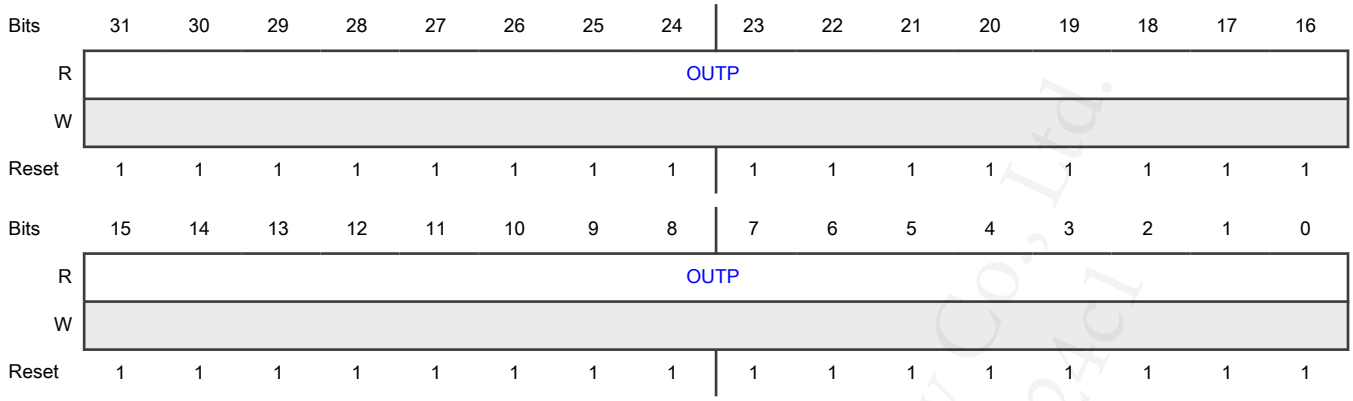
Register	Offset
OUTP1	Ch
OUTP2	1Ch
OUTP3	2Ch

Function

Access: User read-only.

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c462e5-4821-48a8-b251-d3ab624c1

Diagram



Fields

Field	Function
31-0 OUTP	Final CRC signature This register includes the final signature corresponding to the CRC_CSTAT register value eventually swapped and inverted. In the case of the CRC-CCITT polynomial, only the 16 least-significant bits are significant. The 16 MSB bits are set to 0 during the computation. In the case of the CRC-8 polynomial, only the 8 least-significant bits are significant. The 24 most-significant bits are set to 0 during the computation.

74.6 Functional description

The CRC module supports the CRC computation for each context. Each context has its own complete set of registers, including the CRC engine. The data flow of each context can be interleaved. The data stream can be structured as a sequence of bytes, halfwords, or words. The input data sequence is provided that can consist a mix of data formats (byte, halfword and word), which allows writing to the input data register (CRC_INP).

The data stream is generally executed by *n* concurrent DMA data transfers (mem2mem), where *n* is less or equal to the number of contexts.

The standard generator polynomials are provided in Figure 577, Figure 578, and Figure 579 for the CRC computation of each context. Two separate registers are available: CRC_INP for writing data and CRC_CSTAT for retrieving the (accumulated up to now) CRC.

$$x^8+x^4+x^3+x^2+1$$

Figure 577. CRC8 VDA CAN

$$x^{16}+x^{12}+x^5+1$$

Figure 578. CRC-CCITT (x.25 protocol)

$$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$$

Figure 579. CRC-32 (ITU-T V.42 protocol)

$$x^5+x^3+x^2+x+1$$

Figure 580. CRC-8-H2F (AUTOSAR 4.0 protocol)

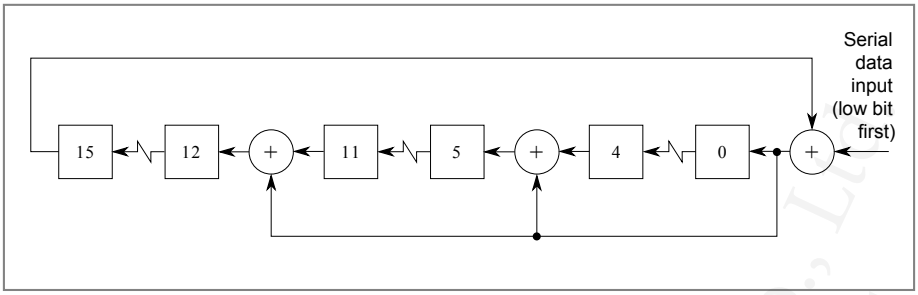


Figure 581. CRC-CCITT engine concept scheme

The initial seed value of the CRC can be programmed by initializing the CRC_CSTAT register. A diagram illustrating serial data loading of the CRC engine is shown in Figure 581 for CRC-CCITT. Note that, conceptually, the least-significant or "right-most" bit shown in Input Register is fed first into the engine. The actual implementation executes the CRC computation in a single clock cycle (parallel data loading). A pipeline scheme has been adopted to decouple the IPS bus interface from the CRC engine to allow the computation of the CRC at speed (zero wait states).

If the CRC signature is used for encapsulation in the data frame of a communication protocol (for example, SPI, and so on.), a bit swap (high bit for low bit or low bit for high bit) and/or bit inversion of the final CRC signature can be applied to CRC_OUTP before transmitting the CRC.

Use of the CRC module is summarized in the flow chart provided in Figure 582.

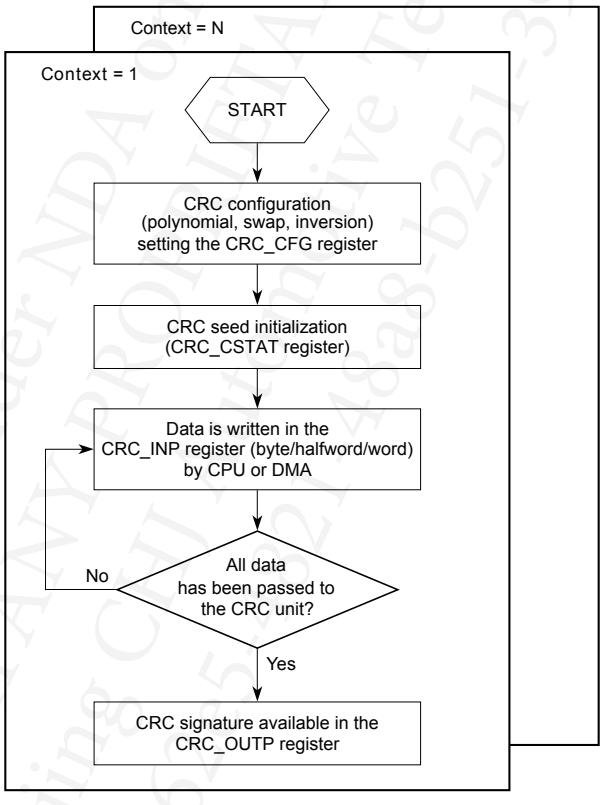


Figure 582. CRC computation flow

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CU Automotive Technology Co., Ltd
c8c40e5e2f4a8-6251-390a8b6c1

74.7 Use cases

74.7.1 Programming example

The number of contexts depends on the application. The overall number of contexts for the CRC peripheral depends on the number of peripherals that concurrently require intervention by the CRC module. Two main use cases are considered:

- Calculation of the CRC of the configuration registers during the process safety time
- Calculation of the CRC on the incoming/outcoming frames for the communication protocols (not protected with CRC by definition of the protocol itself) used as a safety-relevant peripheral

The signature of the configuration registers is computed correctly only if these registers do not contain any status bit.

Assuming that the DMA engine has *n* channels (greater than or equal to the number of contexts) configurable for these types of data transfer—mem2mem, periph2mem, and mem2periph—the following sequence, as shown in Figure 583, is applied to manage the transmission data flow:

1. DMA/CRC module configuration (context x, channel x) by CPU
2. Payload transfer from the MEM to the CRC module (CRC_INP register) after MSB-to-LSB change (input mirroring) to calculate the CRC signature (phase1) by DMA (mem2mem data transfer, channel x)
3. CRC signature copy from the CRC module (CRC_OUTP register) to the MEM (phase 2) by CPU
4. Data block (payload + CRC) transfer from the MEM to the PERIPH module (for example, SPI Tx FIFO) (phase 3) by DMA (mem2periph data transfer, channel x)

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390f8a6262c4

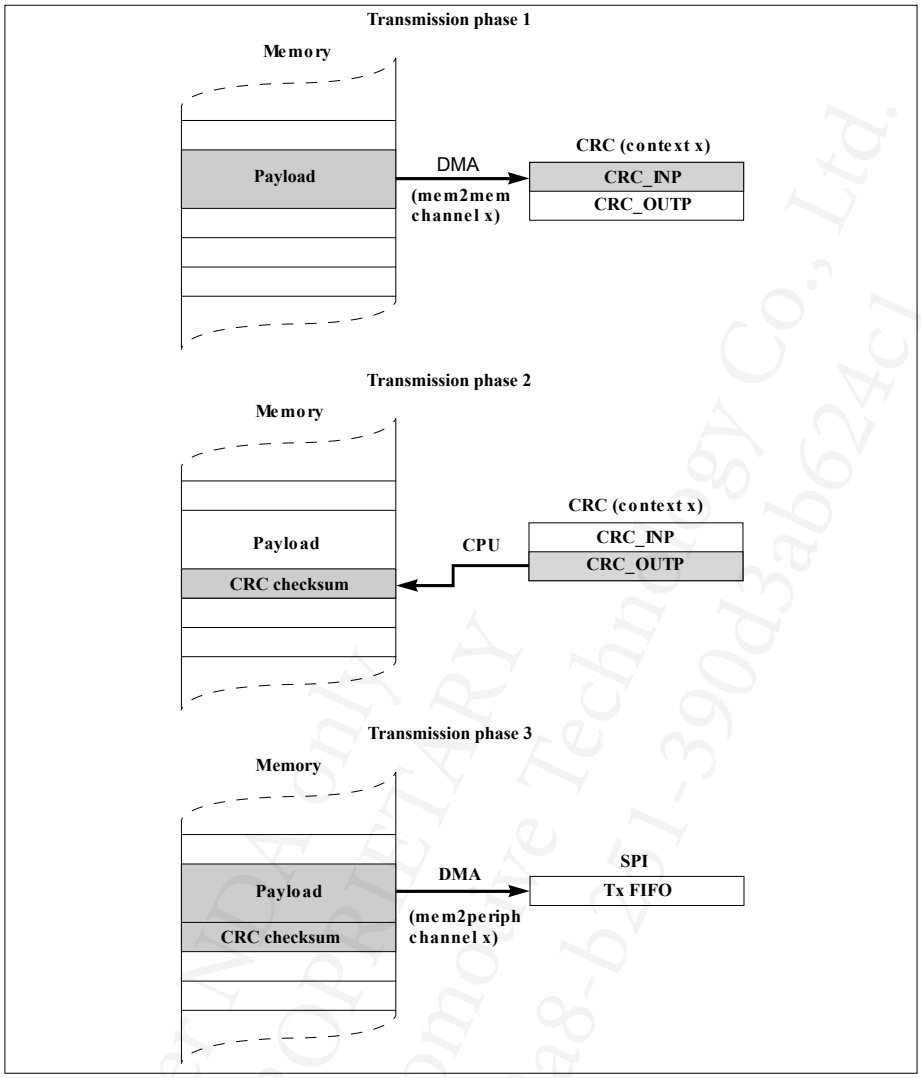


Figure 583. DMA-CRC transmission sequence

74.7.2 Register programming

- INV and/or SWAP (affecting the output only) can be applied to CRC-8, CRC-16, and CRC-32. Both can be applied together.
- SWAP_BITWISE (affecting the input only) can be applied for all polynomials if required by an application.
- SWAP_BYTEWISE (affecting the input only) can be applied for CRC-16 and CRC-32 polynomials only.
- SWAP_BITWISE has priority over SWAP_BYTEWISE when both are applied together.
- When generating CRC-32 for the ITU-T V.42 standard, the user needs to set SWAP_BYTEWISE together with INV and SWAP.
- When generating CRC-16, the user needs to set the SWAP_BITWISE bit.

NOTE

The below results and examples are for **littleEndian** format.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
cc4626f5-1821-48a8-b251-390d3ab624c1

74.7.2.1 Configuration Register values

The configuration register values that must be set for all the hard wired polynomials are as below:

Table 601. Configuration Register values

Bit Field	[31:6]	5	4	3	2	1	0	Hex Value
	RESERVED	SWAP_BYTEWISE	SWAP_BITWISE	POLY		SWAP	INV	
CRC - 16 CCITT	0	0	1	0	0	0	0	0x10
CRC - 32	0	1	0	0	1	1	1	0x27
CRC - 8	0	0	1	1	0	0	1	0x19
CRC - 8 H2F	0	0	1	1	1	0	1	0x1D

74.7.2.2 Current Status Register initial values

The initial values for the Current Status Register that must be set for all the hard wired polynomials are as below:

Table 602. Current Status Register initial values

Bit Field	[31:0]
CRC - 16 CCITT	0x0000_FFFF
CRC - 32	0xFFFF_FFFF
CRC - 8	0x0000_00FF
CRC - 8 H2F	0x0000_00FF

74.7.2.3 Examples of input data and expected output data

Some examples of input data and expected output data are as below:

Table 603. Examples of input data and expected output data

Polynomial	Initial value (CSTAT)	Input data	Output data
CRC - 16 CCITT	0x0000_FFFF	0xA1B2C3D4	0x4063
CRC - 16 CCITT	0x0000_FFFF	0xDEAD_BEEF	0x4097
CRC - 16 CCITT	0x0000_FFFF	0xABCD_1234	0x7C9
CRC - 32	0xFFFF_FFFF	0xA1B2C3D4	0x73201942
CRC - 32	0xFFFF_FFFF	0xDEAD_BEEF	0x7C9CA35A
CRC - 32	0xFFFF_FFFF	0xABCD_1234	0x7B19C06D
CRC - 8	0x0000_00FF	0xA1B2C3D4	0x2B
CRC - 8	0x0000_00FF	0xDEAD_BEEF	0xB3
CRC - 8	0x0000_00FF	0xABCD_1234	0x37
CRC - 8 H2F	0x0000_00FF	0xA1B2C3D4	0xC4

Table continues on the next page...

Table 603. Examples of input data and expected output data (continued)

Polynomial	Initial value (CSTAT)	Input data	Output data
CRC - 8 H2F	0x0000_00FF	0xDEAD_BEEF	0xEB
CRC - 8 H2F	0x0000_00FF	0xABCD_1234	0x37

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 75
Debug

75.1 Introduction

This chapter describes the debug and trace features of this chip. The debug and trace architecture is based on Arm® CoreSight™ architecture. See the Arm CoreSight architecture specification (available in References) for details.

The Arm Cortex M7 and Cortex A53 core-clusters support the debug and trace features described in their Technical Reference Manual (TRM) (available in References). In addition, the debug architecture includes NoC debug and trace modules. These debug components are accessible via the Arm DAP controller-based architecture. The DAP controller works in parallel with the system JTAG Controller (JTAGC). The DAP controller and the JTAGC share the JTAG port and JTAG instruction set.

The debug structure consists of up to three sections. There is a "system" component and Accelerator components. The "system" components are similar on different MCUs in the same family and include the primary cores debug interfaces, the chip level debug interfaces and chip level trace interfaces. The Accelerator components include the debug and trace circuits necessary for any application specific accelerators required for the different application spaces that the MCU supports. These vary from MCU to MCU and could include only trace components.

75.2 Block diagram

The figure below shows the DAP architecture of the Cortex-A53 based MCU.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3a6f0f7a

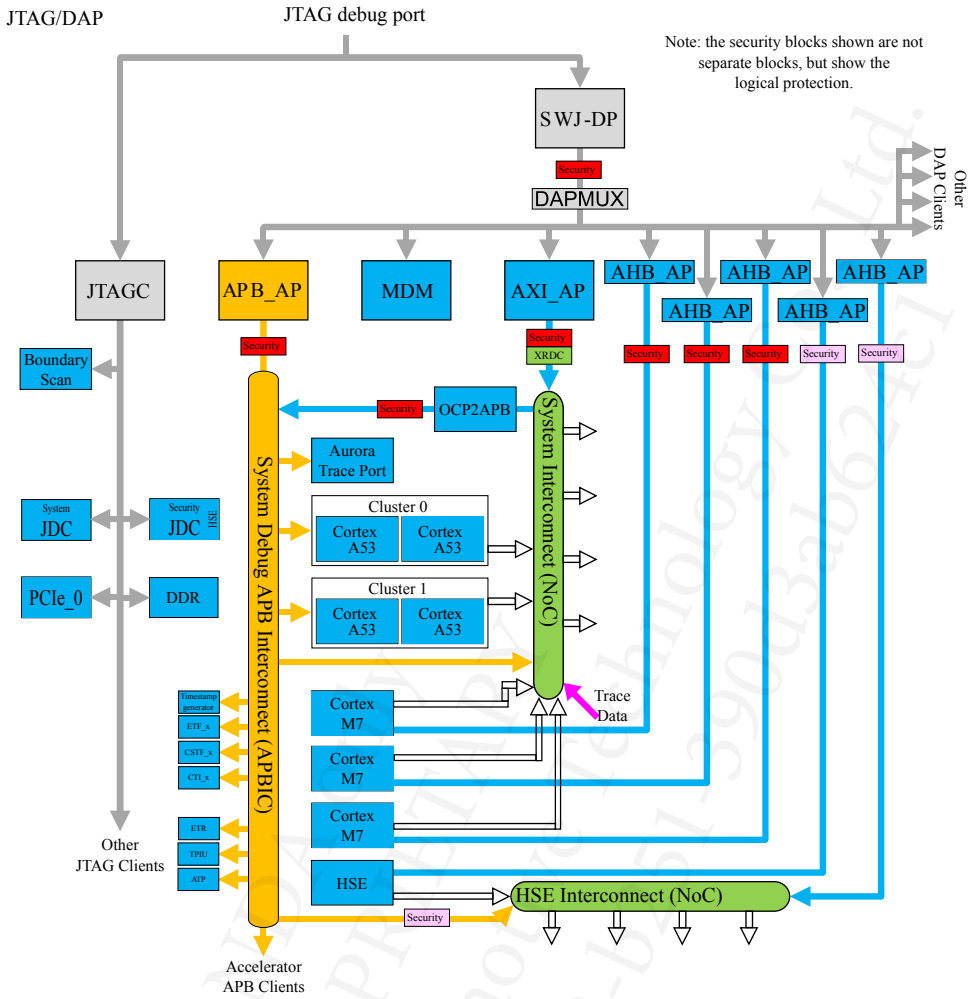


Figure 584. Debug block diagram

75.2.1 S32G accelerator subsystem debug block diagram

The accelerator subsystem of each MCU contains additional debug logic as shown in the following figure. This additional circuitry connects into the full system debug architecture.

Provided under NXP PROPRIETARY COMPANY Beijing CHJ Automotive Technology Co., Ltd. c8c462e5-4821-4ba8-b513-590d3ab624c1

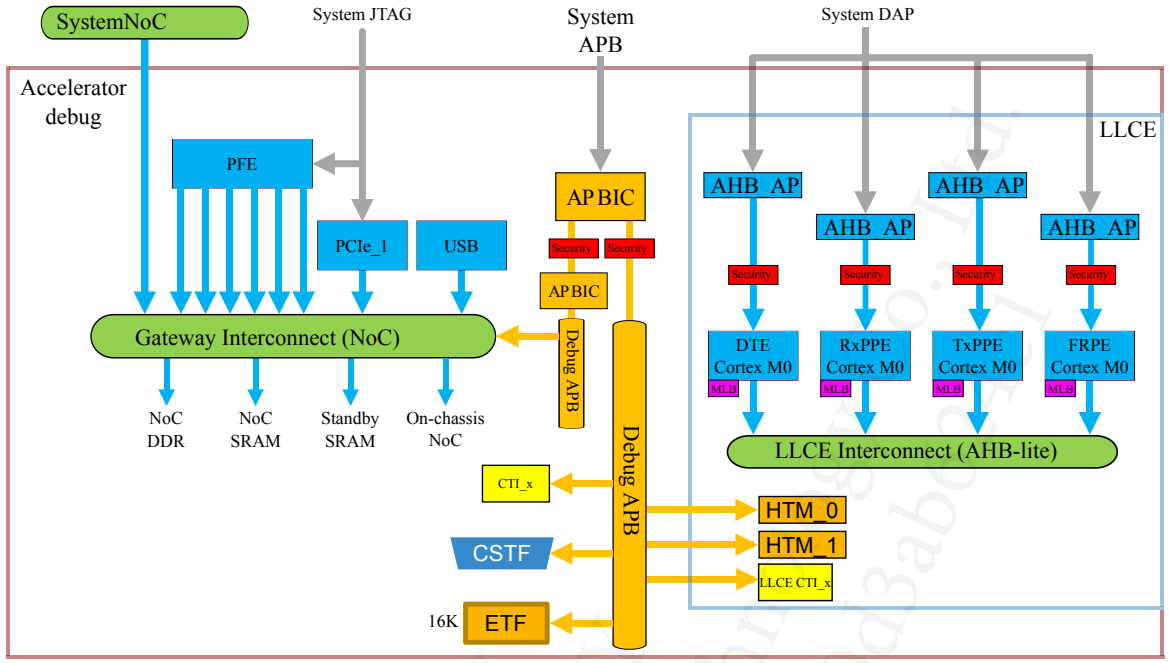


Figure 585. S32G accelerator debug block diagram

75.3 Glossary

This section provides a list of terms used in this chapter.

Table 604. Glossary and acronyms

Term	Definition
ACTIVE_HIGH	Names for signals that are active-high. These are shown in uppercase without an overbar. Signals that are active-high. These are referred as asserted when they are logic 1 and negated when they are logic 0.
ACTIVE_LOW	A bar over a signal name indicates that the signal is active-low. Active-low signals are referred as asserted when they are logic 0 and negated when they are logic 1.
AHB	System Interconnect
APBIC	System Debug APB Interconnect (APBIC)
Asserted	A signal that is asserted in its active state. An active-low signal changes from logic level one to logic level zero when asserted and an active-high signal changes from logic level zero to logic level one.
Clear	To clear a bit or bits means to establish logic level zero on the bit or bits
Customer	The end user of an SoC design or chip
DAP	Debug Access Port
Double word	Two words, or 64-bits
ECT	Embedded Cross Trigger
ETF	Embedded CoreSight Funnels

Table continues on the next page...

Table 604. Glossary and acronyms (continued)

Term	Definition
JDC	JTAG Data Communication
JTAGC	JTAG Controller
Logic level one	The voltage corresponding to boolean set to true (1)
Logic level zero	The voltage corresponding to boolean set to false (0)
MDM_AP	Miscellaneous Debug Module Access Port
MIC	Manufacturer identity code
Negated	A signal that is negated in its inactive state. An active-low signal changes from logic level zero to logic level one when negated and an active-high signal changes from logic level one to logic level zero.
PIN	Part identification number
Set	To set a bit or bits means to establish logic level one on the bit or bits
SIP	System-In-a-Package
TAP	Test and Debug Access Port
TPIU	Trace Port Interface Unit

75.4 Features

The debug and trace features are as follows:

- Debug control
  - Via IEEE 1149.1 compliant JTAG interface
- Test control
  - Via IEEE 1149.1 compliant JTAG interface
  - Uses IEEE 1149.6 extension to IEEE 1149.1
- Trace interfaces
  - Aurora-based 10-pin Trace port interface
    - Four (8-pin) LVDS trace ports output
    - 2-pin LVDS reference clock input (AURORA_EXT_CLK)
  - Capability to trace to internal system RAM via Arm ETR interface
- Debug security
  - JTAG/DAP interface that controls all debug features and is gated by the HSE using the HSE JDC (JTAG Data Communication) module
  - Security modes as described in the security chapter
- Debugging and run control
  - Stop, start, break/watchpoints
  - Each Cortex A53 supports full six hardware breakpoints and four watchpoints
  - Each Cortex M7 supports eight instruction comparators and a watchpoint unit supporting four watchpoints
- Trace

Provided under NDA on behalf of NXP
COMPANY PROPRIETARY
Beijing CATI Automotive Technology Co., Ltd.
8c462e5-491-48a8-b251-390d5ab64d1

  - Trace data routed through funnels and replicators and output via one of following output streams:
    - Four lanes LVDS at 2.5 GHz to 5 GHz High Speed Serial Trace Port (HSSTP)
    - ETR to route data to system memory—this module connects to a port on NoC and allows writing data to system RAM or external DRAM
- Trace source
  - Cortex A53
    - ETM providing instruction trace
    - ITM, time-stamping and diagnostic information

NOTE

A53 ITM does not support global timestamp but a 21 bit local timestamp.

    - 16 KB + 16 KB local ETF RAM (8 KB for each A53 core)
  - Cortex M7
    - ETM-M7 providing instruction and data trace
    - ITM providing DWT, time-stamping, and diagnostic information
    - 8 KB local ETF RAM per M7 core
    - Trace output through the TPIU to the Aurora Trace Port, with 8 KB local ETF RAM per M7 core
  - NoC
    - Provides visibility of data transactions triggered by CPUs that do not support data trace and non-CPU masters
    - Provides trace capability of accesses to DDR memory
- Cross-triggering
  - Controls run-control of cores based on other cores (for example, breakpoint reached on Cortex-M7 can optionally halt execution of Cortex-A53 and vice versa)
  - Matrix includes connections to
    - All Cortex A53 CPUs
    - All Cortex M7 CPUs
    - NoC trace sources
- Ability to view and modify all memory-mapped areas that are not otherwise blocked
  - System bus debug access port
- Performance monitoring of cores
  - PMU implemented on each Cortex A53 core
- Safety
  - Supports monitoring of debug signals to avoid common mode faults
  - Allows checking of erroneous activation of debugging (especially if intrusive, for example a CPU entering debug state)
- Timestamps
  - Generates timestamp bus for distribution to the trace sources
  - 48-bit binary timestamp bus
  - Clocked by XBAR_CLK

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48a8-b251-390d3ab624c1

75.5 Debug

75.5.1 Test and Debug Access Port (TAP) connectivity

This device has two JTAG controllers sharing the common JTAG port of the device, Arm DAP's JTAG-DP and System JTAG Controller JTAGC. Both JTAG-DP and JTAGC have an IR length of eight and are connected to the JTAG port in an overlay scheme. The System JTAG Controller, JTAGC, also supports the device's test interface.

75.5.2 Debug Access Port (DAP) TAP

The Debug Access Port (DAP) is a standard Arm component. DAP provides multiple master driving ports, all accessible and controlled through a single external interface port, to provide system-wide debug.

DAP Instruction Register (DAP IR) overlays with the system JTAG controller IR (JTAGC IR). DAP instructions are listed in the following table. DAP uses five instructions, but because BYPASS is identical to the JTAGC BYPASS, it is not shown.

Table 605. Arm DAP IR codes

Code	DAP IR
1111_1000b	ABORT
1111_1010b	DPACC
1111_1011b	APACC
1111_1110b	IDCODE

DAP offers AXI, AHB, and APB master interfaces to access system buses. It also exports the internal DAP bus to extend the access ports. For more information on DAP TAP, see the Arm Debug Interface Architecture Specification (available in References) for details.

AXI AP provides the debugger access to all memory and registers in the system memory map. The XRDC controls system access. The APB_AP allows access to all CoreSight components except Cortex-M7 components.

A core can access the debug components of another core through a DAPMUX to the bus interconnect.

The exported DAP bus hosts AHB_AP (part of Cortex-M7) and the Miscellaneous Debug Module Access Port (MDM_AP). The MDM_AP hosts system level JTAG status and control registers (see MDM_AP register descriptions) which can be used for cross-triggering, synchronized debug, and other miscellaneous control and status functions.

For accessing any APB-mapped Debug modules, APB_AP uses an APSEL value of 1h. Table 609 and Table 610 describe the access addresses of APB-mapped Debug modules.

The value of DAP SELECT[APSEL], the SELECT register of SWJ-DP, selects the access ports in the DAP. APSEL decoding is shown in the following table.

Table 606. DAP APSEL decode

APSEL (SELECT[31:24])	APBIC base address (access from debugger)	System memory address (access from cores)	Selection	Identification register (IDR) value
00h	0000_0000h	51D0_0000h	AXI_AP to system bus	4477_0004h
01h	0100_0000h	51D0_0100h	APB_AP to all APB mapped Debug modules (for example Cortex-A53, watchpoints, trace funnels)	5477_0002h

Table continues on the next page...

Table 606. DAP APSEL decode (continued)

APSEL (SELECT[31:24])	APBIC base address (access from debugger)	System memory address (access from cores)	Selection	Identification register (IDR) value
02h	0200_0000h	51D0_0200h	Reserved	8477_0001h
03h	0300_0000h	51D0_0300h	AHB_AP0 to Cortex-M7_0 Debug modules	8477_0001h
04h	0400_0000h	51D0_0400h	AHB_AP1 to Cortex-M7_1 Debug modules	8477_0001h
05h	0500_0000h	51D0_0500h	AHB_AP2 to Cortex-M7_2 Debug modules	8477_0001h
06h	0600_0000h	51D0_0600h	MDM_AP Miscellaneous Debug module AP	001C_0030h
07h	0700_0000h	51D0_0700h	Reserved	8477_0001h
08h	0800_0000	51D0_0800h	AHB_AP for LLCE M0+ Core0 Debug modules	001C_0030h
09h	0900_0000h	51D0_0900h	AHB_AP for LLCE M0+ Core1 Debug modules	001C_0030h
0Ah (10d)	0A00_0000h	51D0_0A00h	AHB_AP for LLCE M0+ Core2 Debug modules	001C_0030h
0Bh (11d)	0B00_0000h	51D0_0B00h	AHB_AP for LLCE M0+ Core3 Debug modules	001C_0030h
0Ch—FFh	—	—	Reserved (default AP response)	—

75.5.3 System JTAG Controller (JTAGC)

The JTAGC connects in parallel with the Arm TAP controller (JTAG DP of Arm DAP). The IR length is 8-bits. The JTAGC IR codes overlay the Arm DAP controller IR codes. The DAP uses four instructions and the remaining are used by JTAGC. The outputs of the TAPs (TDO) are multiplexed based on the IR code selected. This chip is fully JTAG-compliant and appears to the JTAG chain as a single TAP.

The following table gives the IR codes for the JTAGC. The instructions used by Arm DAP TAP are shown in Table 605.

Table 607. JTAG instructions for JTAGC

Code	JTAGC IR
0000_0000b	IDCODE
0000_0001b	CENSOR_CTRL
0000_0010b	SAMPLE-PRELOAD
0000_0011b	SAMPLE
0000_0100b	EXTEST
0000_0101b	HI-Z
0000_0110b	Reserved
0000_0111b	Reserved

Table continues on the next page...

Table 607. JTAG instructions for JTAGC (continued)

Code	JTAGC IR
0000_1000b	Reserved
0000_1001b	Reserved
0000_1010b—0000_1011b	Reserved
0000_1100b	CLAMP
0000_1101b—0000_1111b	Reserved
0001_0000b	ENABLE_SOC_DATA1
0001_0001b—0111_1111b	Reserved
1000_0000b	Reserved
1000_0001b	Reserved
1000_0010b	Reserved
1000_0011b	Reserved
1000_0100b	PCIe_0
1000_0101b	Reserved
1000_0110b—1000_0111b	Reserved
1000_1000b	DDR_0
1000_1001b	Reserved
1000_1010b—1000_1100b	Reserved
1000_1101b	Reserved
1000_1110b	Reserved
1000_1111b	Reserved
1001_0000b	Security JDC
1001_0001b	System JDC
1001_0010b—1001_0111b	Reserved for other System Auxiliary clients
1001_1000b—1001_1011b	Reserved
1001_1100b—1011_1111b	Reserved for other System Auxiliary clients
1100_0000b	PFE
1100_0001b	PCIe_1
1100_0010b—1110_1111b	Reserved

Table continues on the next page...

Table 607. JTAG instructions for JTAGC (continued)

Code	JTAGC IR
1111_1000b	ABORT (Arm)
1111_1001b	Reserved
1111_1010b	DPACC (Arm)
1111_1011b	APACC (Arm)
1111_1100b	Reserved
1111_1101b	Reserved
1111_1110b	IDCODE (Arm)
1111_1111b	BYPASS

75.5.3.1 Chip JTAG ID

Each chip in the S32 family includes a unique JTAG ID. This includes changing of the JTAG ID when instantiated with different die in a System-In-a-Package (SIP). This following table shows the JTAGC ID for this chip.

Table 608. JTAG ID

Chip	Part revision number (PRN)	Design center (DC)	Part identification number (PIN)	Manufacturer identity code (MIC)	IDCODE ID	JTAG ID
S32G275A	0	20h (32d)	301h (769d)	Eh (14d)	1	1830_101Dh

75.5.3.2 JTAG Data Communication (JDC) module

JDC allows access to two 32-bit data registers by JTAG interface and by software running on one of the CPUs in the chip. These registers exchange data between an internal CPU and an external debug tool.

75.6 APB memory map

The debug registers are accessed via the APB-AP bus. The table below shows all the addresses for the CoreSight APB components. In addition, all APB registers are accessible from the processing cores (assuming access permission is enabled). This table also shows the addressing for accessing the DAP components via the memory interface.

Arm CoreSight APB_AP ROM auto-discovery method cannot be used for identifying the APB components in the system since some components, specifically the NoC does not provide the standard CoreSight component and peripheral ID register. For details, see following table.

NoC debug registers are reset by a destructive reset. However, they cannot be accessed while functional reset is asserted. In addition, trace packets cannot be accessed during functional reset.

Table 609. APB component mapping

Debug APB component		APBIC base address (access from debugger)	System memory map base address (access from cores)	Slot size	APB_AP0 slot number
	APB_AP ROM Table	F8h	510000F8h	4 KB	APBIC base
Instrumentation Trace Macrocells	A53 Cluster0 Core0 ITM	00001000h	51001000h	4 KB	APB Master0
	A53 Cluster0 Core1 ITM	00002000h	51002000h	4 KB	APB Master1
	A53 Cluster1 Core0 ITM	00003000h	51003000h	4 KB	APB Master2
	A53 Cluster1 Core1 ITM	00004000h	51004000h	4 KB	APB Master3
Reserved	Reserved	00005000h	51005000h	4 KB	APB Master4
	Reserved	00006000h	51006000h	4 KB	APB Master5
	Reserved	00007000h	51007000h	4 KB	APB Master6
	Reserved	00008000h	51008000h	4 KB	APB Master7
Trace funnels	Funnel0	00009000h	51009000h	4 KB	APB Master8
	Funnel1	0000A000h	5100A000h	4 KB	APB Master9
	Funnel2	0000B000h	5100B000h	4 KB	APB Master10
	Funnel3	0000C000h	5100C000h	4 KB	APB Master11
	Funnel4	0000D000h	5100D000h	4 KB	APB Master12
	Funnel5	0000E000h	5100E000h	4 KB	APB Master13
	Funnel6	0000F000h	5100F000h	4 KB	APB Master14
Reserved	Reserved	00010000h	51010000h	4 KB	APB Master15
	Reserved	00011000h	51011000h	4 KB	APB Master16
	Reserved	00012000h	51012000h	4 KB	APB Master17
	Reserved	00013000h	51013000h	4 KB	APB Master18
Embedded Trace FIFO	ETF0	00014000h	51014000h	4 KB	APB Master19
	ETF1	00015000h	51015000h	4 KB	APB Master20
	ETF2	00016000h	51016000h	4 KB	APB Master21
	ETF3	00017000h	51017000h	4 KB	APB Master22
	ETF4	00018000h	51018000h	4 KB	APB Master23
	ETF5	00019000h	51019000h	4 KB	APB Master24

Table continues on the next page...

Table 609. APB component mapping (continued)

Debug APB component		APBIC base address (access from debugger)	System memory map base address (access from cores)	Slot size	APB_AP0 slot number
Reserved	Reserved	0001A000h	5101A000h	4 KB	APB Master25
	Reserved	0001B000h	5101B000h	4 KB	APB Master26
	Reserved	0001C000h	5101C000h	4 KB	APB Master27
	Reserved	0001D000h	5101D000h	4 KB	APB Master28
	Reserved	0001E000h	5101E000h	4 KB	APB Master29
	Reserved	0001F000h	5101F000h	4 KB	APB Master30
	Reserved	00020000h	51020000h	4 KB	APB Master31
	Reserved	00021000h	51021000h	4 KB	APB Master32
	Reserved	00022000h	51022000h	4 KB	APB Master33
	Reserved	00023000h	51023000h	4 KB	APB Master34
	Reserved	00024000h	51024000h	4 KB	APB Master35
	Reserved	00025000h	51025000h	4 KB	APB Master36
	Reserved	00026000h	51026000h	4 KB	APB Master37
Cross Trigger Matrix	CTI_NoC0	00027000h	51027000h	4 KB	APB Master38
	CTI_NoC1	00028000h	51028000h	4 KB	APB Master39
	CTI_NoC2	00029000h	51029000h	4 KB	APB Master40
	CTI_NoC3	0002A000h	5102A000h	4 KB	APB Master41
	CTI_NoC4	0002B000h	5102B000h	4 KB	APB Master42
	CTI_NoC5	0002C000h	5102C000h	4 KB	APB Master43
	CTI_NoC6	0002D000h	5102D000h	4 KB	APB Master44
	CTI_NoC7	0002E000h	5102E000h	4 KB	APB Master45
	Reserved	0002F000h	5102F000h	4 KB	APB Master46
CoreSight output control	ETR	00030000h	51030000h	4 KB	APB Master47
	Reserved	00031000h	51031000h	4 KB	APB Master48
	TPIU	00032000h	51032000h	4 KB	APB Master49
	ATP	00033000h	51033000h	4 KB	APB Master50
	Reserved	00034000h	51034000h	4 KB	APB Master51
Reserved	Reserved	00035000h	51035000h	4 KB	APB Master52
	Reserved	00036000h	51036000h	4 KB	APB Master53

Table continues on the next page...

Table 609. APB component mapping (continued)

Debug APB component		APBIC base address (access from debugger)	System memory map base address (access from cores)	Slot size	APB_AP0 slot number
	Reserved	00037000h	51037000h	4 KB	APB Master54
	Reserved	00038000h	51038000h	32 KB	APB Master55
	Reserved	00039000h	51039000h	4 KB	APB Master56
	Reserved	0003A000h	5103A000h	4 KB	APB Master57
	Reserved	Reserved	Reserved	Reserved	Reserved
Off-Platform Debug	ROM Table	00050000h	51050000h	4 KB	APB master59
	CTI_ETF_ETR0	00051000h	51051000h	4 KB	
	CTI_ETF_ETR1	00052000h	51052000h	4 KB	
	Timestamp Debug	00053000h	51053000h	4 KB	
	CTI_2	0005_4000h	5105_4000h	4 KB	
Reserved for accelerator subsystem[1]		00100000h	51100000h	1 MB	APB master60
A53 Cluster0	ROM Table	00400000h	51400000h	4 MB	APB Master61
	Core0 Debug	00410000h	51410000h		
	Core0 CTI	00420000h	51420000h		
	Core0 PMU	00430000h	51430000h		
	Core0 ETM	00440000h	51440000h		
	Core1 Debug	00510000h	51510000h		
	Core1 CTI	00520000h	51520000h		
	Core1 PMU	00530000h	51530000h		
	Core1 ETM	00540000h	51540000h		
A53 Cluster1	ROM Table	00800000h	51800000h	4 MB	APB Master62
	Core0 Debug	00810000h	51810000h		
	Core0 CTI	00820000h	51820000h		
	Core0 PMU	00830000h	51830000h		
	Core0 ETM	00840000h	51840000h		
	Core1 Debug	00910000h	51910000h		
	Core1 CTI	00920000h	51920000h		

Table continues on the next page...

Table 609. APB component mapping (continued)

Debug APB component		APBIC base address (access from debugger)	System memory map base address (access from cores)	Slot size	APB_AP0 slot number	
	Core1 PMU	00930000h	51930000h			
	Core1 ETM	00940000h	51940000h			
NoC_0 (On-Chassis)	NoC ROM Table	00C40000h	51C40000h	1 MB	APB Master63	Sub APB Master
	a53_cluster0_noncoh_rsp_packet_main_Probe	00C41000h	51C41000h			0
	a53_cluster0_noncoh_xaction_main_Probe	00C42000h	51C42000h			1
	a53_cluster0_noncoh_xaction_main_TransactionStatProfiler	00C42400h	51C42400h			1
	a53_cluster1_noncoh_req_packet_main_Probe	00C43000h	51C43000h			2
	a53_cluster1_noncoh_rsp_packet_main_Probe	00C44000h	51C44000h			3
	a53_cluster1_noncoh_xaction_main_Probe	00C45000h	51C45000h			4
	a53_cluster1_noncoh_xaction_main_TransactionStatProfiler	00C45400h	51C45400h			4
	concerto_req_packet_main_Probe	00C46000h	51C46000h			5
	concerto_rsp_packet_main_Probe	00C47000h	51C47000h			6
	concerto_xaction_main_Probe	00C48000h	51C48000h			7
	concerto_xaction_main_TransactionStatProfiler	00C48400h	51C48400h			7

Table continues on the next page...

Table 609. APB component mapping (continued)

Debug APB component		APBIC base address (access from debugger)	System memory map base address (access from cores)	Slot size	APB_AP0 slot number	
	dram0_a53_req_packet_main_Probe	00C49000h	51C49000h			8
	dram0_a53_rsp_packet_main_Probe	00C4a000h	51C4a000h			9
	dram0_m7_req_packet_main_Probe	00C4b000h	51C4b000h			10
	dram0_m7_rsp_packet_main_Probe	00C4c000h	51C4c000h			11
	dram0_other_req_packet_main_Probe	00C4d000h	51C4d000h			12
	dram0_other_rsp_packet_main_Probe	00C4e000h	51C4e000h			13
	edma0_req_packet_main_Probe	00C4f000h	51C4f000h			14
	edma0_rsp_packet_main_Probe	00C50000h	51C50000h			15
	edma0_xaction_main_Probe	00C51000h	51C51000h			16
	edma0_xaction_main_TransactionStatProfiler	00C51400h	51C51400h			16
	edma1_req_packet_main_Probe	00C52000h	51C52000h			17
	edma1_rsp_packet_main_Probe	00C53000h	51C53000h			18
	edma1_xaction_main_Probe	00C54000h	51C54000h			19
	edma1_xaction_main_TransactionStatProfiler	00C54400h	51C54400h			19
	enet_req_packet_main_Probe	00C55000h	51C55000h			20

Table continues on the next page...

Table 609. APB component mapping (continued)

Debug APB component		APBIC base address (access from debugger)	System memory map base address (access from cores)	Slot size	APB_AP0 slot number	
	enet_rsp_packet_main_Probe	00C56000h	51C56000h			21
	enet_xaction_main_Probe	00C57000h	51C57000h			22
	enet_xaction_main_TransactionStatProfiler	00C57400h	51C57400h			22
	fray_req_packet_main_Probe	00C58000h	51C58000h			23
	fray_rsp_packet_main_Probe	00C59000h	51C59000h			24
	fray_xaction_main_Probe	00C5a000h	51C5a000h			25
	fray_xaction_main_TransactionStatProfiler	00C5a400h	51C5a400h			25
	pcie_req_packet_main_Probe	00C5b000h	51C5b000h			26
	pcie_rsp_packet_main_Probe	00C5c000h	51C5c000h			27
	pcie_xaction_main_Probe	00C5d000h	51C5d000h			28
	pcie_xaction_main_TransactionStatProfiler	00C5d400h	51C5d400h			28
	a53_cluster0_noncoh_req_packet_main_Probe	00C5e000h	51C5e000h			29
	a53_cluster0_noncoh_observer_main_AtbEndPoint	00C70000h	51C70000h			47
	a53_cluster0_noncoh_observer_main_STPv2Converter	00C70080h	51C70080h			47
	a53_cluster1_noncoh_observer	00C71000h	51C71000h			48

Table continues on the next page...

Table 609. APB component mapping (continued)

Debug APB component		APBIC base address (access from debugger)	System memory map base address (access from cores)	Slot size	APB_AP0 slot number	
	_main_AtbEndPoint					
	a53_cluster1_noncoh_observer_main_STPv2Converter	00C71080h	51C71080h			48
	concerto_observer_main_AtbEndPoint	00C72000h	51C72000h			49
	concerto_observer_main_ErrorLogger_0	00C72080h	51C72080h			49
	concerto_observer_main_STPv2Converter	00C72100h	51C72100h			49
	dram0_observer_main_AtbEndPoint	00C73000h	51C73000h			50
	dram0_observer_main_STPv2Converter	00C73080h	51C73080h			50
	edma0_observer_main_AtbEndPoint	00C74000h	51C74000h			51
	edma0_observer_main_ErrorLogger_0	00C74080h	51C74080h			51
	edma0_observer_main_ErrorLogger_1	00C74100h	51C74100h			51
	edma0_observer_main_ErrorLogger_2	00C74180h	51C74180h			51
	edma0_observer_main_STPv2Converter	00C74200h	51C74200h			51
	edma1_observer_main_AtbEndPoint	00C75000h	51C75000h			52

Table continues on the next page...

Table 609. APB component mapping (continued)

Debug APB component		APBIC base address (access from debugger)	System memory map base address (access from cores)	Slot size	APB_AP0 slot number	
	edma1_observer_main_ErrorLogger_0	00C75080h	51C75080h			52
	edma1_observer_main_ErrorLogger_1	00C75100h	51C75100h			52
	edma1_observer_main_ErrorLogger_2	00C75180h	51C75180h			52
	edma1_observer_main_STPv2Converter	00C75200h	51C75200h			52
	enet_observer_main_AtbEndPoint	00C76000h	51C76000h			53
	enet_observer_main_STPv2Converter	00C76080h	51C76080h			53
	fray_observer_main_AtbEndPoint	00C77000h	51C77000h			54
	fray_observer_main_ErrorLogger_0	00C77080h	51C77080h			54
	fray_observer_main_STPv2Converter	00C77100h	51C77100h			54
	pcie_observer_main_AtbEndPoint	00C78000h	51C78000h			55
	pcie_observer_main_STPv2Converter	00C78080h	51C78080h			55

1. See the Accelerator subsection (if required).

For debug access, subtract 5100_0000h from the CPU system access register memory map addresses.

75.6.1 S32G accelerator debug APB memory map

This section describes the debug features implemented in the Accelerator portion of the MCU. The Accelerator consists of the Low Latency Communication Engine (LLCE), a second PCIe_1, and an Ethernet packet Forwarding Engine (PFE). The table below describes the address map of the additional modules.

Table 610. S32G accelerator APB component mapping

Debug component	APBIC base address (access from debugger)	System memory map base address[1]	Memory allocation	APB_AP0 slot number
Accelerator debug base ROM table	00100000h	51100000h	1 M	
16KB ETF	00101000h	51101000h	4 K	
NoC CTI0	00102000h	51102000h	4 K	
NoC CTI1	00103000h	51103000h	4 K	
NoC/Misc CTI2	00104000h	51104000h	4 K	
Trace funnel	00105000h	51105000h	4 K	
LLCE HTM0	00106000h	51106000h	4 K	
LLCE HTM1	00107000h	51107000h	4 K	
LLCE CTI0	00108000h	51108000h	4 K	
Reserved	00109000h	51109000h	28 K	
NoC APBIC ROM table	00110000h	51110000h	64 K	
PCIe1 request probe	00111000h	51111000h	4 K	
PCIe1 response probe	00112000h	51112000h	4 K	
PFE DDR I/F request probe	00113000h	51113000h	4 K	
PFE DDR I/F response probe	00114000h	51114000h	4 K	
PFE HIF0/HIF2 request probe	00115000h	51115000h	4 K	
PFE HIF0/HIF2 response probe	00116000h	51116000h	4 K	
PFE HIF1/HIF3/Util request probe	00117000h	51117000h	4 K	
PFE HIF1/HIF3/Util response probe	00118000h	51118000h	4 K	
USB request/response shared probe	00119000h	51119000h	4 K	
PCIe1 observer (services USB, PFE DDR and PCIe1 probes)	0011A000h	5111A000h	4 K	
PFE observer (services PFE HIF0-3 probes)	0011B000h	5111B000h	4 K	
LLCE M0+ Core0 debug access port	—	51D0_0800h		DAP master #8
LLCE M0+ Core1 debug access port	—	51D0_0900h		DAP master #9
LLCE M0+ Core2 debug access port	—	51D0_0A00h		DAP master #10
LLCE M0+ Core3 debug access port	—	51D0_0B00h		DAP master #11

1. Access from cores.

75.7 Trace

75.7.1 Trace modules and connectivity

The trace subsystem combines trace data from all internal clients that generate trace information. The trace subsystem has a 32-bit TPIU.

The following figure shows the trace architecture.

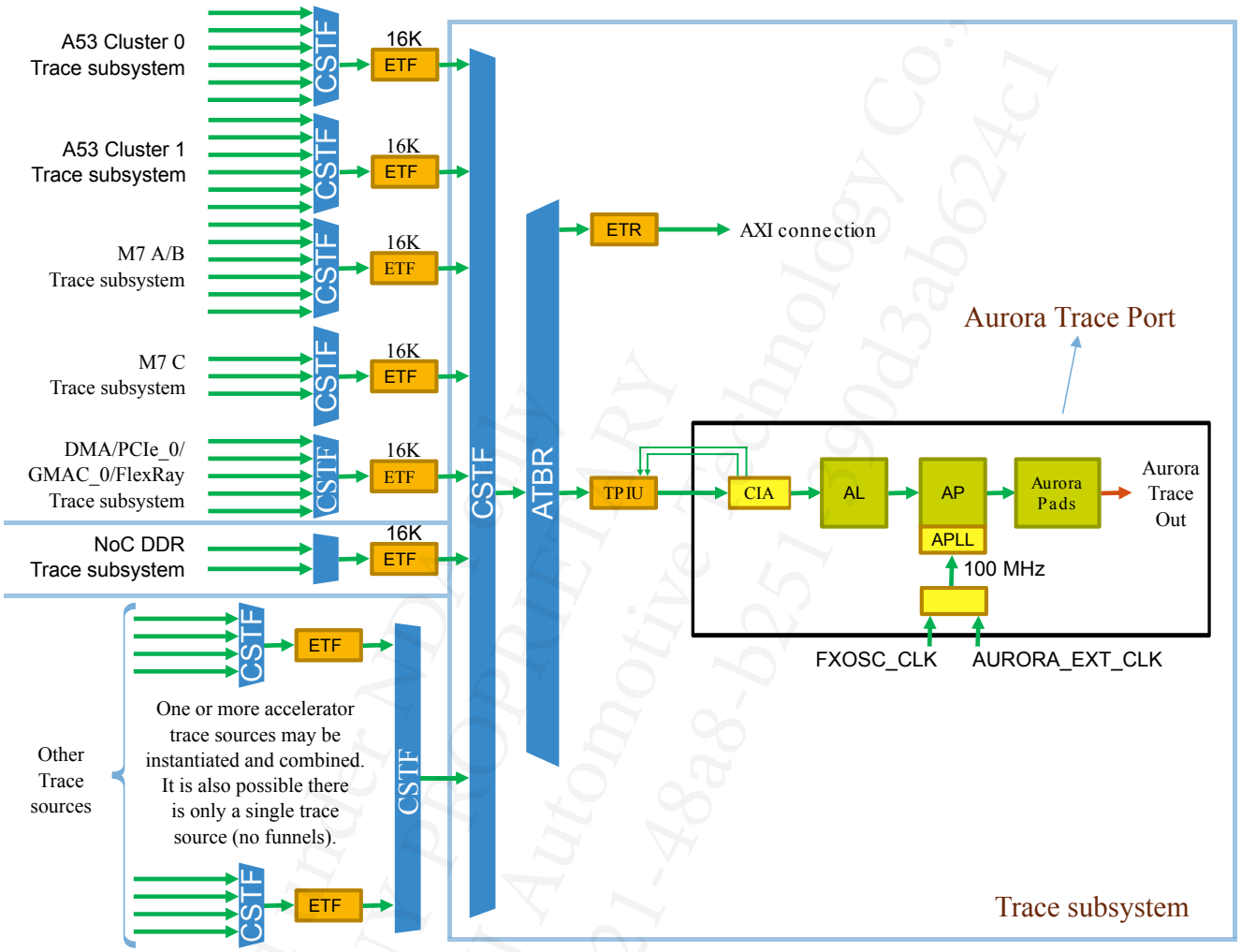


Figure 586. Trace bus connectivity

Table 611. Trace subsystem components

Component	Name
APLL	Aurora PLL
ATB	Advanced Trace Bus
ATBR	ATB Replicator
CIA	CoreSight Interface Adapter (CoreSight to Aurora Interface)
CSTF	CoreSight Trace Funnel

Table continues on the next page...

Table 611. Trace subsystem components (continued)

Component	Name
Debug APB	Debug Advanced Peripheral Bus
ETF	Embedded Trace FIFO
ETR	Embedded Trace Router
AL	Trace Aurora Link
AP	Trace Aurora Physical Interface
TPIU	Trace Port Interface Unit

Multiple options for trace output allow parallel tracing. The trace information can be read from the trace interface or the DAP interface. The traces can be alternatively read out from ETF at slow speeds via APB AP. Additionally, the trace can be written into system RAM via the ETR and you can read it later.

The sizes of the ETFs are shown in the following table.

Table 612. ETF sizes

ETF	Size
ETF 0	16 Kbytes
ETF1	16 Kbytes
ETF2	16 Kbytes
ETF3	16 Kbytes
ETF4	16 Kbytes
ETF5	16 Kbytes

The MDM_AP Control register implements bits to over-ride the speed control from some of the trace sinks (ETR, and TPIU). The complete trace pipeline bandwidth is limited by the slowest sink component. The default settings of these bits allow for maximum bandwidth for the TPIU to trace to the Aurora Trace Port (ATP). When tracing to memory (if supported), these bits may need to be changed.

Table 613. Trace output overrides

Trace destination	SWO_Override	ETR_Override	TPIU_Override
TPIU/Aurora	1	1	0
ETR to memory	1	0	1

The trace sources of the chip are the cores and their related modules. Trace information can be monitored in NoC for some master and slave interfaces. Trace funnels exist for all possible trace clients. For details of the various components in the trace bus connectivity, see the Arm CoreSight Components TRM (available in References) for details.

75.7.1.1 S32G accelerator trace modules and connectivity

Accelerator modules contain several trace sources. These include the NoC observers (discussed in the next section) and the LLCE HTM0 and HTM1 modules from LLCE. Description on the HTMs and the buses they monitor can be found in the LLCE chapter of the Reference Manual. All modules are configured via the debug APB interface using the base addresses defined in the memory map.

The following diagram shows the accelerator trace architecture.

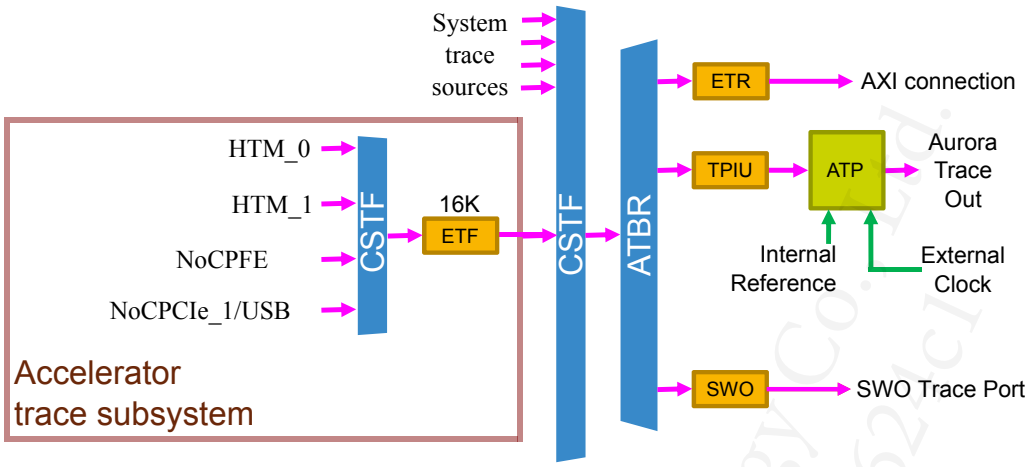


Figure 587. S32G accelerator trace architecture

75.7.2 Trace assignments

The embedded CoreSight Funnels (ETF) combine trace information from multiple clients into a single stream of trace data. The following table shows the CoreSight funnel input assignments.

Table 614. CoreSight funnel assignments

Instance name	Port assignment
Funnel0 (CSTF0)	s0: Cortex-A53a ETM0
	s1: Cortex-A53a ETM1
	s2: Cortex-A53a ITM0
	s3: Cortex-A53a ITM1
	s4: Cortex-A53 Cluster0/1 Coherent
	s5: NoC – Cortex-A53 Cluster0 Noncoherent
Funnel1 (CSTF1)	s0: Cortex-A53b ETM0
	s1: Cortex-A53b ETM1
	s2: Cortex-A53b ITM0
	s3: Cortex-A53b ITM1
	s4: NoC – Cortex-A53 Cluster1 Noncoherent
	s5: Reserved
Funnel2 (CSTF2)	s0: Cortex-M7_0 ETM Data
	s1: Cortex-M7_0 ETM Instr
	s2: Cortex-M7_0 ITM
	s3: Cortex-M7_1 ETM Data
	s4: Cortex-M7_1 ETM Instr
	s5: Cortex-M7_1 ITM

Table continues on the next page...

Table 614. CoreSight funnel assignments (continued)

Instance name	Port assignment
Funnel3 (CSTF3)	s0: Cortex-M7_2 ETM Data
	s1: Cortex-M7_2 ETM Instr
	s2: Cortex-M7_2 ITM
	s3: Reserved
	s4: Reserved
	s5: Reserved
	Reserved
Funnel4 (CSTF4)	s0: NoC – DMA0
	s1: NoC – DMA1
	s2: NoC – PCIe_0
	s3: NoC – GMAC_0
	s4: NoC – FlexRay
Funnel5 (CSTF5)	s0: NoC – DRAM0
	s1: Reserved
Funnel6 (CSTF6)	s0: Funnel0 (ETF0)
	s1: Funnel1 (ETF1)
	s2: Funnel2 (ETF2)
	s3: Funnel3 (ETF3)
	s4: Funnel4 (ETF4)
	s5: Funnel5 (ETF5)
	s6: Accelerator trace sources

75.7.2.1 S32G accelerator trace assignments

The S32G includes a single CoreSight Trace Funnel (CSTF) in the Accelerator debug circuitry. This funnel feeds an Embedded Trace FIFO (ETF) that connects to the system trace funnel. The trace clients are connected as shown in the following table.

Table 615. S32G accelerator trace funnels assignments

Instance name	Port assignment
ACCR_CSTF	s0: PCIe_1 NoC observer (PCIe_1 and USB)
	s1: PFE NoC observer
	s2: HTM32_0
	s3: HTM32_1

Provided by IHS Markit
COMPANY CONFIDENTIAL
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482a-4aa8-b251-390d3ab624c1

75.8 Embedded Cross Trigger (ECT)

ECT allows for multicore run-control and trace cross-triggering, such as synchronous stop-start for all cores or trigger trace on a trigger event from another core or IP. See the *CoreSight Components Technical Reference Manual* (available in References) for detailed information on ECT.

ECT architecture involves CTMs and CTIs. The CTIs provide cross-triggering interface between the cores and other debug and trace modules. The channels of these CTIs are interconnected using CTMs, as shown in the following figure.

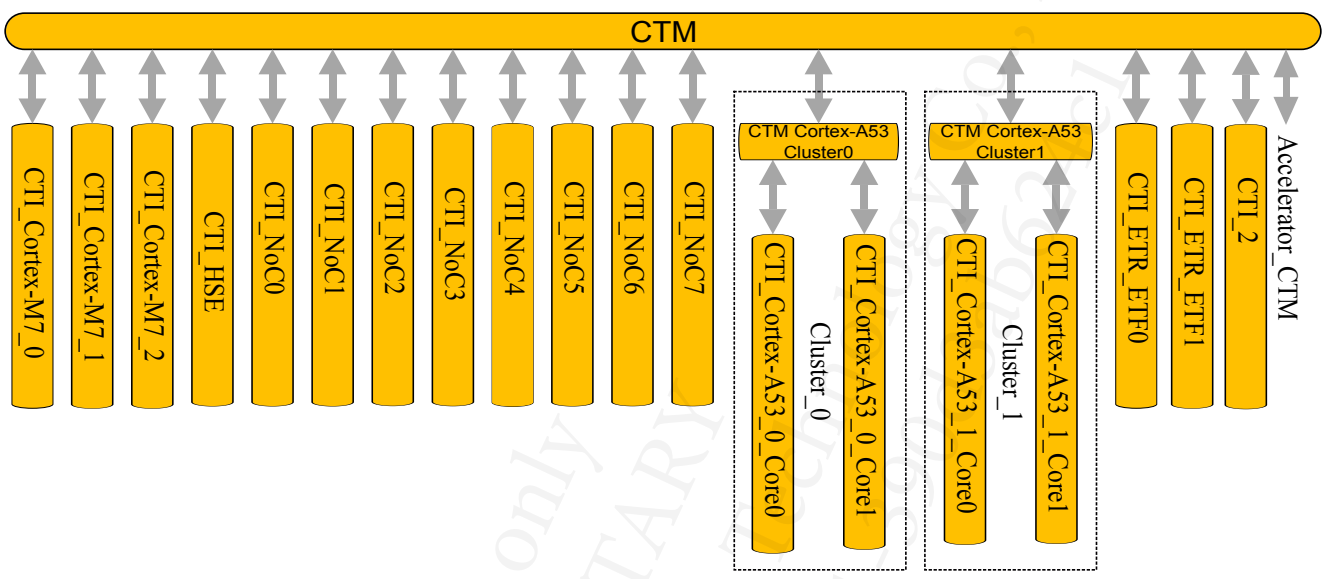


Figure 588. ECT

75.8.1 CTI assignments

The following table shows the connections of the CTI channels.

Table 616. CTI assignments

CTI instance	Trigger number	Trigger in	Trigger out
CTI_Cortex-M7_0	7	Cortex-M7_0 ETM event output3	Cortex-M7_0 processor restart
	6	Cortex-M7_0 ETM event output2	Cortex-M7_0 ETM event input3
	5	Cortex-M7_0 ETM event output1	Cortex-M7_0 ETM event input2
	4	Cortex-M7_0 ETM event output0	Cortex-M7_0 ETM event input1
	3	Cortex-M7_0 DWT comparator output2	Cortex-M7_0 ETM event input0
	2	Cortex-M7_0 DWT comparator output1	Cortex-M7_0 interrupt request 1
	1	Cortex-M7_0 DWT comparator output0	Cortex-M7_0 interrupt request 0
	0	Cortex-M7_0 processor halted	Cortex-M7_0 process debug request
CTI_Cortex-M7_1	7	Cortex-M7_1 ETM event output3	Cortex-M7_1 processor restart
	6	Cortex-M7_1 ETM event output2	Cortex-M7_1 ETM event input3
	5	Cortex-M7_1 ETM event output1	Cortex-M7_1 ETM event input2

Table continues on the next page...

Table 616. CTI assignments (continued)

CTI instance	Trigger number	Trigger in	Trigger out
	4	Cortex-M7_1 ETM event output0	Cortex-M7_1 ETM event input1
	3	Cortex-M7_1 DWT comparator output2	Cortex-M7_1 ETM event input0
	2	Cortex-M7_1 DWT comparator output1	Cortex-M7_1 interrupt request 1
	1	Cortex-M7_1 DWT comparator output0	Cortex-M7_1 interrupt request 0
	0	Cortex-M7_1 processor halted	Cortex-M7_1 process debug request
CTI_Cortex-M7_2	7	Cortex-M7_2 ETM event output3	Cortex-M7_2 processor restart
	6	Cortex-M7_2 ETM event output2	Cortex-M7_2 ETM event input3
	5	Cortex-M7_2 ETM event output1	Cortex-M7_2 ETM event input2
	4	Cortex-M7_2 ETM event output0	Cortex-M7_2 ETM event input1
	3	Cortex-M7_2 DWT comparator output2	Cortex-M7_2 ETM event input0
	2	Cortex-M7_2 DWT comparator output1	Cortex-M7_2 interrupt request 1
	1	Cortex-M7_2 DWT comparator output0	Cortex-M7_2 interrupt request 0
	0	Cortex-M7_2 processor halted	Cortex-M7_2 process debug request
CTI_NoC0	7	ncore_probe_ctitrigout[1]	ncore_probe_ctitrigin[1]
	6	ncore_probe_ctitrigout[0]	Reserved
	5	a53_cluster0_noncoh_probe_ctitrigout[1]	a53_cluster0_noncoh_probe_ctitrigin[1]
	4	a53_cluster0_noncoh_probe_ctitrigout[0]	Reserved
	3	a53_cluster1_noncoh_probe_ctitrigout[1]	a53_cluster1_noncoh_probe_ctitrigin[1]
	2	a53_cluster1_noncoh_probe_ctitrigout[0]	Reserved
	1	pcie0_probe_ctitrigout[1]	pcie0_probe_ctitrigin[1]
	0	pcie0_probe_ctitrigout[0]	Reserved
CTI_NoC1	7	edma0_probe_ctitrigout[1]	edma0_probe_ctitrigin[1]
	6	edma0_probe_ctitrigout[0]	Reserved
	5	edma1_probe_ctitrigout[1]	edma1_probe_ctitrigin[1]
	4	edma1_probe_ctitrigout[0]	Reserved
	3	gmac0_probe_ctitrigout[1]	gmac0_probe_ctitrigin[1]
	2	gmac0_probe_ctitrigout[0]	Reserved
	1	fray_probe_ctitrigout[1]	fray_probe_ctitrigin[1]
	0	fray_probe_ctitrigout[0]	Reserved
CTI_NoC2	7	dram0_a53_probe_ctitrigout[1]	dram0_a53_probe_ctitrigin[1]
	6	dram0_a53_probe_ctitrigout[0]	Reserved
	5	dram0_cm7_probe_ctitrigout[1]	dram0_cm7_probe_ctitrigin[1]

Table continues on the next page...

Table 616. CTI assignments (continued)

CTI instance	Trigger number	Trigger in	Trigger out
	4	dram0_cm7_probe_ctitrigout[0]	Reserved
	3	dram0_other_probe_ctitrigout[1]	dram0_other_probe_ctitrigin[1]
	2	dram0_other_probe_ctitrigout[0]	Reserved
	1	Reserved	Reserved
	0	Reserved	Reserved
CTI_NoC3	7	Reserved	Reserved
	6	Reserved	Reserved
	5	Reserved	Reserved
	4	Reserved	Reserved
	3	Reserved	Reserved
	2	Reserved	Reserved
	1	Reserved	Reserved
	0	Reserved	Reserved
CTI_NoC4	7	ncore_rsp_probe_ctitrigout[1]	ncore_rsp_probe_ctitrigin[1]
	6	ncore_rsp_probe_ctitrigout[0]	Reserved
	5	a53_cluster0_noncoh_rsp_probe_ctitrigout[1]	a53_cluster0_noncoh_rsp_probe_ctitrigin[1]
	4	a53_cluster0_noncoh_rsp_probe_ctitrigout[0]	Reserved
	3	a53_cluster1_noncoh_rsp_probe_ctitrigout[1]	a53_cluster1_noncoh_rsp_probe_ctitrigin[1]
	2	a53_cluster1_noncoh_rsp_probe_ctitrigout[0]	Reserved
	1	pcie0_rsp_probe_ctitrigout[1]	pcie0_rsp_probe_ctitrigin[1]
	0	pcie0_rsp_probe_ctitrigout[0]	Reserved
CTI_NoC5	7	edma0_rsp_probe_ctitrigout[1]	edma0_rsp_probe_ctitrigin[1]
	6	edma0_rsp_probe_ctitrigout[0]	Reserved
	5	edma1_rsp_probe_ctitrigout[1]	edma1_rsp_probe_ctitrigin[1]
	4	edma1_rsp_probe_ctitrigout[0]	Reserved
	3	gmac0_rsp_probe_ctitrigout[1]	gmac0_rsp_probe_ctitrigin[1]
	2	gmac0_rsp_probe_ctitrigout[0]	Reserved
	1	fray_rsp_probe_ctitrigout[1]	fray_rsp_probe_ctitrigin[1]
	0	fray_rsp_probe_ctitrigout[0]	Reserved
CTI_NoC6	7	dram0_a53_rsp_probe_ctitrigout[1]	dram0_a53_rsp_probe_ctitrigin[1]
	6	dram0_a53_rsp_probe_ctitrigout[0]	Reserved
	5	dram0_cm7_rsp_probe_ctitrigout[1]	dram0_cm7_rsp_probe_ctitrigin[1]

Table continues on the next page...

Table 616. CTI assignments (continued)

CTI instance	Trigger number	Trigger in	Trigger out
	4	dram0_cm7_rsp_probe_ctitrigout[0]	Reserved
	3	dram0_other_rsp_probe_ctitrigout[1]	dram0_other_rsp_probe_ctitrigin[1]
	2	dram0_other_rsp_probe_ctitrigout[0]	Reserved
	1	Reserved	Reserved
	0	Reserved	Reserved
CTI_NoC7	7	Reserved	Reserved
	6	Reserved	Reserved
	5	Reserved	Reserved
	4	Reserved	Reserved
	3	Reserved	Reserved
	2	Reserved	Reserved
	1	Reserved	Reserved
	0	Reserved	Reserved
CTI_A53_Cluster0	7	EXTOUT[3]	EXTIN[3]
	6	EXTOUT[2]	EXTIN[2]
	5	EXTOUT[1]	EXTIN[1]
	4	EXTOUT[0]	EXTIN[0]
	3	Reserved	Reserved
	2	Reserved	CTIIRQ
	1	PMUIRQ	DBGRESTART
	0	DBGTRIGGER	EDBGRQ
CTI_A53_Cluster1	7	EXTOUT[3]	EXTIN[3]
	6	EXTOUT[2]	EXTIN[2]
	5	EXTOUT[1]	EXTIN[1]
	4	EXTOUT[0]	EXTIN[0]
	3	Reserved	Reserved
	2	Reserved	CTIIRQ
	1	PMUIRQ	DBGRESTART
	0	DBGTRIGGER	EDBGRQ
CTI_ETF_ETR0	7	Reserved	Reserved
	6	ETF 5 capture complete	ETF 5 trigger in
	5	ETF 4 capture complete	ETF 4 trigger in

Table continues on the next page...

Table 616. CTI assignments (continued)

CTI instance	Trigger number	Trigger in	Trigger out
	4	ETF 3 capture complete	ETF 3 trigger in
	3	ETF 2 capture complete	ETF 2 trigger in
	2	Cortex-A53_1 ETF capture complete	Cortex-A53 1 ETF trigger in
	1	Cortex-A53 0 ETF capture complete	Cortex-A53 0 ETF trigger in
	0	ETR capture complete	ETR trigger in
CTI_ETF_ETR1	7	Reserved	Reserved
	6	ETF5 full	ETF5 flush in
	5	ETF4 full	ETF4 flush in
	4	ETF3 full	ETF3 flush in
	3	ETF2 full	ETF2 flush in
	2	Cortex-A53_1 ETF full	Cortex-A53_1 ETF flush in
	1	Cortex-A53_0 ETF full	Cortex-A53_0 ETF flush in
	0	ETR full	Reserved
CTI_2	7	Reserved	Reserved
	6	Reserved	Reserved
	5	Reserved	Reserved
	4	Reserved	Reserved
	3	Reserved	Reserved
	2	Reserved	Reserved
	1	Used for trigger of non-manageable faults from FCCU (such as reset)	Reserved
	0	Used for trigger of manageable faults from FCCU (such as interrupts)	Reserved

75.8.1.1 S32G accelerator subsystem CTI assignments

Additional cross-triggering inputs and outputs are connected to the accelerator subsystem. These allow cross-triggering to/from the LLCE, USB trace, PCIe_1 trace, and the PFE trace clients. Each Probe in NoC provides up to eight Trigger Outputs (TrigOut) and provides eight Trigger Inputs (TrigIn), but only the two (Least Significant bits 1:0) are used. The others are not used or available.

Provided under NDA only
COMPANY PROPRIETARY
Beijing Hanmotive Technology Co. Ltd.
c8c462e5-821-4818-b25-3e6ab624c1

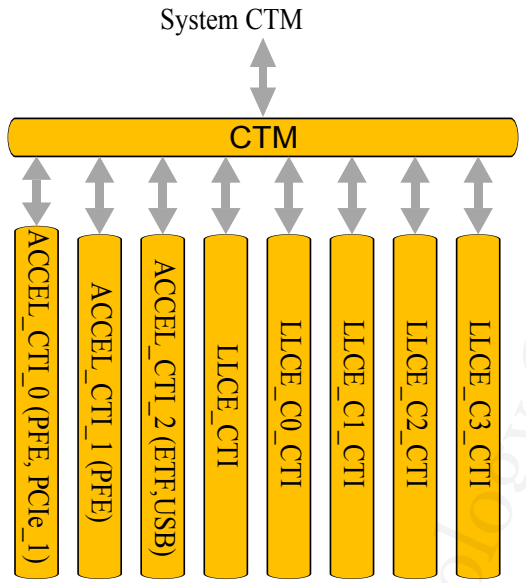


Figure 589. S32G Cross Trigger interfaces

The following table describes the S32G accelerator CTI assignments.

Table 617. S32G accelerator subsystem CTI assignments

CTI instance	Trigger number	Trigger in	Trigger out
ACCEL_CTI0	7	pfe_DDRresp_probe_mainCtiTrigOut[1]	pfe_DDRresp_probe_mainCtiTrigIn[1]
	6	pfe_DDRresp_probe_mainCtiTrigOut[0]	Reserved
	5	pfe_DDRreq_probe_mainCtiTrigOut[1]	pfe_DDRreq_probe_mainCtiTrigIn[1]
	4	pfe_DDRreq_probe_mainCtiTrigOut[0]	Reserved
	3	pcie_resp_probe_mainCtiTrigOut[1]	pcie_resp_probe_mainCtiTrigIn[1]
	2	pcie_resp_probe_mainCtiTrigOut[0]	Reserved
	1	pcie_req_probe_mainCtiTrigOut[1]	pcie_req_probe_mainCtiTrigIn[1]
	0	pcie_req_probe_mainCtiTrigOut[0]	Reserved
ACCEL_CTI1	7	pfe_HIF13Utilresp_probe_mainCtiTrigOut[1]	pfe_HIF13Utilresp_probe_mainCtiTrigIn[1]
	6	pfe_HIF13Utilresp_probe_mainCtiTrigOut[0]	Reserved
	5	pfe_HIF13Utilreq_probe_mainCtiTrigOut[1]	pfe_HIF13Utilreq_probe_mainCtiTrigIn[1]
	4	pfe_HIF13Utilreq_probe_mainCtiTrigOut[0]	Reserved
	3	pfe_HIF02resp_probe_mainCtiTrigOut[1]	pfe_HIF02resp_probe_mainCtiTrigIn[1]
	2	pfe_HIF02resp_probe_mainCtiTrigOut[0]	Reserved

Table continues on the next page...

Table 617. S32G accelerator subsystem CTI assignments (continued)

CTI instance	Trigger number	Trigger in	Trigger out
	1	pfe_HIF02req_probe_mainCtiTrigOut[1]	pfe_HIF02req_probe_mainCtiTrigIn[1]
	0	pfe_HIF02req_probe_mainCtiTrigOut[0]	Reserved
ACCEL_CTI2	7	Reserved	Reserved
	6	Reserved	Reserved
	5	Reserved	Reserved
	4	Reserved	Reserved
	3	event_etf_acqcomp	event_etf_flushin
	2	event_etf_full	event_etf_trigin
	1	usb_probe_mainCtiTrigOut[1]	usb_probe_mainCtiTrigIn[1]
	0	usb_probe_mainCtiTrigOut[0]	Reserved
LLCE_CTI	7	Reserved	Reserved
	6	Reserved	Reserved
	5	Reserved	Reserved
	4	Reserved	Reserved
	3	Reserved	Reserved
	2	Reserved	Reserved
	1	HTM1_TriggerOut	Reserved
	0	HTM0_TriggerOut	Reserved
LLCE_C0_CTI	7	Reserved	C0 DBGRESTART
	6	Reserved	Reserved
	5	Reserved	C0 TSTOP (MTB Trace Stop)
	4	Reserved	C0 TSTART (MTB Trace Start)
	3	Reserved	Reserved
	2	Reserved	Reserved
	1	Reserved	Reserved
	0	C0 Halted	C0 EDBGRQ
LLCE_C1_CTI	7	Reserved	C1 DBGRESTART
	6	Reserved	Reserved
	5	Reserved	C1 TSTOP (MTB Trace Stop)
	4	Reserved	C1 TSTART (MTB Trace Start)

Table continues on the next page...

Table 617. S32G accelerator subsystem CTI assignments (continued)

CTI instance	Trigger number	Trigger in	Trigger out
	3	Reserved	Reserved
	2	Reserved	Reserved
	1	Reserved	Reserved
	0	C1 Halted	C1 EDBGRQ
LLCE_C2_CTI	7	Reserved	C2 DBGRESTART
	6	Reserved	Reserved
	5	Reserved	C2 TSTOP (MTB Trace Stop)
	4	Reserved	C2 TSTART (MTB Trace Start)
	3	Reserved	Reserved
	2	Reserved	Reserved
	1	Reserved	Reserved
	0	C2 Halted	C2 EDBGRQ
LLCE_C3_CTI	7	Reserved	C3 DBGRESTART
	6	Reserved	Reserved
	5	Reserved	C3 TSTOP (MTB Trace Stop)
	4	Reserved	C3 TSTART (MTB Trace Start)
	3	Reserved	Reserved
	2	Reserved	Reserved
	1	Reserved	Reserved
	0	C3 Halted	C3 EDBGRQ

75.9 NoC trace

The system NoC provides all bus interconnectivity between initiators (masters) and targets (slaves). Trace features implemented in NoC allow bus transactions to be observed. NoC contains probes that monitor the buses, and these probes feed data to an observer. NoC generates STPv2 messages and transports the messages on the Arm Trace Bus as 64-bit cells. Trace packets are compliant with the AMBA3 ATB Protocol Specification 1.0.

Two kinds of probes are implemented: packet probes (requesters and responders) and transaction probes. Packet probes have these features:

- Runtime programmable selection of packet probe points
- Runtime programmable filtering for packet tracing
  - Four filters available on each probe
- Recording of traffic and link statistics
- Event counting of signals external to NoC
  - Eight 16-bit counters on each probe

— Counters can be cascaded for additional range

- Generation of trace or statistic packets, or both, for the observation network
- 32 bytes of buffers on each probe
- Optional timestamping of packets
- Most probes support intrusive (stalls upstream processing to avoid overflows) and spy (do not stall, discards data to avoid overflows) modes, however, some probes support only spy mode (see table below)

The following figure shows the high-level hierarchy of the NoC observers.

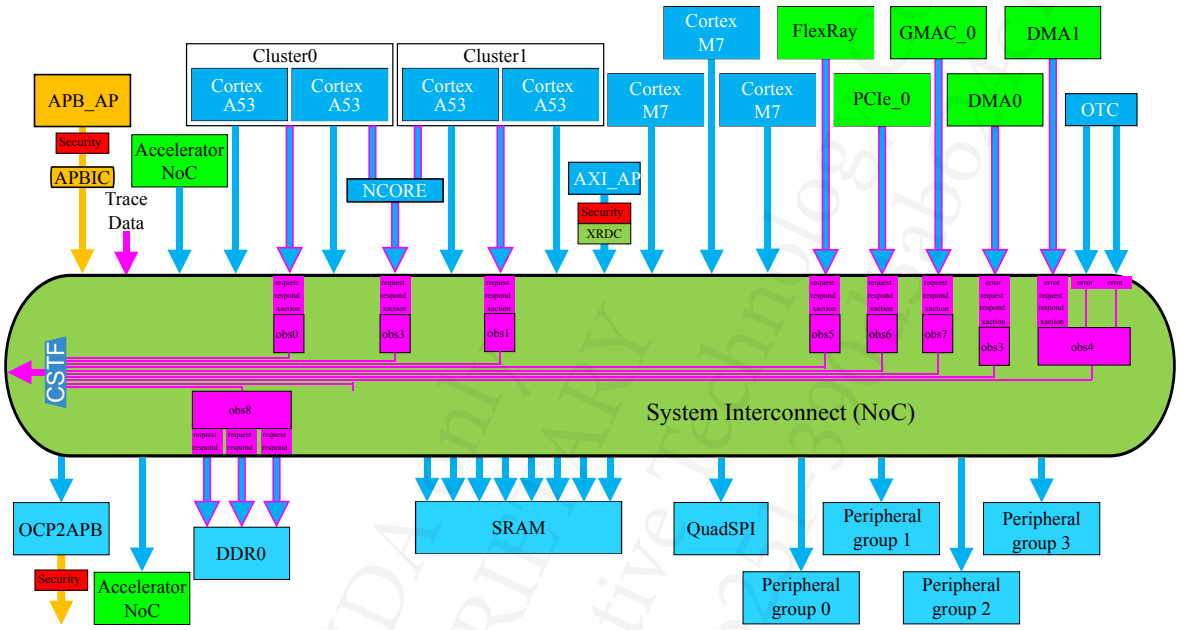


Figure 590. NoC connections

This figure also shows the points at which trace observers/probes can monitor the bus activities. The type of observability is shown in the following table.

Table 618. NoC observation network

Observer	Buffer size	Probe	Probe ID	Probe point	Probe type	Trace modes supported
A53 Cluster 0 non-coherent data access bus (Obs0)	256 bytes 4 packets	a53_cluster0_noncoh_req_probe	0	Data Request	Packet	Intrusive, Spy
		a53_cluster0_noncoh_rsp_probe	1	Data Response	Packet	Intrusive, Spy
		a53_cluster0_noncoh_xaction_probe	2	Transaction	Transaction	Spy
A53 Cluster 1 non-coherent data access bus (Obs1)	256 bytes 4 packets	a53_cluster1_noncoh_req_probe	0	Data Request	Packet	Intrusive, Spy
		a53_cluster1_noncoh_rsp_probe	1	Data Response	Packet	Intrusive, Spy

Table continues on the next page...

Table 618. NoC observation network (continued)

Observer	Buffer size	Probe	Probe ID	Probe point	Probe type	Trace modes supported
		a53_cluster1_noncoh_xaction_probe	2	Transaction	Transaction	Spy
Ncore (A53 coherent) data access bus (Obs2)	256 bytes 4 packets	ncore_req_probe	0	Data Request	Packet	Intrusive, Spy
		ncore_rsp_probe	1	Data Response	Packet	Intrusive, Spy
		ncore_xaction_probe	2	Transaction	Transaction	Spy
DRAM 0 (Obs8)	256 bytes 4 packets	dram0_a53_req_probe	0	Data Request	Packet	Intrusive, Spy
		dram0_a53_rsp_probe	1	Data Response	Packet	Intrusive, Spy
	256 bytes 4 packets	dram0_m7_req_probe	2	Data Request	Packet	Intrusive, Spy
		dram0_m7_rsp_probe	3	Data Response	Packet	Intrusive, Spy
	256 bytes 4 packets	dram0_other_req_probe	4	Data Request	Packet	Intrusive, Spy
		dram0_other_rsp_probe	5	Data Response	Packet	Intrusive, Spy
eDMA0 (Obs3)	128 bytes 4 packets	edma0_req_probe	0	Data Request	Packet	Intrusive, Spy
		edma0_rsp_probe	1	Data Response	Packet	Intrusive, Spy
		edma0_xaction_probe	2	Transaction	Transaction	Spy
		edma0_error_probe		Error Probe (logger)	Error	Spy
		Reserved		Reserved	Reserved	Spy
eDMA0 (Obs3)	128 bytes 4 packets	llce error probe		Error Probe (logger)	Error	Spy
eDMA1 (Obs4)	128 bytes 4 packets	edma1_req_probe	0	Data Request	Packet	Intrusive, Spy
		edma1_rsp_probe	1	Data Response	Packet	Intrusive, Spy
		edma1_xaction_probe	2	Transaction	Transaction	Spy
		edma1_error_probe		Error Probe (logger)	Error	Spy
		otc_ht error probe		Error Probe (logger)	Error	Spy

Table continues on the next page...

Table 618. NoC observation network (continued)

Observer	Buffer size	Probe	Probe ID	Probe point	Probe type	Trace modes supported
		otc_ll error probe		Error Probe (logger)	Error	Spy
GMAC_0 (Obs5)	128 bytes 4 packets	gmac0_req_probe	0	Data Request	Packet	Intrusive, Spy
		gmac0_rsp_probe	1	Data Response	Packet	Intrusive, Spy
		gmac0_xaction_probe	2	Transaction	Transaction	Spy
Flex Ray (Obs6)	128 bytes 4 packets	fray_req_probe	0	Data Request	Packet	Intrusive, Spy
		fray_rsp_probe	1	Data Response	Packet	Intrusive, Spy
		fray_xaction_probe	2	Transaction	Transaction	Spy
PCIe_0 (Obs7)	128 bytes 4 packets	pcie0_req_probe	0	Data Request	Packet	Intrusive, Spy
		pcie0_rsp_probe	1	Data Response	Packet	Intrusive, Spy
		pcie0_xaction_probe	2	Transaction	Transaction	Spy

The debug parameters for system NoC are provided in the following table.

Table 619. Debug parameters for system NoC

Observer	Probe	Probe ID	Probe type	# Filters	Allow filter on user	Allow filter on enabled bytes	Transaction profiling	# Counters	wCounters	Statistics collection	# Statistics counters
a53_cluster0_noncoh_observer	a53_cluster0_noncoh_req_packet	0	packet	4	True	True	False	0	Not applicable	True	8
	a53_cluster0_noncoh_rsp_packet	1	packet	4	True	True	False	0	Not applicable	True	8
	a53_cluster0_noncoh_xaction	2	xaction	0	False	False	True	4	8	True	5

Table continues on the next page...

Table 619. Debug parameters for system NoC (continued)

Observer	Probe	Probe ID	Probe type	# Filters	Allow filter on user	Allow filter on enabled bytes	Transaction profiling	# Counters	wCounters	Statistics collection	# Statistics counters
a53_cluster1_noncoh_observer	a53_cluster1_noncoh_req_packet	0	packet	4	True	True	False	0	Not applicable	True	8
	a53_cluster1_noncoh_rsp_packet	1	packet	4	True	True	False	0	Not applicable	True	8
	a53_cluster1_noncoh_xaction	2	xaction	0	False	False	True	4	8	True	5
concerto_observer	concerto_req_packet	0	packet	4	True	True	False	0	Not applicable	True	8
	concerto_rsp_packet	1	packet	4	True	True	False	0	Not applicable	True	8
	concerto_xaction	2	xaction	0	False	False	True	4	8	True	5
	Link99	Not applicable	error	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable
enet_observer	enet_req_packet	0	packet	4	True	True	False	0	Not applicable	True	8
	enet_rsp_packet	1	packet	4	True	True	False	0	Not applicable	True	8
	enet_xaction	2	xaction	0	False	False	True	4	8	True	5
fray_observer	fray_req_packet	0	packet	4	True	True	False	0	Not applicable	True	8
	fray_rsp_packet	1	packet	4	True	True	False	0	Not applicable	True	8

Table continues on the next page...

Table 619. Debug parameters for system NoC (continued)

Observer	Probe	Probe ID	Probe type	# Filters	Allow filter on user	Allow filter on enabled bytes	Transaction profiling	# Counters	wCounters	Statistics collection	# Statistics counters
	fray_xaction	2	xaction	0	False	False	True	4	8	True	5
	Link113	Not applicable	error	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable
pcie_observer	pcie_req_packet	0	packet	4	True	True	False	0	Not applicable	True	8
	pcie_rsp_packet	1	packet	4	True	True	False	0	Not applicable	True	8
	pcie_xaction	2	xaction	0	False	False	True	4	8	True	5
edma0_observer	edma0_req_packet	0	packet	4	True	True	False	0	Not applicable	True	8
	edma0_rsp_packet	1	packet	4	True	True	False	0	Not applicable	True	8
	edma0_xaction	2	xaction	0	False	False	True	4	8	True	5
	Link107	Not applicable	error	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable
	atf_dtpLinkResp148	Not applicable	error	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable
edma1_observer	edma1_req_packet	0	packet	4	True	True	False	0	Not applicable	True	8
	edma1_rsp_packet	1	packet	4	True	True	False	0	Not applicable	True	8
	edma1_xaction	2	xaction	0	False	False	True	4	8	True	5
	Link108	Not applicable	error	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable	Not applicable

Table continues on the next page...

Table 619. Debug parameters for system NoC (continued)

Observer	Probe	Probe ID	Probe type	# Filters	Allow filter on user	Allow filter on enabled bytes	Transaction profiling	# Counters	wCounters	Statistics collection	# Statistics counters
dram0_a53_observer	dram0_a53_req_packet	0	packet	4	True	True	False	0	Not applicable	True	8
	dram0_a53_rsp_packet	1	packet	4	True	True	False	0	Not applicable	True	8
	dram0_m7_req_packet	0	packet	4	True	True	False	0	Not applicable	True	8
	dram0_m7_rsp_packet	1	packet	4	True	True	False	0	Not applicable	True	8
	dram0_other_req_packet	0	packet	4	True	True	False	0	Not applicable	True	8
	dram0_other_rsp_packet	1	packet	4	True	True	False	0	Not applicable	True	8

75.9.1 S32G accelerator subsystem NoC trace

The S32G accelerator subsystem NoC contains two trace observer ports that can generate CoreSight compliant trace data. Each NoC master has probe(s) associated with it to determine what information to trace. The following table shows the NoC ports and their probe/observer alignment.

Table 620. S32G accelerator subsystem NoC probes and observer mapping

Port	Probes	Type	Buffering	Observers
m_pcie_1	pcie_req_probe	packet	128 bytes 2 header packets	pcie_observer
	pcie_resp_probe	packet		
m_usb	usb_probe	spy		
m_pfe_DDR	pfe_ddr_req_probe	packet		
	pfe_dd_resp_probe	packet		
m_pfe_HIF0	pfe_hif02_req_probe	packet	256 bytes 2 header packets	pfe_observer
m_pfe_HIF2	pfe_hif02_resp_probe	packet		
m_pfe_Util	pfe_hif13Util_req_probe	packet	256 bytes 2 header packets	
m_pfe_HIF1	pfe_hif13_Util_resp_probe	packet		
m_pfe_HIF3				

The debug parameters for system NoC are provided in the following table.

Table 621. Debug parameters for system NoC

Probe	Type	Observer	Mode	Probe ID	# Filter	Allow filter on user	Allow filter on enabled bytes	Transaction profiling	Statistics collection	# Statistics counters
pcie_req_probe	packets	pcie	intrusive	0	2	True	True	False	True	8
pcie_resp_probe	packets	pcie	intrusive	1	2	True	True	False	True	8
ppfe_DDRreq_probe	packets	pcie	intrusive	2	2	True	True	False	True	8
pfe_DDRresp_probe	packets	pcie	intrusive	3	2	True	True	False	True	8
pfe_HIF02_req_probe	packets	pfe	intrusive	4	2	True	True	False	True	8
pfe_HIF02_resp_probe	packets	pfe	intrusive	5	2	True	True	False	True	8
pfe_HIF13Utilreq_probe	packets	pfe	intrusive	6	2	True	True	False	True	8
pfe_HIFT13Utilresp_probe	packets	pfe	intrusive	7	2	True	True	False	True	8
usb_probe	packets	pcie	spy	8	2	True	True	False	True	8

All probes are defined as packet probes except for the USB probe, which is defined as a spy probe. Each probe has four filters associated with them that can be used to filter when a trace packet is generated. All probes also include eight statistic counters that can be used to count various events and a 48-bit CoreSight timestamp. The debug registers to control the probes and observers are programmable via the accelerator NoC's m_debug_apb master port. Probes can be configured to be intrusive or non-intrusive. When intrusive mode is set to 1, no packets are dropped from trace. Back pressure will be applied on the functional data path if the trace FIFOs get full. If intrusive mode is set to 0, no back pressure is applied and it's then possible to lose trace data if the trace FIFOs get full. Details on the registers can be found in the NoC chapter.

The following diagram shows the connectivity between the probes and the observers.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co. Ltd.
c8c462e5-4821-48a6-b1e3-90a0b0f6047

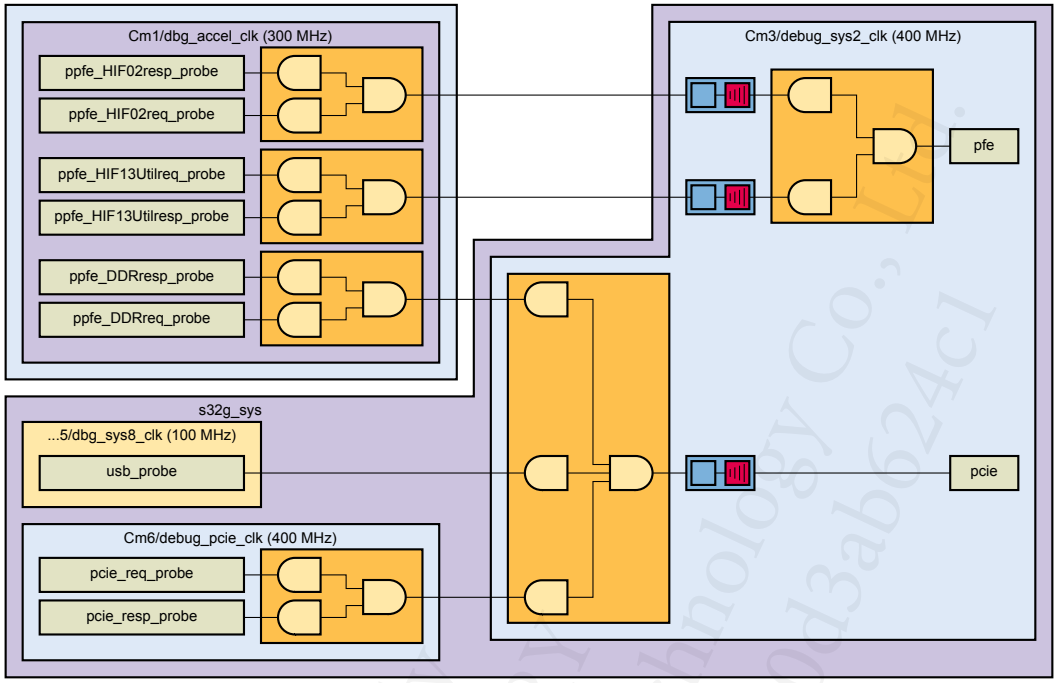


Figure 591. S32G accelerator NoC probe topology

NOTE

Probe buffers may overflow when setting "payload_en=1" and "intrusive_mode=0". When setting intrusive_mode to 1, buffers will not overflow, but backpressure will be applied to the data path so as not to lose trace information.

75.9.2 Observability

75.9.2.1 Concepts

The technology that allows activity within NoC interconnects to be observed and reported, provides the following services:

- Error detection, logging, and reporting
- Packet tracing
- Traffic or link statistics computation
- Transaction latency profiling
- Events or statistics reporting to the system via industry standard third-party interface, MIPI® STPv2 over the Arm® ATB Coresight™ trace interface

To provide these services, NoC technology uses task-specific probes placed on probe points located in the datapath topology. Traced packets and observed events are gathered by observers that can issue interrupts and forward detailed data to the supported third-party interfaces.

75.9.2.1.1 Packet and transaction probes

NoC packet probes provide the following features:

- Run-time programmable selection of packet probe points for probes connected to multiple points
- Run-time programmable filtering for packet tracing
- Recording of traffic and link statistics
- Event counting of signals external to NoC

- Generation of trace or statistic packets, or both, for the observation network
- Optional time stamping of packets

NoC transaction probes provide the following features:

- Run-time programmable selection of transaction probe points
- Transaction profiling in terms of latency or number of pending transactions
- Event counting of signals external to the NoC
- Generation of statistic packets for the observation network
- Optional time stamping of packets

NoC error probes detect packets in error and forward them to the associated observer via the observation network. The error packets are recorded by error loggers in the observer. An error logger can be programmed to assert an output signal when an error packet is received. The signal can be used as a system interrupt. Observers can contain one or more loggers and associated interrupt signals, each dedicated to specific error codes or error security levels.

By default, a new error can only be logged when the previous one is cleared by proper register access. If an error probe reports new errors and the associated logger is already full, then subsequent errors are dropped. Alternatively, the error logger can be programmed to queue up error packets and only accept them once the currently logged error has been cleared.

75.9.2.1.2 Observers

Data collected by the certain NoC probes can be transferred to NoC observer units via the observation network.

Observers can be connected to NoCs via an ATB™ interface, with MIPI STPv2 encapsulation and time stamping. Collected data can be forwarded either to an off-chip interface, or on-chip storage buffer for analysis by an on-chip debugger.

75.9.2.1.2.1 Time stamping probed events

The timestamp is added to the observation packet, regardless of whether the probe is configured for tracing or statistic collection.

The timestamp remains attached to the observation packet until the packet reaches the observer. This timestamp value is extracted and the difference between the current timestamp and the previous timestamp value, which was sent in the STPv2 stream, is calculated. This delta value is appended to the first STPv2 packet embedding the ATB packet.

75.9.2.2 Operation

75.9.2.2.1 Logging errors

75.9.2.2.1.1 Reconstructing transaction addresses

75.9.2.2.1.1.1 Calculating initiator – target transaction addresses

Once the initiator–target mapping associated with the logged packet has been identified, the complete transaction address, either from the perspective of the initiator or the target, can be calculated.

75.9.2.2.1.1.1.1 Initiator transaction address

From the perspective of the initiator, calculate InitAddr by:

```
InitAddr = Init localAddress | Addr
```

where:

- Addr – Logged packet transport address Addr given in register ErrLog3 and, additionally, ErrLog4, in the case where the address exceeds 32 bits
- Init localAddress – Value from the identified initiator-target mapping in subsection "Aperture values” of the structure information file

75.9.2.2.1.1.1.2 Target transaction address

The transaction address from the perspective of the target, calculate TargAddr by:

```
TargAddr = Targ localAddress | Addr
```

Where the definitions of the terms are the same as used for calculating the initiator transaction address (see Initiator transaction address).

75.9.2.2.2 Tracing packets

75.9.2.2.2.1 Basic packet tracing

To trace packets, the packet probe must be programmed as follows:

1. Set field TraceEn of register MainCtl to 1 to enable forwarding of traced packets to the connected observer. Optionally set field PayloadEn of register MainCtl to 1 if the packet payload should be included in the trace
2. Set register TracePortSel to the value corresponding to the probe point of interest
3. Set field GlobalEn of register CfgCtl to 1
4. Set register FilterLut to -1 (all bits set)

NOTE

A lookup table acts as a simple gate when there are no filters.

After competing the programming sequence above, filters must be configured (see Filtering packets).

75.9.2.2.2.2 Filtering packets

Each packet probe supports up to four packet filters.

The following list comprises the complete set of filter registers, where the value for filter *n* ranges from 0 to 3.

- Filters_N_RouteIdBase
- Filters_N_Status
- Filters_N_RouteIdMask
- Filters_N_Length
- Filters_N_AddrBase_Low
- Filters_N_Urgency
- Filters_N_AddrBase_High
- Filters_N_UserBase
- Filters_N_WindowSize
- Filters_N_UserMask
- Filters_N_UserBaseHigh
- Filters_N_UserMaskHigh
- Filters_N_Opcode

75.9.2.2.2.2.1 Filtering on an address range

Packets which target a given address range of a specific mapping may be filtered by programming the set of filter registers.

Licensed under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c12e5-4821-48a8-b231-39d3ab624c1

Table 622. Packet filtering registers

Register	Value
Filters_N_RouteIdBase	RouteId subfields Initflow, TargFlow, and TargSubRange.
Filters_N_RouteIdMask	Set bits to 1 if RouteId fields are to be used by filter.
Filters_N_AddrBase_Low/ High	Address offset from initiator or target local address.
Filters_N_WindowSize	Log_2 of address range.

75.9.2.2.2.2.2 Selecting packets based on packet opcode

Packets can be selected based on the opcode field (Opc) in the packet header using register Filters_N_Opcode. In the following example, the filter is required to select all packets with Opc equal to RD and all packets belonging to a locked sequence.

Program the register fields of Filters_N_Opcode as shown in Table 623.

Table 623. Packet filtering registers

Field name	Setting
RdEn	1
LockEn	1

The other register fields are set to 0. Figure 592 shows a sequence of packets with the resulting filter hits.

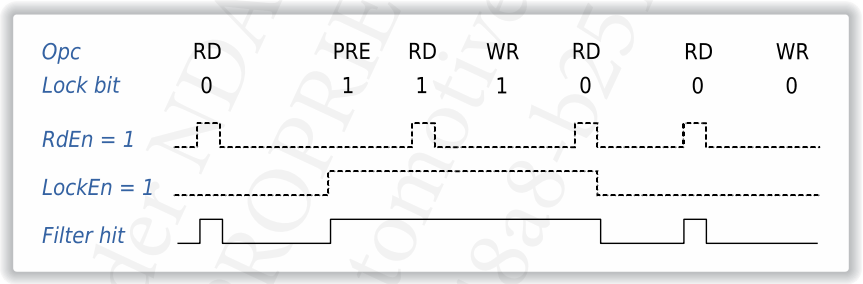


Figure 592. Sequence of packets

75.9.2.2.2.2.3 Excluding packet header fields from filter criteria

To ignore a specific packet header field in the filter, set its associated register with the value in shown Table 624.

Table 624. Excluding packet header field register settings

Register name	Setting
Filters_N_RouteIdMask	0
Filters_N_WindowSize	-1[1]
Filters_N_Opcode	Fh
Filters_N_Status	3h
Filters_N_Length	Fh

Table continues on the next page...

Table 624. Excluding packet header field register settings (continued)

Register name	Setting
Filters_N_Urgency	0
Filters_N_UserMask (if present)	0
Filters_N_UserMaskHigh (if present)	0

1. Setting register Filters_N_WindowSize to –1 programs the filter to accept packets which target anywhere in the full address range supported by the transport network. Hence the address is effectively excluded from the filter criteria.

75.9.2.2.2.2.4 Tracing all packets

If it is required to trace all packets, then the probe's filtering function must be programmed as follows:

1. Program one of the filters to accept all packet header field values. Set the filter registers to the values given in the table in the preceding section "Excluding packet header fields from filter criteria"
2. Program register FilterLut to select the output of filter programmed in the previous step

75.9.2.2.2.3 Tracing error packets

Error packets, that is packets with header field Status set to ERR, are not traced by default. To include error packets in the trace output, write MainCtl[ErrEn] = 1 and MainCtl[ErrEn] = 1.

75.9.2.2.2.4 Combining filters

A look-up table (LUT) is instantiated in the probe, which allows a hit to be generated on any logical combination of packet filters. The logic function is programmed via register FilterLut. The value programmed in the register corresponds to the truth table applied on the filter outputs.

For example, with two filters F0 and F1, sixteen possible functions can be defined, a subset of which is shown in the following table.

Table 625. Example subset of sixteen possible functions

Filter hit			Required filter combination					
F1	F0		F0	F1	F1 & F0	F1 | F0	F1 ^ F0	~F0
0	**0**		0	0	0	0	0	1
0	**1**		1	0	0	1	1	0
1	**0**		0	1	0	1	1	1
1	**1**		1	1	1	1	0	0
			Value to program in FilterLut					
			Ah	Ch	8h	Eh	6h	5h

NOTE

In this example, setting register FilterLut to Fh will generate a LUT hit for any combination of filter hits, effectively bypassing the filters.

Provided under NDA
COMPANY PROPRIETARY
Beijing CTI Automotive Technology Co., Ltd.
c8c462e5-4821-4a68-b2a1-390d3ab624c1

75.9.2.2.2.5 Configuring and managing trace alarm events

Trace alarm events can be separately enabled, recorded, and cleared using registers TraceAlarmEn, TraceAlarmStatus, and TraceAlarmClr respectively. In each register, a dedicated bit is associated with each filter hit; the MSB is associated with the LUT hit.

The trace alarm logic can be configured to drive output signal TraceAlarm if one or more trace alarm events are recorded in register TraceAlarmStatus. Signal TraceAlarm can be used directly as an interrupt source, or can be routed through a sideband manager.

To use the trace alarm feature:

1. Enable hits from selected filters 0 to (nFilter – 1) to generate a trace alarm by setting register TraceAlarmEn bits 0 to (nFilter – 1) to 1.
   - Optionally, enable LUT hits to generate a trace alarm by setting the MSB of TraceAlarmEn.
2. Set field AlarmEn to 1 in register MainCtl in order to activate signal TraceAlarm when a trace alarm occurs.

If TraceAlarm is asserted, the interrupt service routine should:

1. Identify the source or sources of the trace alarm by reading register TraceAlarmStatus.
2. Clear the source or sources of the trace alarm by setting the corresponding bits to 1 in register TraceAlarmClr. Signal TraceAlarm deasserts when all bits in TraceAlarmStatus have been cleared.

75.9.2.2.3 Collecting statistics

75.9.2.2.3.1 Counting packets over fixed period

The following programming sequence counts packets at a given probe point using statistic counter 0:

1. Set field StatEn to 1 in register MainCtl
2. Set register Counters_0_PortSel to the value corresponding to the probe point of interest
3. Set register Counters_0_Src to 6h (PKT) to count packets
4. Specify the period during which the packets should be counted by setting register StatPeriod to: log2 (interval expressed in number of probe clock cycles)
5. Set field GlobalEn of register CfgCtl to 1 to enable packet counting

Once time $2^{StatPeriod}$ has elapsed, the number of packets counted is dumped to the observer.

75.9.2.2.3.2 Counting filtered packets over fixed period

The following programming sequence counts hits from packet filter 2 using statistic counter 0:

NOTE

The important distinction between this sequence and the one in a related programming example (see Counting packets over fixed period) is that the probe port is selected using register TracePortSel rather than Counters_0_PortSel

1. Set field StatEn to 1 in register MainCtl
2. Set register TracePortSel to the value corresponding to the probe port of interest
3. Set register Counters_0_Src to 0xE (FILT2) to count packets from filter 2
4. Specify the period during which the packets should be counted by setting register StatPeriod to: log2(interval expressed in number of probe clock cycles)
5. Set field GlobalEn of register CfgCtl to 1 to enable packet counting

Once time $2^{StatPeriod}$ has elapsed, the number of filtered packets counted is dumped to the observer.

75.9.2.2.4 Transaction profiling operation

The programming sequence examples in this section constitute, for a given category of use cases, the obligatory base from which to develop sequences to handle specific needs within the category.

75.9.2.2.4.1 Measuring latency

The following example of a run-time programming sequence shows how a transaction probe can be used to measure transaction latency.

The measurement specification is:

- For init0, monitor delays for read transactions only
- For init1, monitor delays for read and write transactions

75.9.2.2.4.2 Programming the packet probe statistics subsystem

The statistics counters in the packet probe unit of the transaction probe must be programmed to count the delay events recorded by the transaction profiler unit.

1. Map the five delay event bins, associated with transactions issued by init0, from the transaction profiler unit to statistics counters in the packet probe unit by programming the following registers:
   - Counters_0_Src = 20h => Counts the number of transactions from init0 with a delay between 0 and 1
   - Counters_1_Src = 21h => Counts the number of transactions from init0 with a delay between 2 and 4
   - Counters_2_Src = 22h => Counts the number of transactions from init0 with a delay between 5 and 9
   - Counters_3_Src = 23h => Counts the number of transactions from init0 with a delay between 10 and 99
   - Counters_4_Src = 24h => Counts the number of transactions from init0 with a delay greater than 100
2. Map the five delay event bins, associated with transactions issued by init1, from the transaction profiler unit to statistics counters in the packet probe unit by programming the following registers:
   - Counters_5_Src = 25h => Counts the number of transactions from init1 with a delay between 0 and 1
   - Counters_6_Src = 26h => Counts the number of transactions from init1 with a delay between 2 and 4
   - Counters_7_Src = 27h => Counts the number of transactions from init1 with a delay between 5 and 9
   - Counters_8_Src = 28h => Counts the number of transactions from init1 with a delay between 10 and 99
   - Counters_9_Src = 29h => Counts the number of transactions from init1 with a delay greater than 100
3. Set register StatPeriod to $\log_2$(period), that is, the period during which delay events should be counted before being sent to the observer
4. Set field StatEn of register MainCtl to 1 to enable the statistics counters to count delay events

75.9.2.2.4.3 Enabling the transaction probe

1. Set field GlobalEn of register CfgCtl to 1 to enable the statistics counters
2. Set register En to 1 to enable the transaction profiling counters

75.9.2.2.4.4 Programming the init0 NIU transaction filter

1. Set register Mode to 1 to set the delay mode to latency
2. Set field RdEn of register Opcode to 1 to monitor read transactions only

75.9.2.2.4.5 Programming the init1 NIU transaction filter

1. Set register Mode to 1 to set the delay mode to latency
2. Set fields RdEn and WrEn of register Opcode both to 1 to monitor read and write transactions

75.9.2.2.5 Activating observer debug output

75.9.2.2.5.1 STPv2 interface

The observers generate STPv2 formatted trace packets. To enable trace output, the following registers must be programmed.

1. Specify the required period between STP synchronization sequences in register AsyncPeriod. Ensure that the value of AsyncPeriod is large enough to allow STPv2 cells, encapsulating Trace / Statistic Packets, to be inserted in between STPv2 synchronization sequences
2. Ensure that register STPv2En is set to 1 to enable the STPv2 interface
3. Set register AtbId to a value that is unique not only within the NoC but also within the debug system
4. Set register AtbEn to 1 to enable the ATB function

75.9.2.3 Technical reference

NoC observability technology comprises two principle components: observation probes and observer units.

The section provides detailed technical reference information for these components, along with the NoC packet format description needed to analyze error logs and packet trace information.

75.9.2.3.1 Observation probes

This section provides the detailed technical reference information for the NoC Observability technology collectively called observation probe units.

75.9.2.3.1.1 Packet probe

The NoC packet probe unit collects trace and statistics data. The unit can be used in conjunction with the NoC transaction profiler unit to create a transaction probe.

75.9.2.3.1.1.1 Packet probe features

The unit has the following main features:

- Support for any transport packet format and serialization
- Trace mode with programmable packet filters
- Optional packet payload tracing
- Programmable flow-control modes: Overflow Management and Intrusive
- Statistics can be collected and sent to an observer either periodically or after manual triggering
- Alarms on both trace filters and statistics counters
- Optional time stamping of probed events

75.9.2.3.1.2 Overflow flow-control for tracing

When not using Intrusive Mode (see register field IntrusiveMode of register MainCtl), an overflow packet is added after the stalled packet(s), which reports how many observation packets have been lost.

This Overflow packet has the following format:

- Opc = WR, Status = ERR

- Debug = (probeId, Urgency= 0, Mod=1)
- RouteId = RouteID of the packet that triggered the overflow
- Addr[8:0] = number of lost or uncompleted observation packets, minus one
- Len: 0 if no payload lost; 1 otherwise
- Lock = Addr[8:0] = Security = User = 0

In case of payload overflow, the current packet is completed according to its type; then an overflow packet is also sent:

- Write request packets are fully completed with the correct number of words, all having null Data and BE
- A single data word with null data that terminates the tracing of the read payload for that transaction completes Read response and Read continuous packets

75.9.2.3.1.3 Transaction filter

The NoC transaction filter unit uses programmable criteria to select request and response events.

The unit is integrated into the NIU and is connected to the NIU generic-to-transport unit. More than one transaction filter can be connected to a NoC transaction profiler unit.

75.9.2.3.1.4 Transaction profiler

The NoC transaction profiler unit receives events from one or more transaction filters and organizes them into histogram bins, defined by programmable thresholds.

The unit is connected to a standard Packet Probe via the unit external event input ports.

75.9.2.3.2 Observers

This section provides detailed technical reference information for the NoC Observability technology Observer unit.

75.9.2.3.2.1 Error logging

Errors that occur in packets observed by probes can be logged.

75.9.2.3.2.1.1 Features

- Error filtering can be based on selected error codes, user and security flags and sideband signals
- An optional fault interrupt can be asserted on an error
- Can be used in conjunction with trace probe filtering to monitor a specific port

75.9.2.3.2.1.2 NoC error codes

The following table lists all error codes that are relevant to NoC interconnect operation.

Table 626. NoC error codes

Code	Value	Source	Type
SLV	0	Target	Target error detected by slave
DEC	1	Initiator NIU	Address decode error
UNS	2	Target NIU	Unsupported request
DISC	3	Power disconnect	Disconnected target or domain

Table continues on the next page...

Table 626. NoC error codes (continued)

Code	Value	Source	Type
SEC	4	Initiator NIU or firewall	Security violation
HIDE	5	Firewall	Hidden security violation, reported as ok to initiator
TMO	6	Target NIU	Timeout
RSV	7	None	Reserved

75.9.2.3.2.2 STPv2 stream generation

The NoC STPv2 adapter in the observer encapsulates ATB packets and generates synchronization sequences to provide a MIPI-STPv2 compliant STP stream. Time stamps can optionally be included in the stream.

75.9.2.3.2.2.1 Features

- Compliant with AMBA®-3 ATB and MIPI® STPv2 specifications
- Supports ATB data widths from 8 to 64 bits
- Supports STPv2 opcodes ASYNC, VERSION, NULL, NULL_TS, D64, D64MTS, and MERR
- Synchronization sequence generation with optional time stamp
- Encapsulation of probe time stamps
- Supports all time stamp length encodings

NOTE

For details, see MIPI® Alliance Specification for System Trace Protocol (STP), version 2.00.01, 24 February 2010.-STPv2 specification.

75.9.2.3.2.2.2 Synchronization sequence generation

The synchronization sequence permits the SoC debug system to identify the STPv2 packet boundary. A time stamp can be optionally inserted in the sequence to which the SoC debug system may resynchronize itself.

The synchronization sequence is sent when one of the following events occurs:

- After a reset
- On the rising edge of input signal SyncReq
- When the number of clock cycles since the last synchronization sequence is between 2AsyncPeriod and 223 inclusive, and register AsyncPeriod is greater than 0

NOTE

The synchronization sequences are inserted instead of ATB synchronization cells.

The STPv2 adapter can issue synchronization sequences either with or without time stamps. The examples in figures that follow are represented according to the following key.

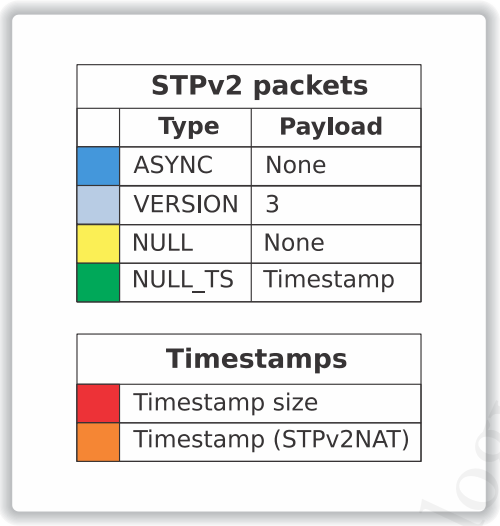


Figure 593. STPv2 synchronization sequence key

75.9.2.3.2.2.2.1 STPv2 synchronization sequence without timestamp

Synchronization sequences are sent without a timestamp when the observer STPv2 parameter wTimeStamp is set to 0. In this case, the sequence comprises the STPv2 packets ASYNC and VERSION, as well as NULL packets for padding. Such a sequence is shown in the following figure.

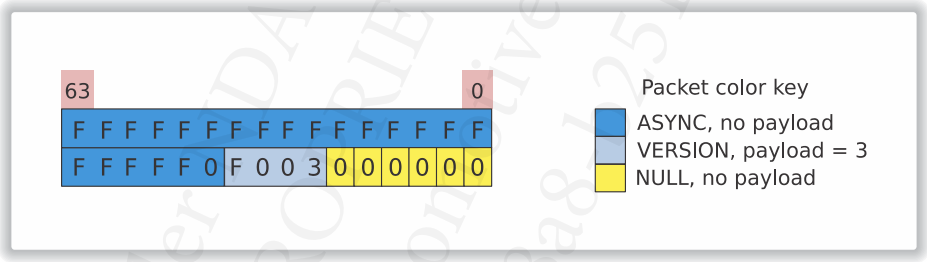


Figure 594. STPv2 synchronization sequence without timestamp

NOTE

The sequence in the figure assumes a data width of 64 for the ATB debug port, as set by parameter wData (Specification: Interface: Observation: debugOutput).

75.9.2.3.2.2.2.2 STPv2 synchronization sequence with timestamp

If wTimeStamp is greater than zero, synchronization sequences are sent with time stamps. The sequence comprises the STPv2 packets ASYNC, VERSION, NULL, and NULL_TS. The time stamp value, received on observer input signal TimeStamp and extended to 64 bits, is sent in the payload of NULL_TS.

The synchronization sequence with time stamp is shown in the following figure.

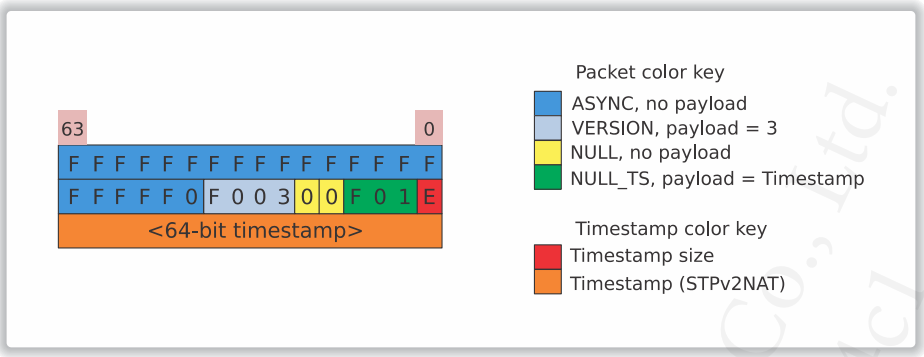


Figure 595. STPv2 synchronization sequence with timestamp

In the figure:

- Regardless of the width of the time stamp, specified by wTimeStamp, the time stamp is extended to 64 bits
- All 16 nibbles of the time stamp are significant, indicated by the time stamp size nibble being set to Eh
- The protocol version is set to 3 in the payload of packet VERSION. This indicates that the time stamp is encoded using the uncompressed STPv2NAT format

75.9.2.3.2.2.3 Encapsulation of ATB packets

The STPv2 adapter encapsulates 64–bit ATB cells into the payload of 64–bit STPv2 data packets. The first ATB cell of an observation packet is encapsulated into a D64MTS STPv2 packet; D64 STPv2 packets are used for subsequent cells of the same observation packet. If observer parameter wTimeStamp is set to a value greater than 0, a Timestamp Size nibble and the time stamp are appended to the D64MTS packet. If wTimeStamp is 0, only the Timestamp Size nibble, set to a value of 0, is appended.

If an overflow packet is received by the observer, an MERR STPv2 packet is sent, indicating an STPv2 master error, prior to the encapsulation of the probed data into D64MTS and D64 packets.

75.9.2.3.2.2.3.1 Timestamp compression

When wTimeStamp is greater than 0, the number of bits sent in the Timestamp field of D64MTS packets depends on the difference between the time stamp from the current observation packet and the previous time stamp value sent in the STPv2 stream. The previous time stamp may be either itself the time stamp of the previous observation packet or the absolute time stamp sent in an STPv2 synchronization sequence.

Only nibbles which are different between the current and previous time stamps are sent, left-aligned, in the Timestamp field of the D64MTS packet. The field is preceded by a Timestamp Size nibble, which indicates the field size, in nibbles. The encoding of TimeStamp Size is defined in Section 7.2 of MIPI STP specification version 2.00.01.

The following table shows several examples of time stamp compression values.

Table 627. Timestamp compression values

Timestamp value		STP timestampe	
Previous	Current	Timpstamp size	Compressed timestamp
0000_0000_1111_2222h	0000_0000_1111_2222h	0	—
0000_0000_1111_2222h	0000_0000_1111_2234h	2	34h
0000_0000_1234_5678h	0000_0000_1254_5601h	6	545601h
0123_0000_0000_0000h	0226_0000_0002_0030h	16[1]	0226_0000_0002_0030h
0000_0000_0012_3459h	0000_0000_0012_3457h	1	7h[2]

1. The Timestamp Size nibble is encoded for 16 nibbles even though only 15 nibbles are different because there is no encoding for 15 (see MIPI STP specification).
2. It is possible for the time stamp value of the packet being processed to be less than the time stamp of the previous packet. This can occur if an observation packet, which has an earlier time stamp than a packet from another probe, is delayed due to buffering in the observation network.

75.9.2.3.2.3 Observer registers

Observer functions, such as error logging, ATB formatting, and STPv2 encapsulation, are run-time programmable via registers. For a given function, in addition to the specific registers for that function, the register set includes standard identification registers.

See APB memory map.

See "Memory maps/register definitions" in this document for reset values of registers.

75.9.2.3.2.3.1 Error logging registers

Registers related to error logging are summarized in the following table.

Address offset	Name	Type	Retention
0008h	FaultEn	Control	Yes
000Ch	ErrVld	Status	No
0010h	ErrClr	Pulse	No
0014h	ErrLog0	Status	No
0018h	ErrLog1	Status	No
001Ch	ErrLog2	Status	No
0020h	ErrLog3	Status	No
0024h	ErrLog4	Status	No
0028h	ErrLog5	Status	No
002Ch	ErrLog6	Status	No
0030h	ErrLog7	Status	No
0034h	ErrLog8	Status	No
0038h	StallEn	Control	Yes

75.9.2.3.2.3.1.1 FaultEn

When 1-bit register FaultEn is set to 1, error reporting signal Fault is enabled and asserted when register ErrVld is 1. When FaultEn is cleared to 0, signal Fault is driven to 0.

- Bit – (0)

Related signals:

- Fault

75.9.2.3.2.3.1.2 ErrVld

Register ErrVld indicates that the ErrLog registers contain valid error information.

- Bit – (0)

75.9.2.3.2.3.1.3 ErrLog0

Register ErrLog0 logs fields from the NoC packet transport protocol header of the logged packet.

The MSB field indicates the transport protocol version number.

The following table shows the field assignments. All other bits are reserved and read as 0.

Table 628. ErrLog0 field assignments

Bit range	Field name	Type	Access
0	Lock	Status	Read only
4:1	Opc	Status	Read only
10:8	ErrCode	Status	Read only
27:16	Len	Status	Read only
31	Format	Status	Read only

75.9.2.3.2.3.1.3.1 Lock

Register field Lock contains the packet header bit Lock.

75.9.2.3.2.3.1.3.2 Opc

Register field Opc contains the packet header field Opc.

75.9.2.3.2.3.1.3.3 Len1

Register field Len1 contains the packet header field Len1.

75.9.2.3.2.3.1.3.4 ErrCode

Register field ErrCode contains the packet header field ErrCode.

If the packet field does not exist, the register field contains 0.

75.9.2.3.2.3.1.3.5 Format

Register field Format specifies the version of the NoC packet transport protocol.

75.9.2.3.2.3.1.4 ErrLog1

Register Errlog1 contains the least significant part of the NoC packet transport protocol packet header field RouteId of the logged error. Up to 32 bits will be stored, depending on the width of RouteId. The field is right-aligned in the register, and unused bits are read as 0.

The RouteId value can be used to identify the initiator–target mapping associated with the logged error, in order to calculate the complete address.

When decoding RouteId sub-field SeqId, bit index 0 corresponds to the value for packet header field Opc when:

- The transaction comes from an interface with a protocol, such as AHB, AXI, or AMBA, that allows response disorder based on transaction opcodes.
- The NoC supports other interface protocols that do not.

A description of the sub-fields comprising RouteId is provided in related reference documentation.

- Type – Status

75.9.2.3.2.3.1.5 ErrLog2

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co., Ltd.
c8c2e2c2-482-48a8-b251-390d3ab624c1

Register Errlog2 contains the most significant part of the NoC packet transport protocol packet header field RouteId of the logged error. Up to 16 MSBs of RouteId are stored, depending on the width of RouteId. The field is right-aligned in the register, and unused bits are read as 0.

The RouteId value can be used to identify the initiator–target mapping associated with the logged error, in order to calculate the complete address.

A description of the sub-fields comprising RouteId is provided in related reference documentation.

This register is only present if the width of RouteId is greater than 32 bits.

- Type – Status

75.9.2.3.2.3.1.6 ErrLog3

Register Errlog3 contains the least significant part of the NoC packet transport protocol packet header field Addr of the logged error. Up to 32 bits will be stored, depending on the width of Addr. The field is right-aligned in the register and unused bits are read as 0.

Because ErrLog3 only contains the transport address offset within the particular mapped region, its value must be used in conjunction with the local addresses of the initiator–target mapping, in order to reconstitute the complete transaction address. The mapping is identified using RouteId, logged in ErrLog1 and optionally ErrLog2.

75.9.2.3.2.3.1.7 ErrLog4

Register Errlog4 contains the most significant part of the NoC packet transport protocol packet header field Addr of the logged error. Up to 16 MSBs of Addr are stored, depending on the width of Addr. The field is right-aligned in the register and unused bits are read as 0.

Because ErrLog4 only contains the transport address offset within the particular mapped region, its value, concatenated with ErrLog3, must be used in conjunction with the local addresses of the initiator–target mapping in order to reconstitute the complete transaction address. The mapping is identified using RouteId, logged in ErrLog1, and optionally, ErrLog2.

This register is only present if the width of Addr is greater than 32 bits.

75.9.2.3.2.3.1.8 ErrLog5

Register Errlog5 contains the least significant part of the NoC packet transport protocol packet header field User of the logged error.

Up to 32 bits will be stored, depending on the width of User. The field is right-aligned in the register, and unused bits are read as 0.

75.9.2.3.2.3.1.9 ErrLog6

Register Errlog6 contains the midrange part of field User contained in the packet header of the packet transport protocol. Bit User (32) up to bit User (63) will be stored, depending on the width of User. The field is right-aligned in the register, and unused bits are read as 0.

The register is present only if the width of User is greater than 32 bits.

75.9.2.3.2.3.1.10 ErrLog7

Register Errlog7 contains the NoC packet transport protocol packet header field Security of the logged error. The field is right-aligned in the register and unused bits are read as 0.

75.9.2.3.2.3.1.11 ErrLog8

Register Errlog8 contains the most significant part of the NoC packet transport protocol packet header field User of the logged error. Bits User(64) up to User(95) are stored, depending on the width of User. The field is right-aligned in the register, and unused bits are read as 0.

The register is present only if the width of User is greater than 64 bits.

75.9.2.3.2.3.1.12 StallEn

Register StallEn controls the stall mode of the NoC error logger.

When set to 1, error packets (status = ERR) are stalled, that is, queued at the input to the error logger. The packets are only passed into the error logger once the currently logged error packet has been cleared by setting register ErrClr to 1.

When set to 0, only the first error packet is stored in the error logger. Subsequent error packets are lost.

NOTE

The number of error packets which may be queued is 4.

75.9.2.3.2.3.2 ATB endpoints registers

Registers which are used to control the ATB endpoint are summarized in the following table.

Address offset	Name	Type	Retention	Width
0008h	AtbId	Control	Yes	7 bits
000Ch	AtbEn	Control	Yes	1 bit
0010h	SyncPeriod	Control	Yes	3 bits (4:0)

75.9.2.3.2.3.2.1 AtbId

Register AtbId sets the ATB identifier of the observer. The identifier is carried on signal ATId.

Each observer within the NoC must have a unique ID value. The value is used along with probe IDs to identify probes.

NOTE

Values 00h and 70h to 7Fh are reserved for Arm® CoreSight™ debug systems.

75.9.2.3.2.3.2.2 AtbEn

When AtbEn = 1, register AtbEn enables ATB output.

NOTE

Writing AtbEn = 0 is equivalent to a power disconnect on the affected socket: the current transmission of an ATB packet is completed before the ATB output is disabled.

75.9.2.3.2.3.2.3 SyncPeriod

Register SyncPeriod sets the period, in terms of the number of ATB cells sent, between insertion of ATB synchronization cells.

The following table summarizes the relationship between SyncPeriod and the delay between synchronization cells.

Table 629. SyncPeriod and delay between synchronization cells

SyncPeriod value	Period of sync cell insertion
0	No sync cell insertion
1...2	8
3...22	$2^{(SyncPeriod + 1)}$
23...31	Not supported

SyncPeriod is not present when setting parameter debugOutput of the observer to STPv2, since ATB synchronization cells are not generated when using the STPv2 adapter.

75.9.2.3.2.3.3 STPv2 encapsulation registers

Registers relative to STPv2 encapsulation are summarized in the following table.

Address offset	Name	Type	Retention	Width
0008h	AsyncPeriod	Control	Yes	5 bits (4:0)
000Ch	STPv2En	Control	Yes	1 bit (bit 0)

SyncPeriod is not present when setting parameter debugOutput of the observer to STPv2, since ATB synchronization cells are not generated when using the STPv2 adapter.

75.9.2.3.2.3.3.1 AsyncPeriod

Register AsyncPeriod defines the period, in observer clock cycles, between transmission of STPv2 synchronization sequences.

The following table summarizes the relationship between AsyncPeriod and the delay between synchronization sequences.

Table 630. AsyncPeriod and delay between synchronization cells

AsyncPeriod value	Period between synchronization sequences
0	No sync sequence inserted due to AsyncPeriod
1...22	$2^{(SyncPeriod + 1)}$
23...31	2^{23}

SyncPeriod is not present when setting parameter debugOutput of the observer to STPv2, since ATB synchronization cells are not generated when using the STPv2 adapter.

75.9.2.3.2.3.3.2 STPV2En

When STPV2En = 1, register STPV2En enables the STPv2 output.

When STPV2En = 0, the observer debug interface is disabled: neither STPv2 packets nor synchronization sequences are sent.

By default, the reset value is 1 so that STPv2 stream generation is enabled. If it should be enabled explicitly by software, the reset value can be changed to 0 by setting its parameter rstVal (Structure: Parameters: Control Setting).

NOTE

Writing STPv2En =0 has an equivalent effect to a power disconnect on this interface: the STPv2 encapsulation and transmission of an ATB packet will be completed before the observer debug output is disabled.

75.10 Debug and trace under functional reset

Cores Cortex-M7_0, Cortex-M7_1, Cortex-M7_2 and Cortex-M7_HSE under main reset domain support debug and trace over Functional reset.

Cores Cortex-A53_A0, Cortex-A53_A1, Cortex-A53_B0, and Cortex-A53_B1 which are under software resettable domain support retention of debug and trace configuration over functional reset. This is achieved only when Bit[7] (DBG_RST_MSK_0a) of the DEBUG_CONTROL register (see the SRC chapter for details) is set before giving functional reset to the chip. This bit masks debug resets of the cores from functional reset imitating *Cluster cold reset with debug active* as per Cortex A53 Technical Reference Manual (available in References).

Bit DBG_RST_MSK_0a must only be set after enabling Cortex-A53 partition (See "Software Reset Partition Turn-On Flow Chart" in Reset chapter for details).

Under functional reset, trace from Cortex-A53 cores are not available.

NOC probes and observers support only retention of debug configurations under functional reset.

The LLCE's M0-3 cores support debug and trace over functional reset by asserting Bit[8] (DBG_RST_MSK_1a) of the DEBUG_CONTROL register. The CoreSight AHB_APs are on the destructive reset and thus not cleared by a functional reset.

Also, the accelerator related CoreSight debug components (accelerator CTIs, ETF, etc) are on the destructive reset and thus will not be affected by a functional reset. Likewise, the accelerator NoC debug and QoS registers will have their values retained during a functional reset and can only be cleared by a software write to each register or a destructive reset event.

NOTE

Note only debug register access is allowed while functional reset is asserted, trace is not available.

75.11 MDM_AP register descriptions

The debugger has access to the status and control elements implemented as registers in MDM_AP. APSEL (6h) selects the MDM_AP on the Debug Access Port (DAP). These registers provide additional control and status information for typical debug, cross-triggering, and run-control scenarios. The status register fields also provide a way for the debugger to get the updated status of the core without having to initiate a bus transaction across the crossbar switch, thus remaining less intrusive during a debug session.

The MDM_AP is accessible as Debug Access Port (see Debug Access Port (DAP) TAP). The MDM_AP registers are listed in the following table and detailed information about the registers is provided in the following sections.

Certain modules have a Debug mode (IPG DEBUG) that freezes the module when the cores enter Debug mode. If you want to detach dependency of a particular core's Debug mode on entry of modules into Debug mode, then you must write a 1 to the corresponding Cortex_**n**_IP_DBGDIS field in MDM_AP IP_DBG registers. MDM_AP IP_DBG register pertains to Cortex_**n**_IP_DBG registers in memory map. The following table shows the modules that depend on the core's debug state.

Table 631. Dependency of modules on the core's debug

Module	Affected by
DSPI	MDM_AP IP_DBG registers
eDMA	MDM_AP IP_DBG registers
FCCU	MDM_AP IP_DBG registers
FlexCAN	MDM_AP IP_DBG registers
FlexTimer	MDM_AP IP_DBG registers
I2C	MDM_AP IP_DBG registers
PIT	MDM_AP IP_DBG registers
RTC	MDM_AP IP_DBG registers
SBSW	MDM_AP IP_DBG registers
STM	MDM_AP IP_DBG registers
SWT_0	Debug Mode status of Cortex-M7 core 0
SWT_1	Debug Mode status of Cortex-M7 core 1
SWT_2	Debug Mode status of Cortex-M7 core 2
SWT_3	Debug Mode status of Cortex-A53 A0 core
SWT_4	Debug Mode status of Cortex-A53 A1 core
SWT_5	Debug Mode status of Cortex-A53 B0 core
SWT_6	Debug Mode status of Cortex-A53 B1 core

Provided under NDA
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c84625-48...

75.11.1 MDM_AP memory map

MDM_AP base address: 51D0_0600h

Offset	Register	Width (In bits)	Access	Reset value
0h	MDM_AP status (STATUS)	32	RO	0000_0000h
4h	MDM_AP control (CONTROL)	32	RW	0040_0000h
30h	WIR enable (WIR_EN)	32	RW	0000_0000h
38h	WIR release (WIR_REL)	32	RW	0000_0000h
40h	MDM_AP status 2 (STATUS2)	32	RO	0000_0000h
70h	Cortex-A53 A0 debug (CORTEX_A53_A0_IP_DBG)	32	RW	0000_0000h
74h	MDM_AP control 2 (CONTROL2)	32	RW	0000_0000h
78h	Cortex-A53 A1 debug (CA53_A1_IP_DBG)	32	RW	0000_0000h
80h	Cortex-A53 B0 debug (CORTEX_A53_B0_IP_DBG)	32	RW	0000_0000h
88h	Cortex-A53 B1 debug (CORTEX_A53_B1_IP_DBG)	32	RW	0000_0000h
90h	Cortex M7_0 debug (CORTEX_M7_0_IP_DBG)	32	RW	0000_0000h
98h	Cortex M7_1 debug (CORTEX_M7_1_IP_DBG)	32	RW	0000_0000h
A0h	Cortex M7_2 debug (CORTEX_M7_2_IP_DBG)	32	RW	0000_0000h
C0h	DAP enable control (DAP_EN_CTRL)	32	RW	FFFF_FFFFh

75.11.2 MDM_AP status (STATUS)

Offset

Register	Offset
STATUS	0h

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	M7_2_DB...	M7_1_DB...	M7_0_DB...	Reserved				Reserved	CM7_2_S...	CM7_1_S...	CM7_0_S...	Reserved	CM7_2_S...	CM7_1_S...	CM7_0_S...
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	CM7_2_H...	CM7_1_H...	CM7_0_H...	Reserved			JTAG_AC...	CDBGPWR...	CSYSPWR...	READY_F...	Reserved		SYSTEM_...	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under IHS Markup Licensing CONFIDENTIAL PROPRIETARY COMPANY NPI Automotive Technology Co.,Ltd c8c4623e-4821-48a8-b251-39004ab6a4d1

Fields

Field	Function
31 —	Reserved
30 M7_2_DBGRSTRTD	Handshake ssignal for M7_2_DBGRSTRT. If there is no external debug request to the core, the value read after reset will be 1. 0b - Cortex-M7_2 is still in Debug mode 1b - Cortex-M7_2 has returned to Normal mode from Debug mode
29 M7_1_DBGRSTRTD	Handshake signal for M7_1_DBGRSTRT. If there is no external debug request to the core, the value read after reset will be 1. 0b - Cortex-M7_1 is still in Debug mode 1b - Cortex-M7_1 has returned to Normal mode from Debug mode
28 M7_0_DBGRSTRTD	Handshake signal for M7_0_DBGRSTRT. If there is no external debug request to the core, the value read after reset will be 1. 0b - Cortex-M7_0 is still in Debug mode 1b - Cortex-M7_0 has returned to Normal mode from Debug mode
27-24 —	Reserved
23 —	Reserved
22 CM7_2_SLEEPING	Cortex-M7_2 Asserted Sleeping 0b - The processor is running or wants to leave sleep mode 1b - The processor and ETM are ready to enter a low-power state
21 CM7_1_SLEEPING	Cortex-M7_1 Asserted Sleeping 0b - The processor is running or wants to leave sleep mode 1b - The processor and ETM are ready to enter a low-power state
20 CM7_0_SLEEPING	Cortex-M7_0 Asserted Sleeping 0b - The processor is running or wants to leave sleep mode 1b - The processor and ETM are ready to enter a low-power state
19 —	Reserved
18 CM7_2_SLEEPDEEP	Cortex-M7_2 Asserted SLEEPDEEP 0b - Inactive value

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - The processor and ETM are ready to enter a low-power state and the wake-up time is not critical. Only active when STATUS[CM7_2_SLEEPING]=1
17 CM7_1_SLEEPDEEP	Cortex-M7_1 Asserted SLEEPDEEP 0b - Inactive value 1b - The processor and ETM are ready to enter a low-power state and the wake-up time is not critical. Only active when STATUS[CM7_1_SLEEPING]=1
16 CM7_0_SLEEPDEEP	Cortex-M7_0 Asserted SLEEPDEEP 0b - Inactive value 1b - The processor and ETM are ready to enter a low-power state and the wake-up time is not critical. Only active when STATUS[CM7_0_SLEEPING]=1
15 —	Reserved
14 CM7_2_HALTED	Cortex-M7_2 is halted due to Debug mode entry. 0b - Normal mode 1b - Debug mode
13 CM7_1_HALTED	Cortex-M7_1 is halted due to Debug mode entry. 0b - Normal mode 1b - Debug mode
12 CM7_0_HALTED	Cortex-M7_0 is halted due to Debug mode entry. 0b - Normal mode 1b - Debug mode
11-9 —	Reserved
8 JTAG_ACTIVE	Status of Reset to AUX TAPs 0b - TRST of AUX TAPs are asserted 1b - TRST of AUX TAPs are deasserted
7 CDBGPWRUPREQ	Status of CDBGPWRUPREQ from SWJ-DP When the debugger asserts CDBGPWRUPREQ in SWJ-DP requesting access to APs, then this field becomes 1. This field can be used by internal masters to detect debugger request for attachment onto the SOC. 0b - CDBGPWRUPREQ is deasserted 1b - CDBGPWRUPREQ is asserted
6	Status of CSYSPWRUPREQ from SWJ-DP

Table continues on the next page...

Table continued from the previous page...

Field	Function
CSYSPWRUPREQ	When the debugger asserts CSYSPWRUPREQ in SWJ-DP requesting access to APs, then this field becomes 1. 0b - CSYSRUPREQ is deasserted 1b - CSYSPWRUPREQ is asserted
5 READY_FOR_DBG	Status of ready for debug field is asserted by Boot Sequence routine. 0b - Debugger to wait until the fields get asserted 1b - Debugger can go ahead to run its routine
4-3 —	Reserved
2 SYSTEM_RESET	System Reset State 0b - System is not in reset 1b - System is in reset
1-0 —	Reserved

75.11.3 MDM_AP control (CONTROL)

Offset

Register	Offset
CONTROL	4h

Function

A system destructive reset in this register allows the debugger to give the destructive reset to the system. The debugger also loses connection with the system with this reset. A system functional reset in this register allows the debugger to hold the system in functional reset.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	M7_2_DB...	M7_1_DB...	M7_0_DB...	Reserved			MSK_FCC...	MSK_DBG...	Reserved	ETR_OVE...	TPIU_OV...	CA53_B1...	CA53_B0...	CA53_A1...	CA53_A0...
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DIS_PO R...	MSK_LOC...	MSK_LOC...	MSK_LOC...	Reserved	CM7_2_E...	CM7_1_E...	CM7_0_E...	ETR_HAN...	ETR_HAN...	SYSFUNC...	SYSRESE...	DBG_SET...	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
IC Manufacturing 482-48a8-b251-390d3ab624c1

Fields

Field	Function
31 —	Reserved
30 M7_2_DBGRSTRT	Cortex-M7 Debug Restart Input to Cortex-M7_2 Core DBGRESTRT is used to bring the processor out of debug halt state. The processor acknowledges DBGRESTRT by asserting M7_2_DBGRSTRTD. So, to finish the handshake and restart the core, this field must be cleared. 0b - Inactive value 1b - Request asserted to restart the core
29 M7_1_DBGRSTRT	Cortex-M7 Debug Restart Input to Cortex-M7_1 Core DBGRESTRT is used to bring the processor out of debug halt state. The processor acknowledges DBGRESTRT by asserting M7_1_DBGRSTRTD. So, to finish the handshake and restart the core, this field must be cleared. 0b - Inactive value 1b - Request asserted to restart the core
28 M7_0_DBGRSTRT	Cortex-M7 Debug Restart Input to Cortex-M7_0 Core DBGRESTRT is used to bring the processor out of debug halt state. The processor acknowledges DBGRESTRT by asserting M7_0_DBGRSTRTD. So, to finish the handshake and restart the core, this field must be cleared. 0b - Inactive value 1b - Request asserted to restart the core
27-25 —	Reserved
24 MSK_FCCU_RST_TRIGGER	Mask Reset Trigger from FCCU This field is used to mask reset trigger from FCCU. If this field is not set then a reset would occur on FCCU reset trigger. This masking feature enables inspection of the system to determine the source. Some parts of the system may not be operating correctly. The only course of action following inspection is to reset the chip. 0b - FCCU reset trigger is not masked 1b - FCCU reset trigger is masked
23 MSK_DBG_FAULT	Mask Debug Faults This field is programmed via debug software while initiating self-hosted debug, it is used to mask debug faults to FCCU. 0b - Debug faults are not masked 1b - Debug faults are masked

Table continues on the next page...

Table continued from the previous page...

Field	Function
22 —	Reserved
21 ETR_OVERRIDE	ETR Override When ETR is not the selected trace sink target, override trace response. 0b - ETR trace response is not overridden, and the trace response is generated by the ETR (default) 1b - ETR trace response is overridden (default)
20 TPIU_OVERRIDE	TPIU Override When TPIU is not the selected trace sink target, override trace response. 0b - TPIU trace response is not overridden, and the trace response is generated by TPIU (default) 1b - TPIU trace response is overridden and forced asserted
19 CA53_B1_EDBGREQ	EDBGREQ Input for Cortex-A53 B1 When the core goes into debug state, it acknowledges with HALTED output signal (MDM status register2[3]). If the core is in Stop mode, this field is used to wake up the core and transition to Halted state. 0b - Inactive value 1b - Request asserted to keep core in Debug mode
18 CA53_B0_EDBGREQ	EDBGREQ Input for Cortex-A53 B0 When the core goes into debug state, it acknowledges with HALTED output signal (MDM status register2[2]). If the core is in Stop mode, this field is used to wake up the core and transition to Halted state. 0b - Inactive value 1b - Request asserted to keep core in Debug mode
17 CA53_A1_EDBGREQ	EDBGREQ Input for Cortex-A53 A1 When the core goes into debug state, it acknowledges with HALTED output signal (MDM status register2[1]). If the core is in Stop mode, this field is used to wake up the core and transition to Halted state. 0b - Inactive value 1b - Request asserted to keep core in Debug mode
16 CA53_A0_EDBGREQ	EDBGREQ Input for Cortex-A53 A0 When the core goes into debug state, it acknowledges with HALTED output signal (MDM status register2[0]). If the core is in Stop mode, this field is used to wake up the core and transition to Halted state. 0b - Inactive value

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Request asserted to keep core in Debug mode
15 DIS_POR_WDOG_MSK	POR Watchdog Timer This field disables masking of POR WDOG Timer that must timeout if assertion of functional reset via CONTROL[SYSFUNCRST] exceeds 30ms. 0b - Disables POR WDOG timer (by default disabled) 1b - Enables POR WDOG timer
14 MSK_LOCKSTEP_ALARM_CM72	Mask RCCU Alarm for Cortex M7_2 This field is used to mask RCCU alarm for Cortex M7_2. This field is intended to mask any lockstep error that could occur during Debug mode. 0b - Disables masking 1b - Enables masking
13 MSK_LOCKSTEP_ALARM_CM71	Mask RCCU Alarm for Cortex M7_1 This field is used to mask RCCU alarm for Cortex M7_1. This field is intended to mask any lockstep error that could occur during Debug mode. 0b - Disables masking 1b - Enables masking
12 MSK_LOCKSTEP_ALARM_CM70	Mask RCCU Alarm for Cortex M7_0 This field is used to mask RCCU alarm for Cortex M7_0. This field is intended to mask any lockstep error that could occur during Debug mode. 0b - Disables masking 1b - Enables masking
11 —	Reserved
10 CM7_2_EDBGREQ	EDBGREQ Input for Cortex-M7_2 When the core goes into debug state, it acknowledges with HALTED output signal (MDM status register [14]). If the core is in Stop mode, this field is used to wake up the core and transition to Halted state. 0b - Inactive value 1b - Request asserted to keep core in Debug mode
9 CM7_1_EDBGREQ	DEDBGREQ Input for Cortex-M7_1 When the core goes into debug state, it acknowledges with HALTED output signal (MDM status register [13]). If the core is in Stop mode, this field is used to wake up the core and transition to Halted state. 0b - Inactive value 1b - Request asserted to keep core in Debug mode

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 CM7_0_EDBGREQ	EDBGREQ Input for Cortex-M7_0 When the core goes into debug state, it acknowledges with HALTED output signal (MDM status register [12]). If the core is in Stop mode, this field is used to wake up the core and transition to Halted state. 0b - Inactive value 1b - Request asserted to keep core in Debug mode
7 ETR_HANDSHAKE_1	ETR Handshake_1 After Functional reset de-assertion to reinitiate the pending transactions of ETR, this field must be set to 1. If the ETR_HANDSHAKE_0 field is set to 1, clear this field before asserting functional reset. 0b - Inactive value 1b - Enables ETR
6 ETR_HANDSHAKE_0	ETR Handshake_0 When ETR is being used, handshake with RGM must be done before asserting Functional reset to stall the ETR transactions via NoC. Asserting this field enables the handshake mechanism. 0b - Disables ETR handshake with RGM 1b - Enables ETR handshake with RGM
5 SYSFUNCRST	System Functional Reset Asserts functional reset to the chip. The debugger maintains the connection with the chip. This field must be programmed only via a debugger. 0b - Deasserts functional reset 1b - Asserts functional reset
4 SYSRESETREQ	System Destructive Reset Asserts destructive reset to the chip. The debugger loses the connection. When this field is reset, the entire system comes out of reset. This field must be programmed only via a debugger. 0b - Deasserts destructive reset 1b - Asserts destructive reset
3 DBG_SETUP_DONE	BootROM - Debugger Handshake field, used with READY_FOR_DBG. Debugger writes to this field after its setup is over to complete the handshake with debugger. 0b - Inactive value 1b - Debug setup is complete
2-0 —	Reserved

Provided by NXP
COMPANY CONFIDENTIAL
Beijing CUBE Automotive Technology Co., Ltd.
c8c4625-921-48a8-b251-39d3ab64c1

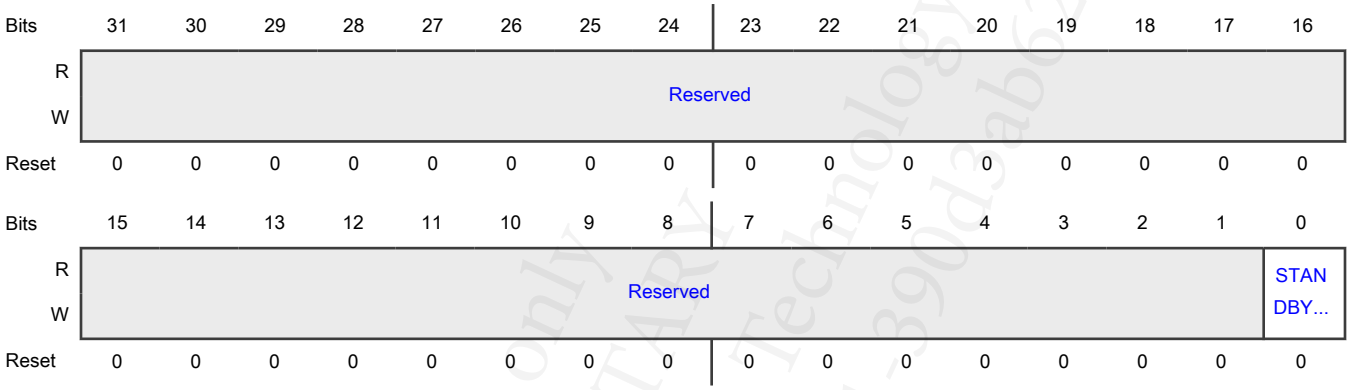
75.11.4 WIR enable (WIR_EN)

Offset

Register	Offset
WIR_EN	30h

Function
MDM_AP standby debug enable

Diagram



Fields

Field	Function
31-1 —	Reserved
0 STANDBY_EN	Setting this field enables debug through Standby mode.

75.11.5 WIR release (WIR_REL)

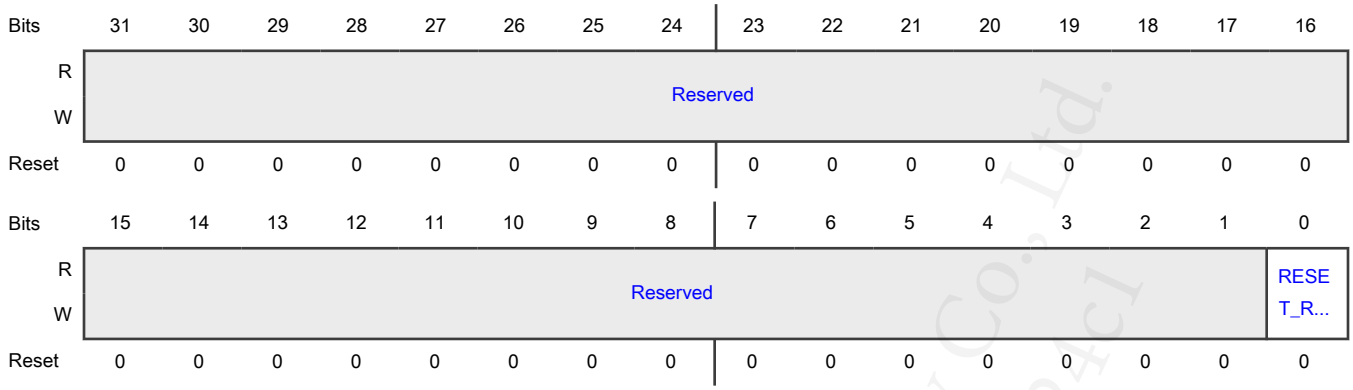
Offset

Register	Offset
WIR_REL	38h

Function
MDM_AP reset release register

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-4aa8-b251-390d3ab624c1

Diagram



Fields

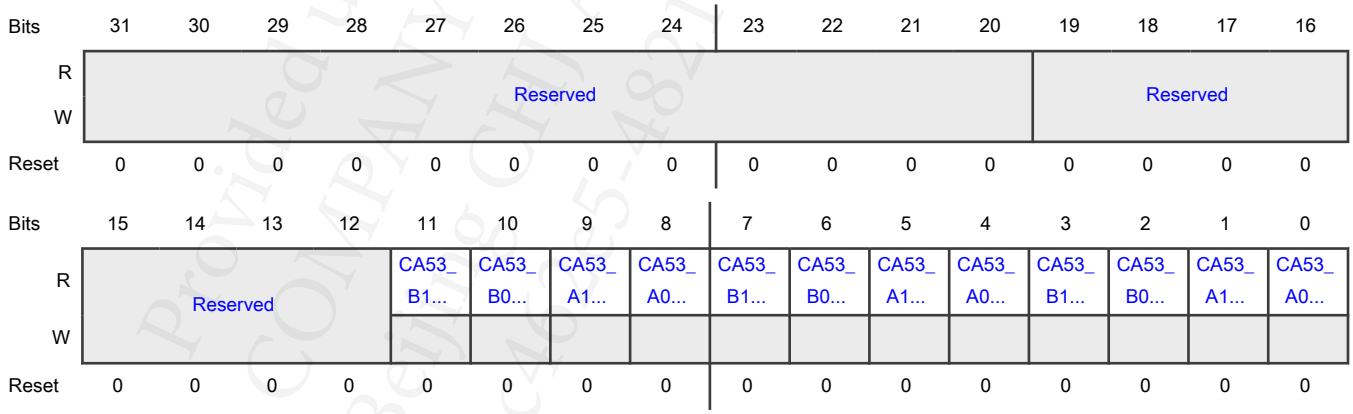
Field	Function
31-1 —	Reserved
0 RESET_REL	Reset release When debug through Standby mode is enabled, then on Standby exit, set this field to 1 to lift the chip out of functional reset after programming the debug components.

75.11.6 MDM_AP status 2 (STATUS2)

Offset

Register	Offset
STATUS2	40h

Diagram



Provided under NDA Only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
c8c4e5-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-20 —	Reserved
19-16 —	Reserved
15-12 —	Reserved
11 CA53_B1_WFE	MC_ME Low-Power Mode Entry using Cortex-A53 B1 Wait for Event (WFE) Indicates whether core is in WFE low-power state. 0b - Core not in WFE low-power state 1b - Core in WFE low-power state
10 CA53_B0_WFE	MC_ME Low-Power Mode Entry using Cortex-A53 B0 WFE Indicates whether core is in WFE low-power state. 0b - Core not in WFE low-power state 1b - Core in WFE low-power state
9 CA53_A1_WFE	MC_ME Low-Power Mode Entry using Cortex-A53 A1 WFE Indicates whether core is in WFE low-power state. 0b - Core not in WFE low-power state 1b - Core in WFE low-power state
8 CA53_A0_WFE	MC_ME Low-Power Mode Entry using Cortex-A53 A0 WFE (Wait For Event) Indicates whether core is in WFE low-power state. 0b - Core not in WFE low-power state 1b - Core in WFE low-power state
7 CA53_B1_WFI	MC_ME Low-Power Mode Entry using Cortex-A53 B1 WFI Indicates whether core is in WFI low-power state. If core is in WFI state, the value read after reset will be 1. 0b - Core not in WFI low-power state 1b - Core in WFI low-power state
6 CA53_B0_WFI	MC_ME Low-Power Mode Entry using Cortex-A53 B0 WFI Indicates whether core is in WFI low-power state. If core is in WFI state, the value read after reset will be 1. 0b - Core not in WFI low-power state

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1b - Core in WFI low-power state
5 CA53_A1_WFI	MC_ME Low-Power Mode Entry using Cortex-A53 A1 WFI Indicates whether core is in WFI low-power state. If core is in WFI state, the value read after reset will be 1. 0b - Core not in WFI low-power state 1b - Core in WFI low-power state
4 CA53_A0_WFI	MC_ME Low-Power Mode Entry using Cortex-A53 A0 WFI (Wait For Interrupt) Indicates whether core is in WFI low-power state. If core is in WFI state, the value read after reset will be 1. 0b - Core not in WFI low-power state 1b - Core in WFI low-power state
3 CA53_B1_DBG MODE	Indicates that the Cortex-A53 B1 is halted due to Debug mode entry. 0b - External debug request not acknowledged 1b - External debug request acknowledged
2 CA53_B0_DBG MODE	Indicates that the Cortex-A53 B0 is halted due to Debug mode entry. 0b - External debug request not acknowledged 1b - External debug request acknowledged
1 CA53_A1_DBG MODE	Indicates that the Cortex-A53 A1 is halted due to Debug mode entry. 0b - External debug request not acknowledged 1b - External debug request acknowledged
0 CA53_A0_DBG MODE	Indicates that the Cortex-A53 A0 is halted due to Debug mode entry. 0b - External debug request not acknowledged 1b - External debug request acknowledged

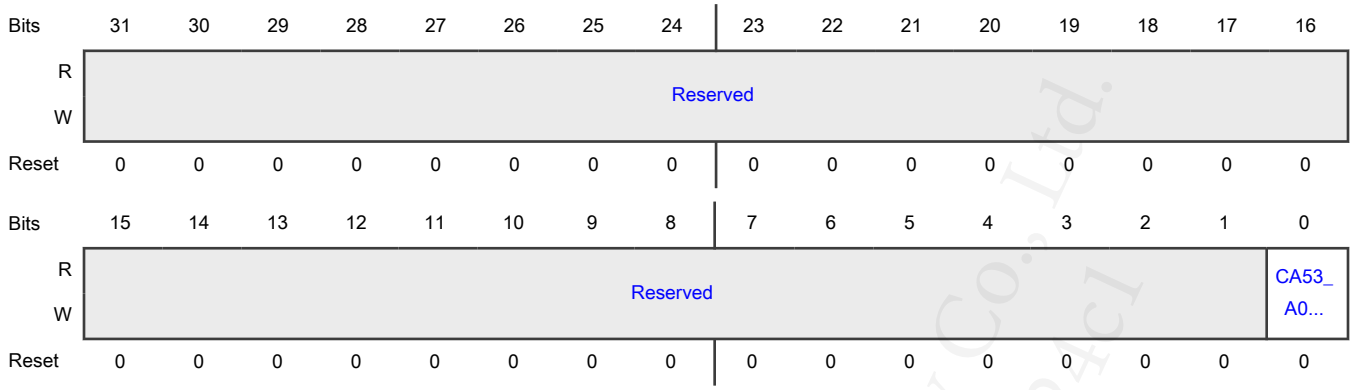
75.11.7 Cortex-A53 A0 debug (CORTEX_A53_A0_IP_DBG)

Offset

Register	Offset
CORTEX_A53_A0_IP_DBG	70h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHU Automotive Technology Co., Ltd.
c8c462e5-42fa-4a8a-9251-390d3ab624c1

Diagram



Fields

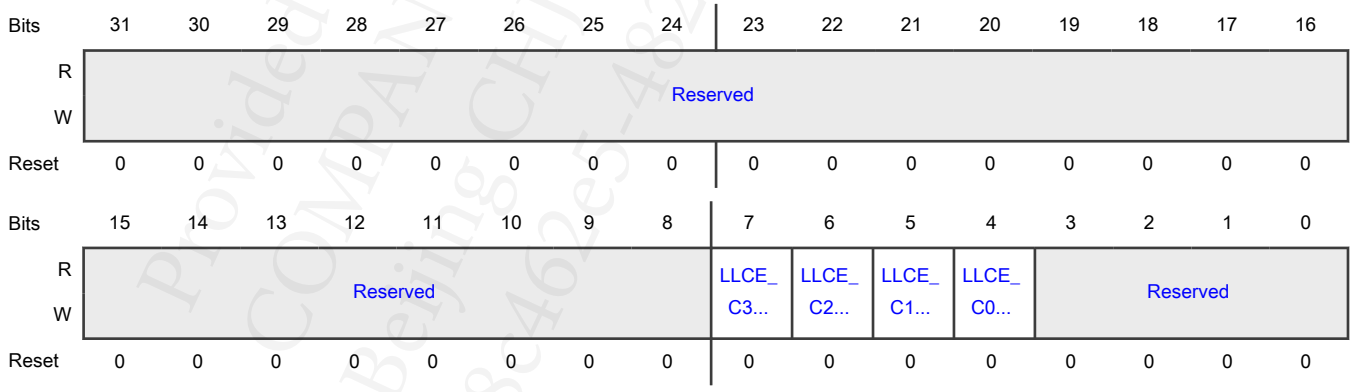
Field	Function
31-1 —	Reserved
0 CA53_A0_IP_DBGDIS	If Cortex-A53 A0 enters Debug mode, but you don't want the modules to enter Debug mode, you must write a 1 to this field. 0b - Modules also enter Debug mode when Cortex-A53 A0 enters Debug mode. 1b - Modules do not enter Debug mode when Cortex-A53 A0 enters Debug mode.

75.11.8 MDM_AP control 2 (CONTROL2)

Offset

Register	Offset
CONTROL2	74h

Diagram



Provided under NDA
COMPANY PROPRIETARY
Beijing CHI Automotive Technology Co.,Ltd.
c8462e5-4821-48a8-b251-390d3ab624c1

Fields

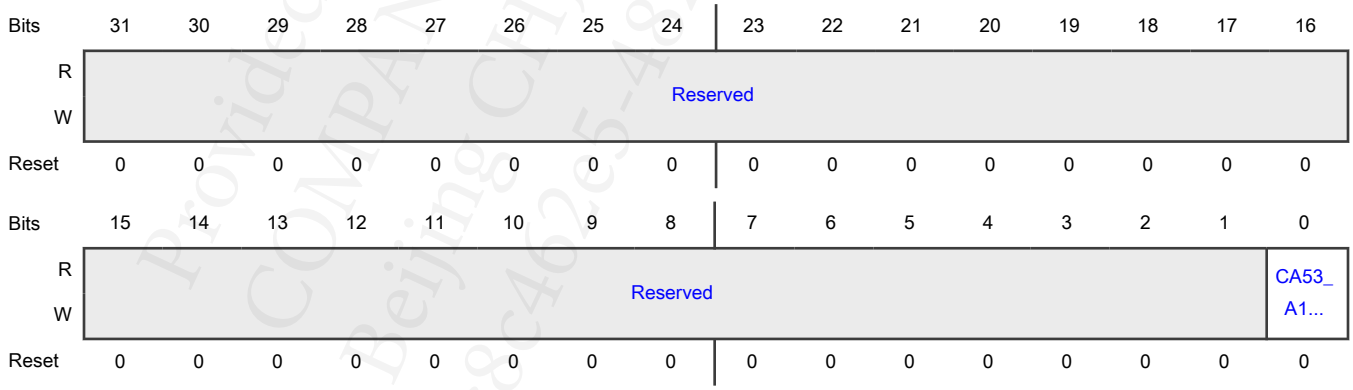
Field	Function
31-8 —	Reserved
7 LLCE_C3_EDBGRQ	LLCE C3 EDBGRQ Drives EDBGRQ, external debug request, input of the M3 processor. When asserted, forces the processor to enter the debug state.
6 LLCE_C2_EDBGRQ	LLCE C2 EDBGRQ Drives EDBGRQ, external debug request, input of the M2 processor. When asserted, forces the processor to enter the debug state.
5 LLCE_C1_EDBGRQ	LLCE C1 EDBGRQ Drives EDBGRQ, external debug request, input of the M1 processor. When asserted, forces the processor to enter the debug state.
4 LLCE_C0_EDBGRQ	LLCE C0 EDBGRQ Drives EDBGRQ, external debug request, input of the M0 processor. When asserted, forces the processor to enter the debug state.
3-0 —	Reserved

75.11.9 Cortex-A53 A1 debug (CA53_A1_IP_DBG)

Offset

Register	Offset
CA53_A1_IP_DBG	78h

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-300d3ab624c1

Fields

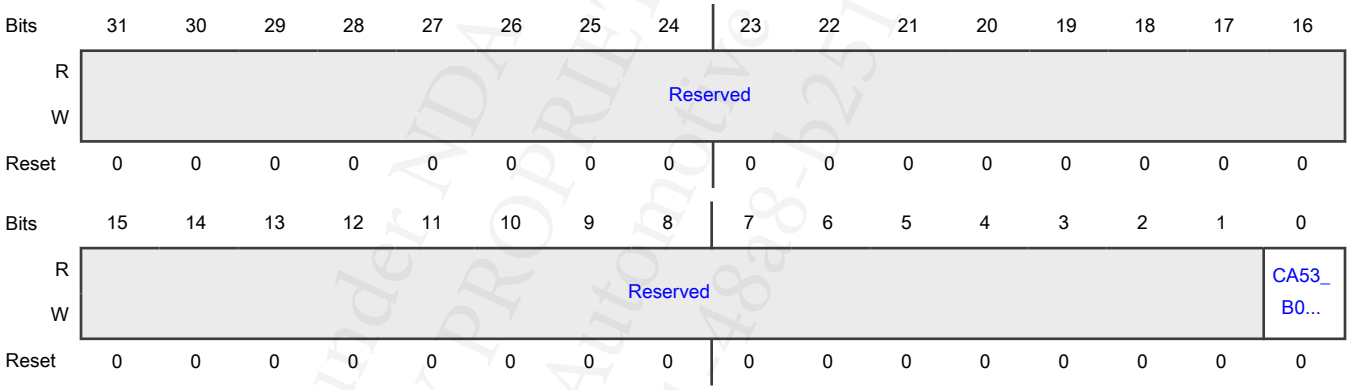
Field	Function
31-1 —	Reserved
0 CA53_A1_IP_DBGDIS	If Cortex-A53 A1 enters Debug mode, but you don't want the modules to enter Debug mode, you must write a 1 to this field. 0b - Modules also enter Debug mode when Cortex-A53 A1 enters Debug mode 1b - Modules do not enter Debug mode when Cortex-A53 A1 enters Debug mode

75.11.10 Cortex-A53 B0 debug (CORTEX_A53_B0_IP_DBG)

Offset

Register	Offset
CORTEX_A53_B0_IP_DBG	80h

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CA53_B0_IP_DBGDIS	If Cortex-A53 B0 enters Debug mode, but you don't want the modules to enter Debug mode, you must write a 1 to this field. 0b - Modules also enter Debug mode when Cortex-A53 B0 enters Debug mode 1b - Modules do not enter Debug mode when Cortex-A53 B0 enters Debug mode

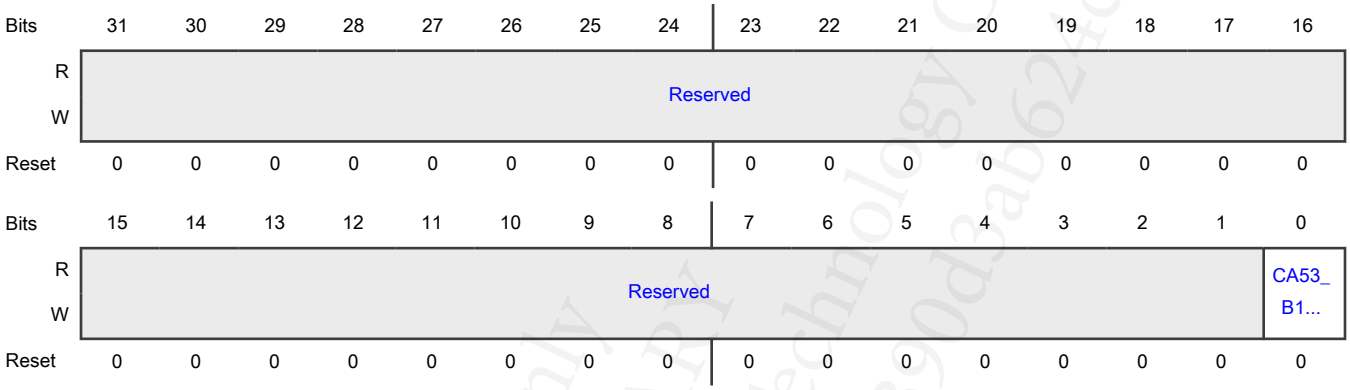
Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e5-4821-48a8-b251-390d8ab62ac1

75.11.11 Cortex-A53 B1 debug (CORTEX_A53_B1_IP_DBG)

Offset

Register	Offset
CORTEX_A53_B1_IP_DBG	88h

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CA53_B1_IP_DBGDIS	If Cortex-A53 B1 enters Debug mode, but you don't want the modules to enter Debug mode, you must write a 1 to this field. 0b - Modules also enter Debug mode when Cortex-A53 B1 enters Debug mode 1b - Modules do not enter Debug mode when Cortex-A53 B1 enters Debug mode

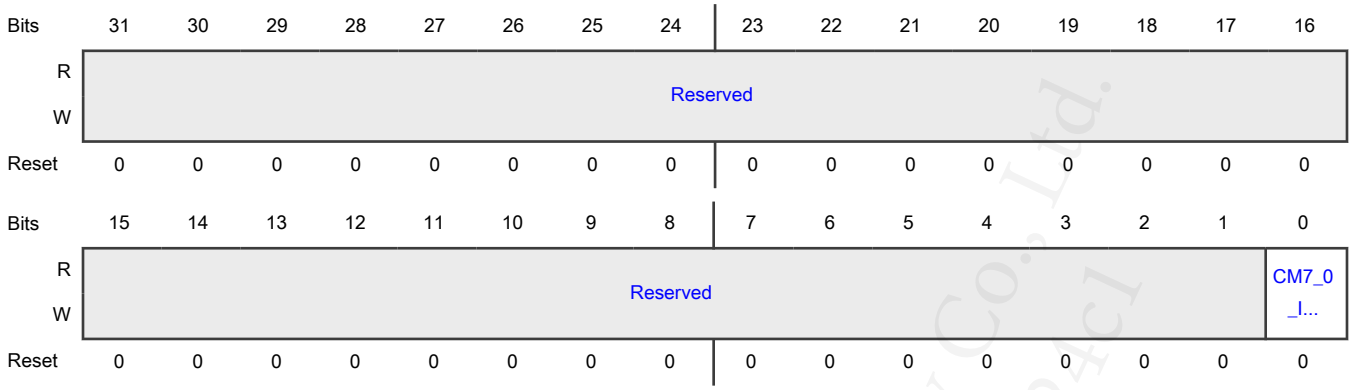
75.11.12 Cortex M7_0 debug (CORTEX_M7_0_IP_DBG)

Offset

Register	Offset
CORTEX_M7_0_IP_DBG	90h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-481a-48a8-b251-390d3ab624c1

Diagram



Fields

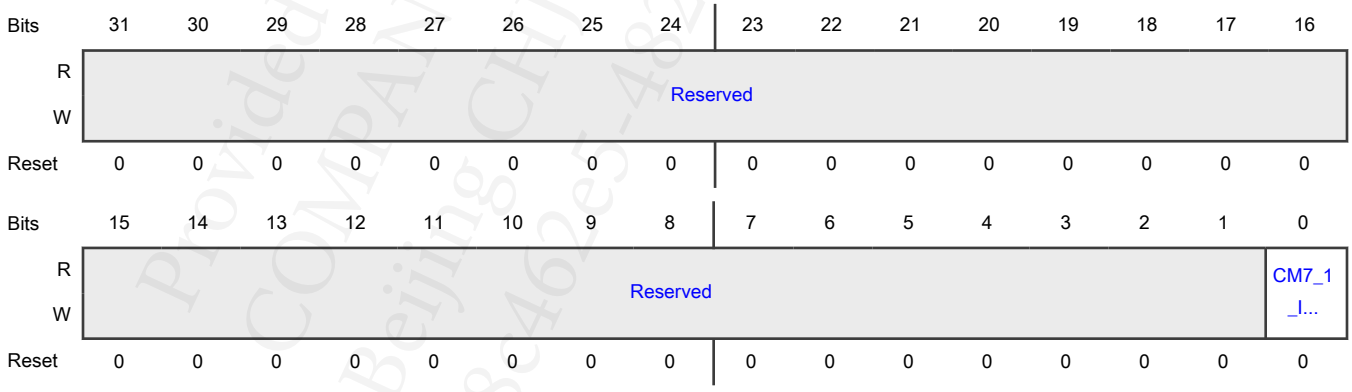
Field	Function
31-1 —	Reserved
0 CM7_0_IP_DBGDIS	If Cortex-M7_0 enters Debug mode, but you don't want the modules to enter Debug mode, you must write a 1 to this field. 0b - Modules also enter Debug mode when Cortex-M7_0 enters Debug mode 1b - Modules do not enter Debug mode when Cortex-M7_0 enters Debug mode

75.11.13 Cortex M7_1 debug (CORTEX_M7_1_IP_DBG)

Offset

Register	Offset
CORTEX_M7_1_IP_DBG	98h

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHI Automotive Technology Co.,Ltd.
c8462e5-4821-48a8-b251-390d3ab624c1

Fields

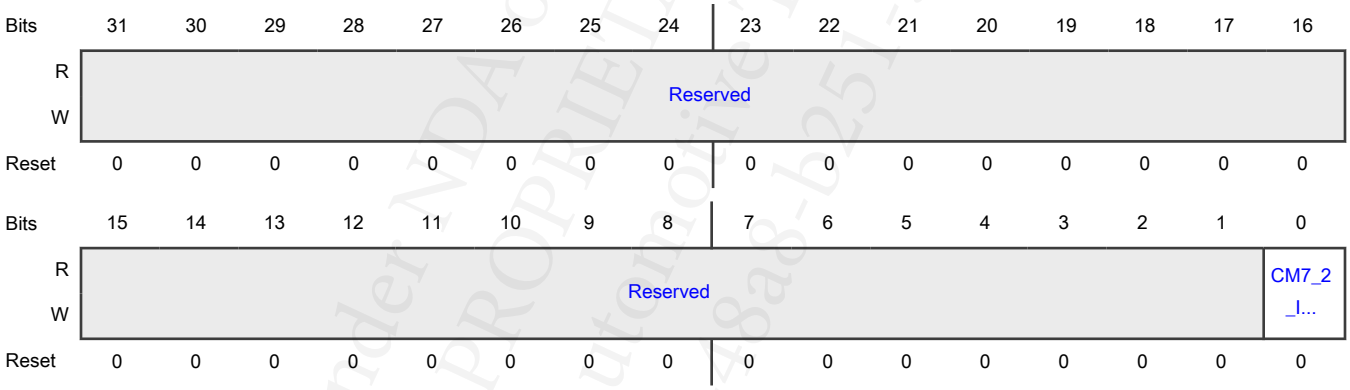
Field	Function
31-1 —	Reserved
0 CM7_1_IP_DBGDIS	If Cortex-M7_1 enters Debug mode, but you don't want the modules to enter Debug mode, you must write a 1 to this field. 0b - Modules also enter Debug mode when Cortex-M7_1 enters Debug mode 1b - Modules do not enter Debug mode when Cortex-M7_1 enters Debug mode

75.11.14 Cortex M7_2 debug (CORTEX_M7_2_IP_DBG)

Offset

Register	Offset
CORTEX_M7_2_IP_DBG	A0h

Diagram



Fields

Field	Function
31-1 —	Reserved
0 CM7_2_IP_DBGDIS	If Cortex-M7_2 enters Debug mode, but you don't want the modules to enter Debug mode, you must write a 1 to this field. 0b - Modules also enter Debug mode when Cortex-M7_2 enters Debug mode 1b - Modules do not enter Debug mode when Cortex-M7_2 enters Debug mode

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462625-4821-48a8-b251-390d3ab62ac1

75.11.15 DAP enable control (DAP_EN_CTRL)

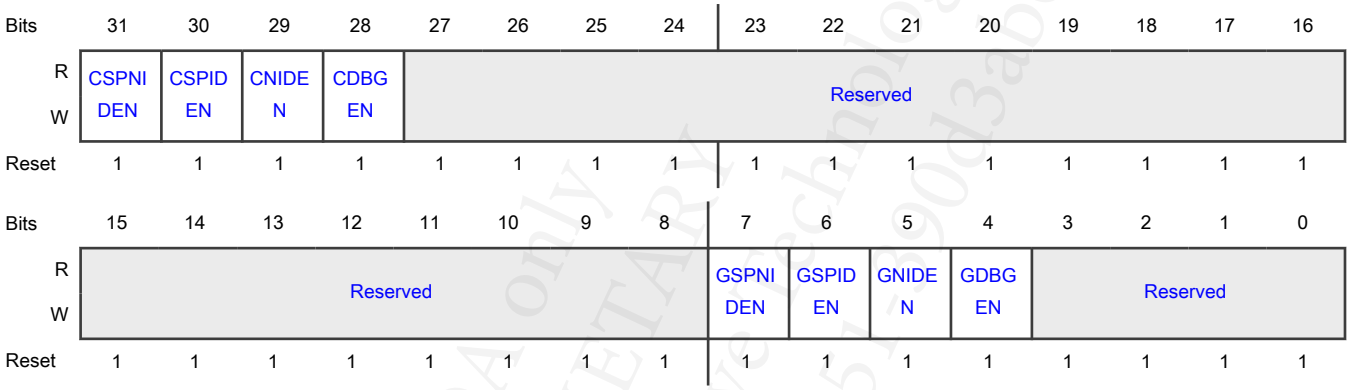
Offset

Register	Offset
DAP_EN_CTRL	C0h

Function

Controls the standard 4-pin Coresight authentication of debug modules. Authentication controls are separate for cores (including ETM,ITM, CTIs, and so on.) and the rest of the chip. It allows you to debug and trace other sources exempting the cores.

Diagram



Fields

Field	Function
31 CSPNIDEN	Controls SPNIDEN of debug blocks coupled with cores subsystem (ETM, ITM, CTIs). Secure privileged noninvasive debug enable. 0b - Disabled 1b - Enabled
30 CSPIDEN	Controls SPIDEN of debug blocks coupled with cores subsystem (ETM, ITM, CTIs). Secure privileged invasive debug enable. 0b - Disabled 1b - Enabled
29 CNIDEN	Controls NIDEN of debug blocks coupled with cores subsystem (ETM, ITM, CTIs). Noninvasive debug enable. 0b - Disabled 1b - Enabled
28 CDBGEN	Controls DBGEN of debug blocks coupled with cores subsystem (ETM, ITM, CTIs). Invasive debug enable.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - Disabled 1b - Enabled
27-8 —	Reserved
7 GSPNIDEN	Controls SPNIDEN of NoC Probes/Observers. Secure privileged noninvasive debug enable. 0b - Disabled 1b - Enabled
6 GSPIDEN	Controls SPIDEN of NoC Probes/Observers. Secure privileged invasive debug enable. 0b - Disabled 1b - Enabled
5 GNIDEN	Controls NIDEN of NoC Probes/Observers, CTIs. Noninvasive debug enable. 0b - Disabled 1b - Enabled
4 GDBGEN	Controls DBGEN of AHB_AP, AXI_AP, CTIs, and DEVICEEN of APB_AP. Invasive debug enable 0b - Disabled 1b - Enabled
3-0 —	Reserved

75.12 Debug resets

The debug subsystem receives the following sources of reset:

- POR reset the complete debug logic.
- JCOMP pin resets the TAP modules (JTAGC and SWJ-DP).
- Debug reset (CDBGRSTREQ bit within the SWJ-DP CTRL/STAT register) is not used.

Conversely, the debug subsystem can generate a system reset using these mechanisms:

- System destructive reset in the MDM AP control register that allows the debugger to give the destructive reset to the system. The debugger loses connection with the system with this reset.
- System functional reset in the MDM AP control register that allows the debugger to hold the system in functional reset.

To program various debug registers, the functional clocks must be enabled. All debug and trace components must be on Destructive Reset.

75.13 Security

The Hardware Security Engine (HSE):

- Provides chip security

- Controls access to the chip's debug featurese)

Authentication by HSE must be complete before handshake using CDBGPWRUPREQ and CDBGPRWUPACK via SWJ-DP.

75.13.1 Arm authentication signals

Authentication interface aims to restrict access to debug and trace functionality by controlling the Arm debug authentication signals. (For more details on these signals, see CoreSight v2.0 Architecture Specification). The Arm authentication signals are shown in the following table.

Table 632. Arm authentication signals

Control signal name	Meaning
DBGEN	Invasive debug enable
NIDEN	Non invasive debug enable
SPIDEN	Secure invasive debug enable
SPNIDEN	Secure non-invasive debug enable

For information on control, see DAP enable control register.

75.13.2 Security fuses

HSE offers a range of security levels and blocks the access to debug based on eFuse configuration and authentication.

75.14 Programming checklists

75.14.1 JTAG switching sequence

75.14.2 FCCU considerations

The Fault Collection and Control Unit (FCCU) monitors whether a core enters Debug mode while in normal operation. This fault is monitored on NCF[59].

In case of self-hosted debug, it is recommended that CONTROL[MSK_DBG_FAULT] must be asserted before entering Debug mode. This will mask NCF[59] and no fault is triggered to FCCU.

75.14.3 Lockstep considerations

In Lockstep mode, the two Cortex-A53 clusters operate in the delayed lockstep configuration, that is, core0 of cluster0 (Cortex-A53 A0) is in Lockstep mode with core0 of cluster1 (Cortex-A53 B0) and similarly for core1 (Cortex-A53 A1 and Cortex-A53 B1). Only cluster0 is available in the system for the debugger while cluster1 works as a checker or redundant module. The debug configurations of cluster1 and its cores must not be programmed in this mode, as when cluster 0 (Cortex-A53 A0 or Cortex-A53 A1) is in Debug mode, the other lockstep core also enters in Debug mode without any software intervention. Therefore, the trace is generated by both the clusters observable on the ATB bus.

75.15 Pin interface

A summary of the number of functional and power pins for the debug interface is shown in the following table.

Table 633. Pin Interface

Pin type	Pins	Number of pins (balls)[1]	Nominal voltage
Functional JTAG	JCOMP, TCK, TMS, TDI, TDO	5 (only TDI and TDO can be multiplexed with GPIO or other functions)	3.3 V, 5 V
Functional Aurora (high-speed serial trace)	AUR_TX*x*_p, AUR_TX*x*_n	10 (8 LVDS outputs, 2 LVDS clock inputs)	—
	AUR_RX_CLK_p, AUR_RX_CLK_n		—
Power	VDD_LV_PLL_AUR	1	0.8 V
	VDD_HV_PLL_AUR	1	1.8 V
	VDD_IO_AUR	1 to 2	1.8 V
Ground	VSS	Shared with common chip VSS pins	0 V

1. The terms pins and balls are used interchangeably. Some chip packages have pins and some packages have balls.

75.15.1 Debug port pin descriptions

Table 634. Debug port pins

Pin name	JTAG debug port		Internal pullup/down
	Type	Description	
JCOMP	I	JCOMP pin	Pulldown
TMS	I	JTAG test mode selection (TMS)	Pullup
TCK/SWCLK	I	JTAG test clock (TCK)	Pulldown
TDI	I	JTAG test data input (TDI)	Pullup
TDO	O	JTAG test data output (TDO)	NC

75.15.2 Trace port pin descriptions

This chip generates the trace via the Trace Port Interface Unit (TPIU) that transmits the trace data out of the MCU over a high-speed serial - Aurora trace port. The trace port consists of Aurora reference clock (AUR_CLK_p and AUR_CLK_n) input port and four pairs of LVDS TXn[3:0] ports for trace output.

Table 635. Trace port pins

Pin name	Type	Description	Internal pullup/down
AUR_TX0_p	O	Aurora Transmit lane 0 positive	NC
AUR_TX0_n	O	Aurora Transmit lane 0 negative	NC
AUR_TX1_p	O	Aurora Transmit lane 1 positive	NC
AUR_TX1_n	O	Aurora Transmit lane 1 negative	NC
AUR_TX2_p	O	Aurora Transmit lane 2 positive	NC
AUR_TX2_n	O	Aurora Transmit lane 2 negative	NC
AUR_TX3_p	O	Aurora Transmit lane 3 positive	NC

Table continues on the next page...

Table 635. Trace port pins (continued)

Pin name	Type	Description	Internal pullup/down
AUR_TX3_n	O	Aurora Transmit lane 3 negative	NC
AUR_CLK_p	I	Aurora reference Clock input	NC
AUR_CLK_n	I	Aurora reference Clock input	NC

NOTE

The RX pins are by default floating and not pulled inside. An internal active pulldown exists only when you enable RX via software (IBE).

The Aurora PLL (AURORA_EXT_CLK) uses the 100 MHz clock input clock (AURORA_REF_CLK) as reference.

Table 636. Aurora transmit frequencies

Clock multiplier	Transmit data frequency
25	2.5 Gbps
30	3.0 Gbps
34	3.4 Gbps
40	4.0 Gbps
50	5.0 Gbps

75.16 Timestamp distribution network

The timestamp distribution network uses CoreSight timestamp components to generate a 48-bit timestamp value for the trace sources as shown in the following diagram. The CoreSight Timestamp generator generates a 64-bit counter value, but only the least significant 48 bits are distributed to the trace sources. A 7-bit narrow timestamp is derived from the 48-bit timestamp and distributed to the trace client, where a decoder regenerates the 48-bit timestamp. The generator must be programmed. The programming details can be found in Arm CoreSight Components TRM (available in References).

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a1-9251-390d3ab624c1

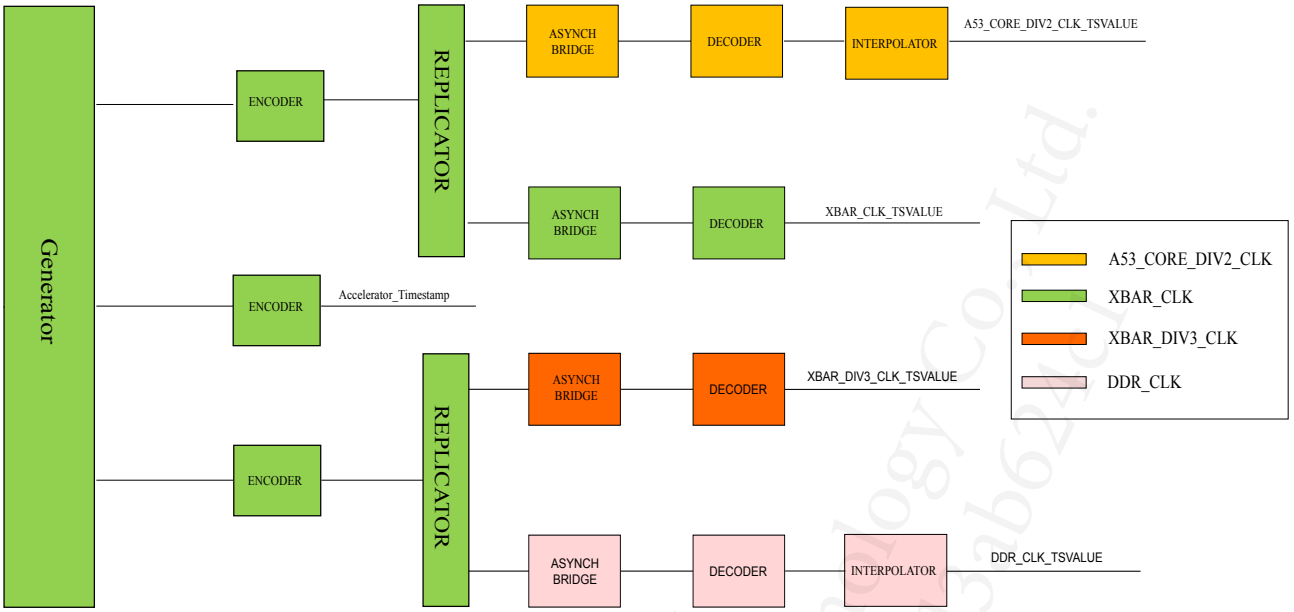


Figure 596. Timestamp distribution network

75.16.1 S32G accelerator timestamp distribution network

The following figure shows the configuration of the accelerator timestamp network that is outside of the system core circuitry.

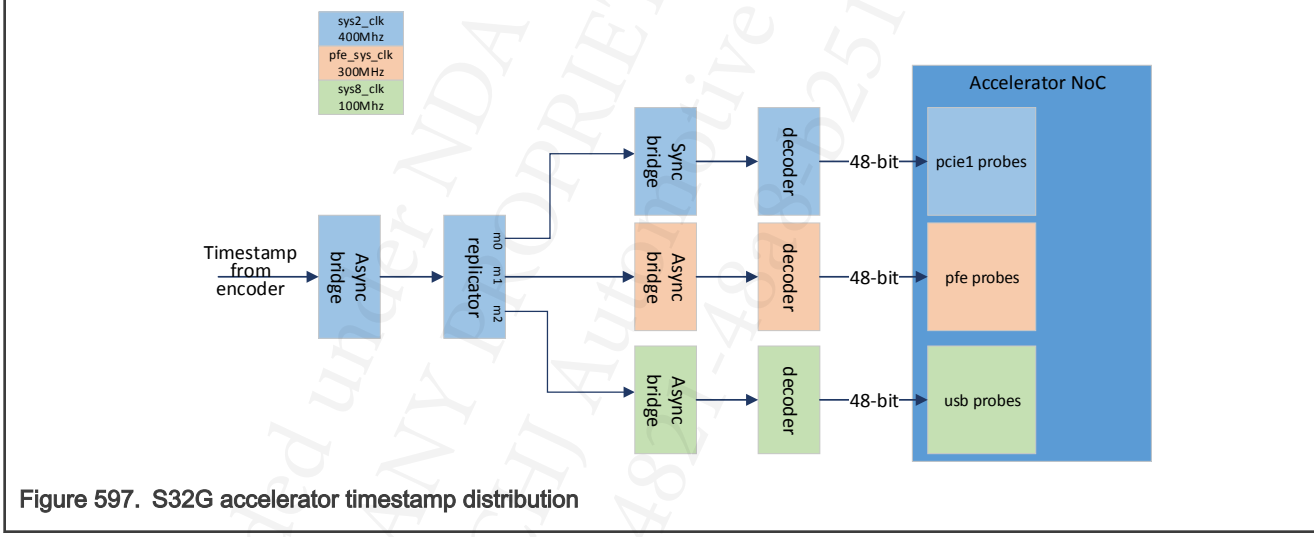


Figure 597. S32G accelerator timestamp distribution

75.17 Low power debug

The device supports debug through Standby mode, to allow the tool to finish operations and the device to transmit all pending trace messages before the device goes into Standby mode. The following sequence achieves low power debug:

1. Set the MDM AP WIR_ENABLE register bit[0] to 1 to notify the device that debug through standby is required.
2. When the device is ready to enter Standby mode, standby is held off until all trace messages have been transmitted from the device.
3. While exiting Standby mode, initialize debug registers, then set the MDM AP WIR_RELEASE register bit[0] to 1 to lift the device out of functional reset.

75.18 References

This chapter refers to the following Arm technical documents:

- Arm CoreSight Architecture Specification

  http://infocenter.arm.com/help/topic/com.arm.doc.ihi0029d/IHI0029D_coresight_architecture_spec_v2_0.pdf
- CoreSight Components Technical Reference Manual

  http://infocenter.arm.com/help/topic/com.arm.doc.ddi0314h/DDI0314H_coresight_components_trm.pdf
- Arm Debug Interface Architecture Specification

  https://static.docs.arm.com/ihi0031/c/IHI0031C_debug_interface_as.pdf
- Cortex A53 Technical Reference Manual

  http://infocenter.arm.com/help/topic/com.arm.doc.ddi0500f/DDI0500F_cortex_a53_r0p4_trm.pdf
- Cortex M7 Technical Reference Manual

  http://infocenter.arm.com/help/topic/com.arm.doc.ddi0489b/DDI0489B_cortex_m7_trm.pdf

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Chapter 76
JTAG Controller (JTAGC)

76.1 Chip-specific JTAGC information

76.1.1 JTAGC device identification register

For the JTAGC device identification values in this chip, see the "Device JTAG ID" section in the Debug Subsystem chapter.

76.2 Introduction

The JTAGC block provides the means to test chip functionality and connectivity while remaining transparent to system logic when not in test mode. Testing is performed via a boundary scan technique, as defined in the IEEE 1149.1-2001 standard. All data input to and output from the JTAGC block is communicated in serial format.

76.2.1 Block diagram

The following is a simplified block diagram of the JTAG Controller (JTAGC) block. See the chip-specific configuration information within this chapter as well as Register description for more information about the JTAGC registers.

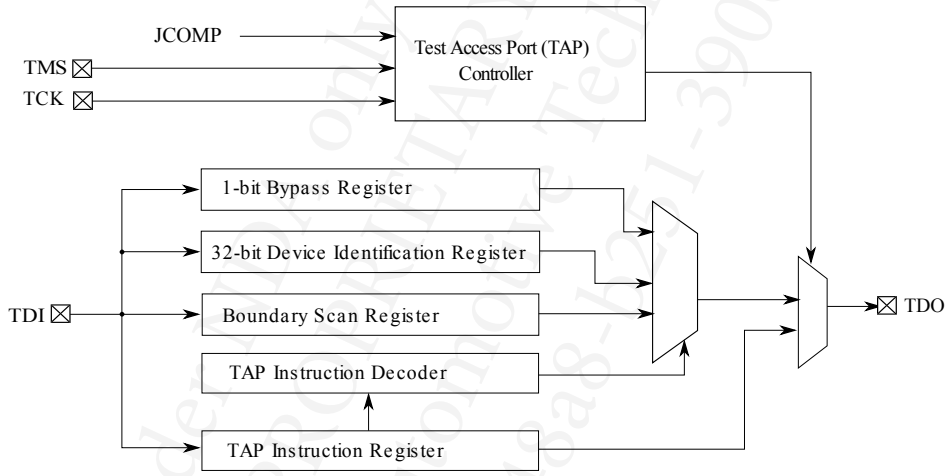


Figure 598. JTAG (IEEE 1149.1) block diagram

76.2.2 Features

The JTAGC block is compliant with the IEEE 1149.1-2001 standard, and supports the following features:

- IEEE 1149.1-2001 TAP interface
  - Four pins (TDI, TMS, TCK, and TDO)
- JCOMP input that provides reset control
- Instruction register that supports several IEEE 1149.1–2001 defined instructions as well as several public and private device-specific instructions (see JTAGC block instructions for a list of supported instructions).
- Sharing of the TAP with other TAP controllers via ACCESS_AUX_x instructions
- Bypass register, boundary scan register, and device identification register
- TAP controller state machine that controls the operation of the data registers, instruction register, and associated circuitry

76.2.3 Modes of operation

The JTAGC block uses JCOMP and a power-on reset indication as its primary reset signals. Several IEEE 1149.1–2001 defined test modes are supported, as well as a bypass mode.

76.2.3.1 Reset

The JTAGC block is placed in reset when:

- Power-on reset is asserted
- JCOMP is negated
- TMS input is held high for enough consecutive rising edges of TCK to sequence the TAP controller state machine into the Test-Logic-Reset state

Holding TMS high for five consecutive rising edges of TCK guarantees entry into the Test-Logic-Reset state regardless of the current TAP controller state. Asserting power-on reset or setting JCOMP to a value other than the value required to enable the JTAGC block results in asynchronous entry into the reset state.

When in reset, the following actions occur:

- The TAP controller is forced into the Test-Logic-Reset state, thereby disabling the test logic and allowing normal operation of the on-chip system logic to continue unhindered.
- The instruction register is loaded with the IDCODE instruction.

76.2.3.2 IEEE 1149.1–2001 defined test modes

The JTAGC block supports several IEEE 1149.1–2001 defined test modes. A test mode is selected by loading the appropriate instruction into the instruction register when the JTAGC is enabled. Supported test instructions include EXTEST, HIGHZ, CLAMP, SAMPLE, and SAMPLE/PRELOAD.

Each instruction defines the set of data register(s) that may operate and interact with the on-chip system logic when the instruction is current. Only one test data register path is enabled to shift data between TDI and TDO for each instruction.

The boundary scan register is enabled for serial access between TDI and TDO when the EXTEST, SAMPLE, or SAMPLE/PRELOAD instructions are active. The single-bit bypass register shift stage is enabled for serial access between TDI and TDO when the following instructions are active:

- BYPASS
- HIGHZ
- CLAMP

The functionality of each test mode is explained in more detail in JTAGC block instructions.

76.2.3.3 Bypass mode

When no test operation is required, the BYPASS instruction can be loaded to place the JTAGC block into bypass mode. When in bypass mode, the single-bit bypass shift register is used to provide a minimum-length serial path to shift data between TDI and TDO.

76.3 External signal description

The JTAGC consists of a set of signals that connect to off-chip development tools and allow access to test support functions. The JTAGC signals are outlined in the following table and described in the following sections.

Table 637. JTAG signal properties

Name	I/O	Function	Reset state
TCK	Input	Test clock	Weak pulldown
TDI	Input	Test data in	Weak pullup
TDO	Output	Test data out	High Z[1]
TMS	Input	Test mode select	Weak pullup
JCOMP	Input	JTAG compliancy	Weak pulldown

1. TDO output buffer enable is negated when the JTAGC is not in the Shift-IR or Shift-DR states. A weak pull may be implemented at the TDO pad for use when JTAGC is inactive.

76.3.1 Test clock input (TCK)

Test Clock Input (TCK) is an input pin used to synchronize the test logic and control register access through the TAP.

76.3.2 Test data input (TDI)

Test Data Input (TDI) is an input pin that receives serial test instructions and data. TDI is sampled on the rising edge of TCK.

76.3.3 Test data output (TDO)

Test Data Output (TDO) is an output pin that transmits serial output for test instructions and data. TDO is tristateable and is actively driven only in the Shift-IR and Shift-DR states of the TAP controller state machine, which is described in TAP controller state machine.

76.3.4 Test mode select (TMS)

Test Mode Select (TMS) is an input pin used to sequence the IEEE 1149.1-2001 test control state machine. TMS is sampled on the rising edge of TCK.

76.3.5 JCOMP JTAG compliancy

The JCOMP signal provides IEEE 1149.1-2001 compatibility and provides the ability to share the TAP. The JTAGC TAP controller is enabled when JCOMP is set to the JTAGC enable encoding—otherwise the JTAGC TAP controller remains in reset.

76.4 Register description

This section provides a detailed description of the JTAGC block registers accessible through the TAP interface, including data registers and the instruction register. Individual bit-level descriptions and reset states of each register are included. These registers are not memory-mapped and can only be accessed through the TAP.

76.4.1 Instruction register

The JTAGC block uses a 8-bit instruction register as shown in the following figure.

The instruction register allows instructions to be loaded into the block to select the test to be performed, or the test data register to be accessed, or both. Instructions are shifted in through TDI when the TAP controller is in the Shift-IR state, and latched on the falling edge of TCK in the Update-IR state. The latched instruction value can only be changed in the Update-IR and Test-Logic-Reset TAP controller states.

Synchronous entry into the Test-Logic-Reset state results in the IDCODE instruction being loaded on the falling edge of TCK. Asynchronous entry into the Test-Logic-Reset state results in asynchronous loading of the IDCODE instruction. During the

Capture-IR TAP controller state, the instruction shift register is loaded with the value 0b1, making this value the register's read value when the TAP controller is sequenced into the Shift-IR state.

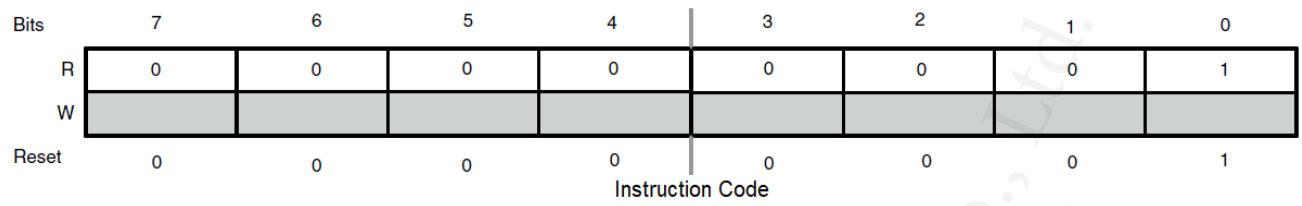


Figure 599. Instruction register

76.4.2 Bypass register

The bypass register is a single-bit shift register path selected for serial data transfer between TDI and TDO when the following instructions are active:

- BYPASS
- HIGHZ
- CLAMP

After entry into the Capture-DR state, the single-bit shift register is set to a logic 0. Therefore, the first bit shifted out after selecting the bypass register is always a logic 0.

76.4.3 Device identification register

The device identification (JTAG ID) register, shown in the following figure, allows the revision number, part number, manufacturer, and design center responsible for the design of the part to be determined through the TAP.

The device identification register is selected for serial data transfer between TDI and TDO when the IDCODE instruction is active. Entry into the Capture-DR state when the device identification register is selected loads the IDCODE into the shift register to be shifted out on TDO in the Shift-DR state. No action occurs in the Update-DR state.

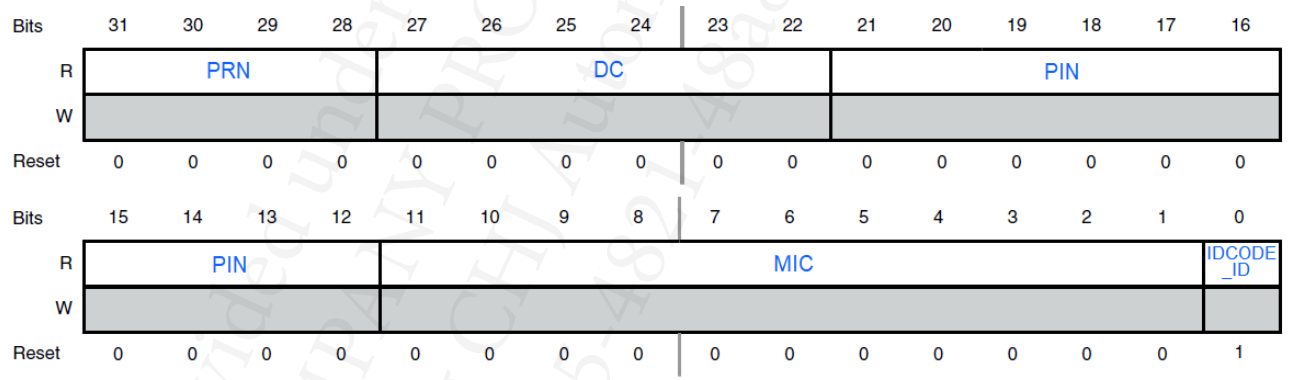


Figure 600. Device identification register

The following table describes the device identification register functions. The device identification register values are described in the chip-specific JTAGC information.

Table 638. Device identification register field descriptions

Field	Function
PRN	Part revision number Contains the revision number of the part.
DC	Design center Indicates the design center.
PIN	Part identification number Contains the part number of the device.
MIC	Manufacturer identity code Contains the reduced Joint Electron Device Engineering Council (JEDEC) ID .
IDCODE ID	IDCODE register ID Identifies this register as the device identification register and not the bypass register. Always set to 1.

76.4.4 Boundary scan register

The boundary scan register is connected between TDI and TDO when the EXTEST, SAMPLE, or SAMPLE/PRELOAD instructions are active. It is used to:

- Capture input pin data
- Force fixed values on output pins
- Select a logic value and direction for bidirectional pins

Each bit of the boundary scan register represents a separate boundary scan register cell, as described in the IEEE 1149.1-2001 standard and discussed in Boundary scan. The size of the boundary scan register and bit ordering is device-dependent and can be found in the device BSDL file.

76.5 Functional description

This section explains the JTAGC functional description.

76.5.1 JTAGC reset configuration

When in reset, the TAP controller is forced into the Test-Logic-Reset state, thus disabling the test logic and allowing normal operation of the on-chip system logic. In addition, the instruction register is loaded with the IDCODE instruction.

76.5.2 IEEE 1149.1-2001 (JTAG) TAP

The JTAGC block uses the IEEE 1149.1-2001 TAP for accessing registers. This port can be shared with other TAP controllers on the MCU. Ownership of the port is determined by the value of the currently loaded instruction. For more detail on TAP sharing via JTAGC instructions, see ACCESS_AUX_x instructions .

Data is shifted between TDI and TDO through the selected register starting with the least significant bit, as illustrated in the following figure. This applies for the instruction register, test data registers, and the bypass register.

Product under NDA only
Bentley PROPRIETARY
Continental Automotive Technology Co., Ltd.
c8c2e5c-4a91-48a8-b251-90e3a1624c1

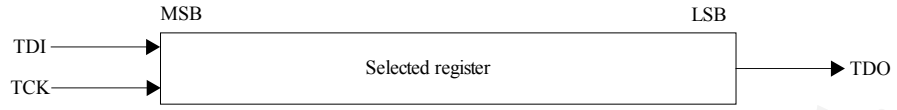
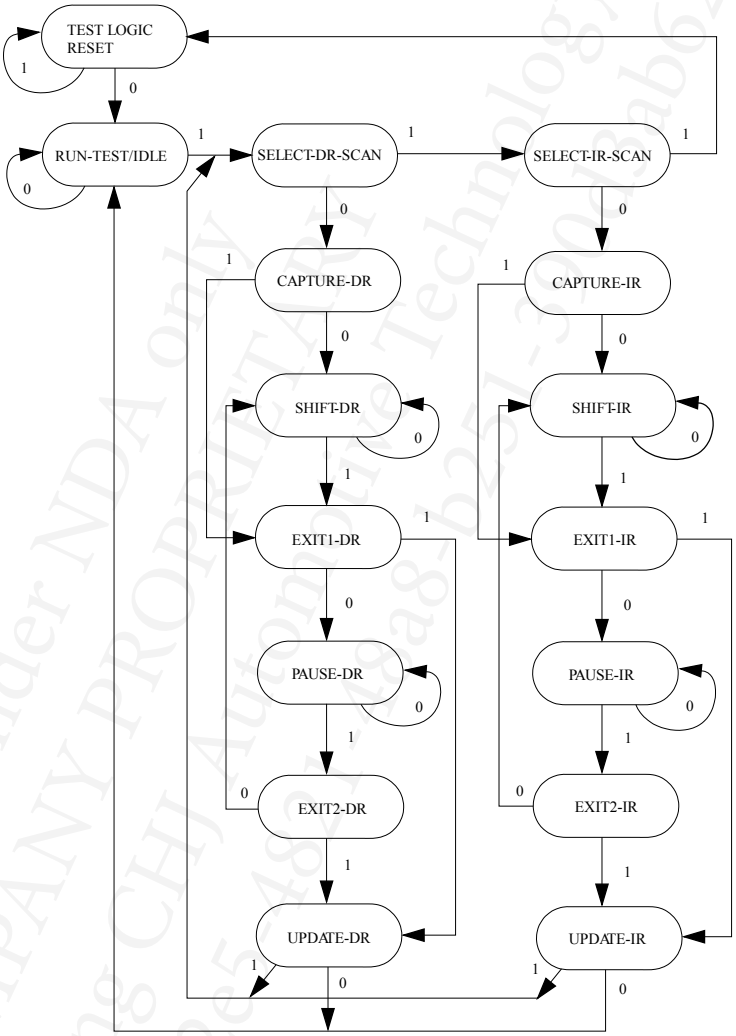


Figure 601. Shifting data through a register

76.5.3 TAP controller state machine

The TAP controller is a synchronous state machine that interprets the sequence of logical values on the TMS pin.

The following figure shows the machine's states. The value shown next to each state is the value of the TMS signal sampled on the rising edge of the TCK signal. As the figure shows, holding TMS at logic 1 when clocking TCK through a sufficient number of rising edges also causes the state machine to enter the Test-Logic-Reset state.



The value shown adjacent to each state transition in this figure represents the value of TMS at the time of a rising edge of TCK.

Figure 602. IEEE 1149.1-2001 TAP controller finite state machine

76.5.3.1 Enabling the TAP controller

The JTAGC TAP controller is enabled by setting JCOMP to a logic 1 value.

76.5.3.2 Selecting an IEEE 1149.1-2001 register

Access to the JTAGC data registers is achieved by loading the instruction register with any of the JTAGC block instructions when the JTAGC is enabled. Instructions are shifted in via the Select-IR-Scan path and loaded in the Update-IR state. At this point, all data register access is performed via the Select-DR-Scan path.

The Select-DR-Scan path is used to read or write the register data by shifting in the data (LSB first) during the Shift-DR state. When reading a register, the register value is loaded into the IEEE 1149.1-2001 shifter during the Capture-DR state. When writing a register, the value is loaded from the IEEE 1149.1-2001 shifter to the register during the Update-DR state. When reading a register, there is no requirement to shift out the entire register contents. Shifting may be terminated after the required number of bits have been acquired.

76.5.4 JTAGC block instructions

The JTAGC implements instructions defined in IEEE 1149.1-2001. See the chip-specific JTAGC information for the JTAGC instructions implemented on this chip. See the IEEE 1149.1-2001 standard for additional information. All undefined opcodes are reserved.

76.5.4.1 IDCODE instruction

IDCODE selects the 32-bit device identification register as the shift path between TDI and TDO. This instruction allows interrogation of the MCU to determine its version number and other part identification data. IDCODE is the instruction placed into the instruction register when the JTAGC block is reset.

76.5.4.2 SAMPLE/PRELOAD instruction

The SAMPLE/PRELOAD instruction has two functions:

- The SAMPLE portion of the instruction obtains a sample of the system data and control signals present at the MCU input pins, and just before the boundary scan register cells at the output pins. This sampling occurs on the rising edge of TCK in the Capture-DR state when the SAMPLE/PRELOAD instruction is active. The sampled data is viewed by shifting it through the boundary scan register to the TDO output during the Shift-DR state. Both the data capture and the shift operation are transparent to system operation.
- The PRELOAD portion of the instruction initializes the boundary scan register cells before selecting the EXTEST or CLAMP instructions to perform boundary scan tests. This is achieved by shifting in initialization data to the boundary scan register during the Shift-DR state. The initialization data is transferred to the parallel outputs of the boundary scan register cells on the falling edge of TCK in the Update-DR state. The data is applied to the external output pins by the EXTEST or CLAMP instruction. System operation is not affected.

76.5.4.3 SAMPLE instruction

The SAMPLE instruction obtains a sample of the system data and control signals present at the MCU input pins, and just before the boundary scan register cells at the output pins. This sampling occurs on the rising edge of TCK in the Capture-DR state when the SAMPLE instruction is active. The sampled data is viewed by shifting it through the boundary scan register to the TDO output during the Shift-DR state. There is no defined action in the Update-DR state. Both the data capture and the shift operation are transparent to system operation.

76.5.4.4 EXTEST external test instruction

EXTEST selects the boundary scan register as the shift path between TDI and TDO. It allows testing of off-chip circuitry and board-level interconnections by driving preloaded data contained in the boundary scan register onto the system output pins. Typically, the preloaded data is loaded into the boundary scan register using the SAMPLE/PRELOAD instruction before the selection of EXTEST. EXTEST asserts the internal system reset for the MCU to force a predictable internal state when performing external boundary scan operations.

76.5.4.5 HIGHZ instruction

HIGHZ selects the bypass register as the shift path between TDI and TDO. When HIGHZ is active all output drivers are placed in an inactive drive state (for example, high impedance). HIGHZ also asserts the internal system reset for the MCU to force a predictable internal state.

76.5.4.6 CLAMP instruction

CLAMP allows the state of signals driven from MCU pins to be determined from the boundary scan register when the bypass register is selected as the serial path between TDI and TDO. CLAMP enhances test efficiency by reducing the overall shift path to a single bit (the bypass register) when conducting an EXTEST type of instruction through the boundary scan register. CLAMP also asserts the internal system reset for the MCU to force a predictable internal state.

76.5.4.7 ACCESS_AUX_x instructions

The JTAGC is configurable to allow other TAP controllers on the device to share the port with it. This is done by providing ACCESS_AUX_x instructions for each of these TAP controllers.

When this instruction is loaded, control of the JTAG pins is transferred to the selected TAP controller. Any data input via TDI and TMS is passed to the selected TAP controller, and any TDO output from the selected TAP controller is sent back to the JTAGC to be output on the pins.

The JTAGC regains control of the JTAG port during the UPDATE-DR state if the PAUSE-DR state was entered. Auxiliary TAP controllers are held in RUN-TEST/IDLE when they are inactive. Instructions not used to access an auxiliary TAP controller on a device are treated like the BYPASS instruction.

76.5.4.8 BYPASS instruction

BYPASS selects the bypass register, creating a single-bit shift register path between TDI and TDO. BYPASS enhances test efficiency by reducing the overall shift path when no test operation of the MCU is required. This allows more rapid movement of test data to and from other components on a board that are required to perform test functions. When the BYPASS instruction is active the system logic operates normally.

76.5.5 Boundary scan

The boundary scan technique allows signals at component boundaries to be controlled and observed through the shift-register stage associated with each pad. Each stage is part of a larger boundary scan register cell, and cells for each pad are interconnected serially to form a shift-register chain around the border of the design. The boundary scan register consists of this shift-register chain, and is connected between TDI and TDO when the EXTEST, SAMPLE, or SAMPLE/PRELOAD instructions are loaded. The shift-register chain contains a serial input and serial output, as well as clock and control signals.

76.6 Initialization/application information

The test logic is a static logic design, and TCK can be stopped in either a high or low state without loss of data. However, the system clock is not synchronized to TCK internally. Any mixed operation using both the test logic and the system functional logic requires external synchronization.

To initialize the JTAGC block and enable access to registers, the following sequence is required:

1. Set the JCOMP signal to the JTAGC enable value, thereby enabling the JTAGC TAP controller.
2. Load the appropriate instruction for the test or action to be performed.

Chapter 77
JTAG Data Communication (JDC)

77.1 Introduction

The JTAG Data Communication (JDC) module provides the capability to move register data between the IPS and JTAG domains. This facilitates communication between internal resources that access memory-mapped register space and an external tool that accesses the JTAG port.

77.2 Overview

The JDC module consists of:

- IPS-accessible registers
- JTAG-accessible registers
- Associated logic to coordinate movement of data from one register domain to another

The JDC implements the following IPS data registers, occupying separate memory space:

- 1 32-bit memory mapped register that can be read or written via IPS (JTAG Output Data Register (JOUT_IPS)), and whose contents are ported out for capture into a JTAG register (JOUT) to be read via the JTAG port.
- 1 32-bit memory mapped register that can only be read via IPS (JTAG Input Data Register (JIN_IPS)), and whose contents are loaded from a JTAG register (JIN).

JDC indicates when:

- New data has been shifted in via the JTAG port and is ready to be read from the JTAG Input Data Register (JIN_IPS) register
- New data has been written to the JTAG Output Data Register (JOUT_IPS) register and is ready to be read via the JTAG port

A Module Status Register (MSR) captures the state of these flags and also provides the flags to a JTAG register (JOUT) for tool visibility.

There is a single bus interface to port register data out to the JTAGC, and a single bus interface to port data in from the JTAGC. The architecture is shown in the following figure.

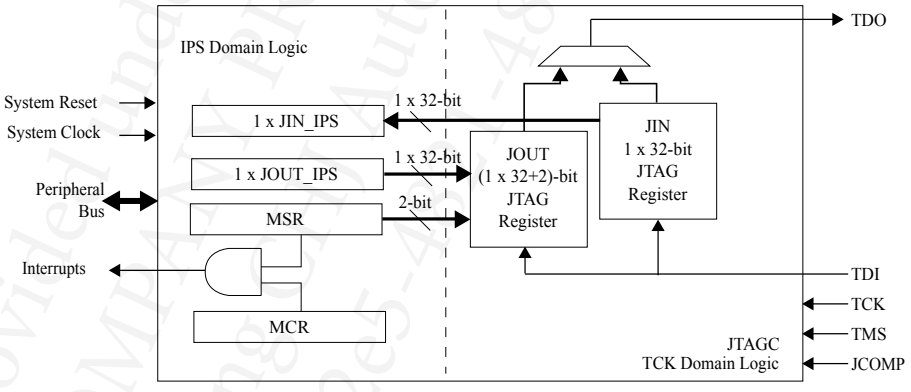


Figure 603. JDC block diagram

77.3 Signal description

JDC signals are listed in the following table.

Table 639. Signal properties

Name	Function	I/O	Reset
TCK	Test clock input	I	—
TMS	Test mode select	I	0
TDI	Test data in	I	0
JCOMP	JTAG compliance pin	I	0
TDO	Test data out	O	HiZ

77.4 JDC register descriptions

The following sections describe the implemented 32-bit registers. Only 32-bit accesses are valid. The effects of access that are not 32 bits are not defined.

77.4.1 JDC memory map

JDC base address: 402F_4000h

Offset	Register	Width (In bits)	Access	Reset value
0h	Module Configuration Register (MCR)	32	RW	0000_0000h
4h	Module Status Register (MSR)	32	W1C	0000_0000h
8h	JTAG Output Data Register (JOUT_IPS)	32	RW	0000_0000h
Ch	JTAG Input Data Register (JIN_IPS)	32	RO	0000_0000h

77.4.2 Module Configuration Register (MCR)

Offset

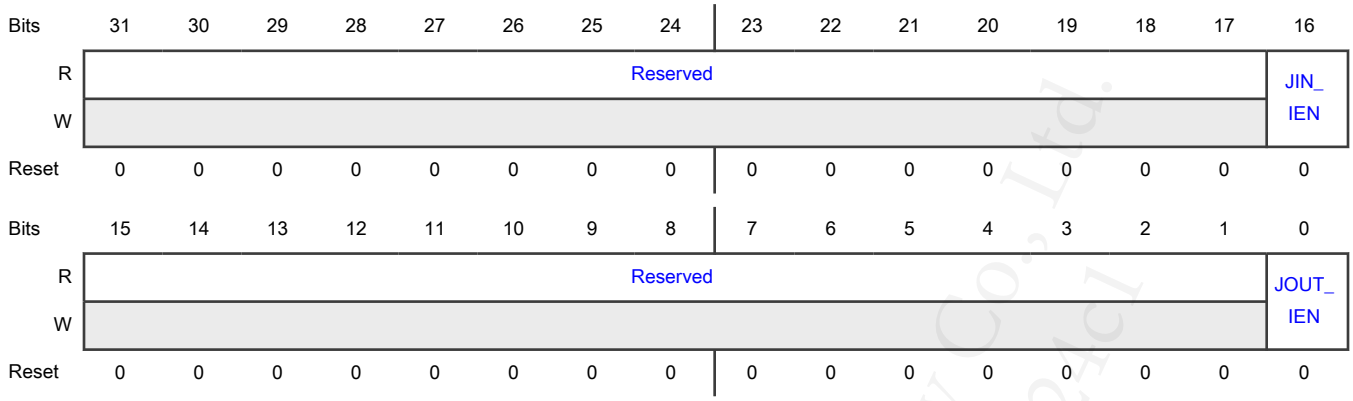
Register	Offset
MCR	0h

Function

The MCR enables the interrupt outputs of the JDC. This register is reset by system functional reset.

Provided by IHS Markit under license with NXP Semiconductors
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46265-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-17 —	Reserved
16 JIN_IEN	JIN Interrupt Enable. 0b - Setting MSR[JIN_INT] bit does not assert the JIN interrupt 1b - Setting MSR[JIN_INT] bit asserts the JIN interrupt
15-1 —	Reserved
0 JOUT_IEN	JOUT Interrupt Enable. 0b - Setting MSR[JOUT_INT] bit does not assert the JOUT interrupt 1b - Setting MSR[JOUT_INT] bit asserts the JOUT interrupt

77.4.3 Module Status Register (MSR)

Offset

Register	Offset
MSR	4h

Function

The MSR holds the JTAG register status and interrupt bits. This register is reset by system functional reset.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48e8-b251-390d8ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved													JIN_RDY	Reserved	JIN_INT
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved													JOUT_RDY	Reserved	JOUT_INT
W																W1C
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-19 —	Reserved
18 JIN_RDY	JIN Ready (read only). 0b - Cleared upon software read of JIN_IPS contents via IPS 1b - Set when new data is written to the JIN_IPS register
17 —	Reserved
16 JIN_INT	JIN Interrupt. 0b - Cleared by writing logic 1 1b - Set when new data is written to the JIN_IPS register
15-3 —	Reserved
2 JOUT_RDY	JOUT Ready (read only). 0b - Cleared upon tool read of JOUT register via JTAG port 1b - Set when new data is written to the JOUT_IPS register
1 —	Reserved
0 JOUT_INT	JOUT Interrupt. 0b - Cleared by writing logic 1 1b - Set when JOUT_RDY bit is cleared by tool reading JOUT register

Provided under NDA only
COMPANY PROPRIETARY
Beijing CJJ Automotive Technology Co., Ltd.
c0462e54-821-4a8-b21-390d3ab624

77.4.4 JTAG Output Data Register (JOUT_IPS)

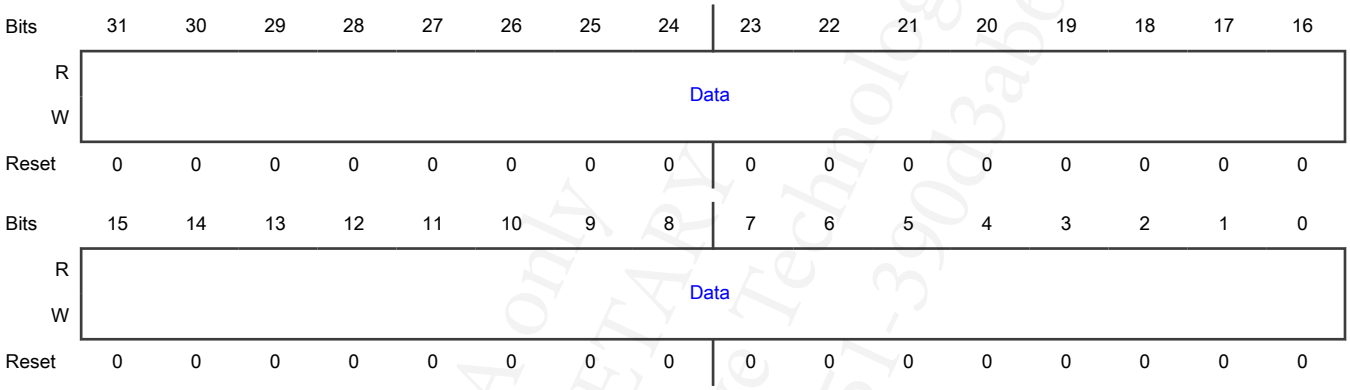
Offset

Register	Offset
JOUT_IPS	8h

Function

The JOUT_IPS register holds data written via IPS. The JDC captures the JOUT_IPS contents into the JOUT register to be read via the JTAG port. This register is reset by system functional reset.

Diagram



Fields

Field	Function
31-0 Data	JOUT_IPS data.

77.4.5 JTAG Input Data Register (JIN_IPS)

Offset

Register	Offset
JIN_IPS	Ch

Function

The JIN_IPS register holds data written to the JTAG input data register (JIN) via the JTAG port, where the data can be read via IPS. Any IPS write to the JIN_IPS register returns a transfer error. This register is reset by system functional reset.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c62e5-4821-48a8-b251-39003ab62401

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Data															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Data															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 Data	JIN_IPS data.

77.5 Non-memory-mapped register definition

The JDC also implements two JTAG accessible registers that are not memory-mapped. One JTAG register shifts in data to be placed in the JIN_IPS register. The other JTAG register captures data from the JOUT_IPS register plus ready status from the MSR, to be shifted out via the JTAG port.

77.5.1 JTAG output data register (JOUT)

The JOUT register captures data from the JOUT_IPS register upon execution of the JOUT_READ JTAG instruction. It also holds the JIN_RDY and JOUT_RDY status bits. This register is reset by system power-on reset and JTAG reset. The reset value of the JOUT register is 0. The following figure shows the format of the JOUT register.

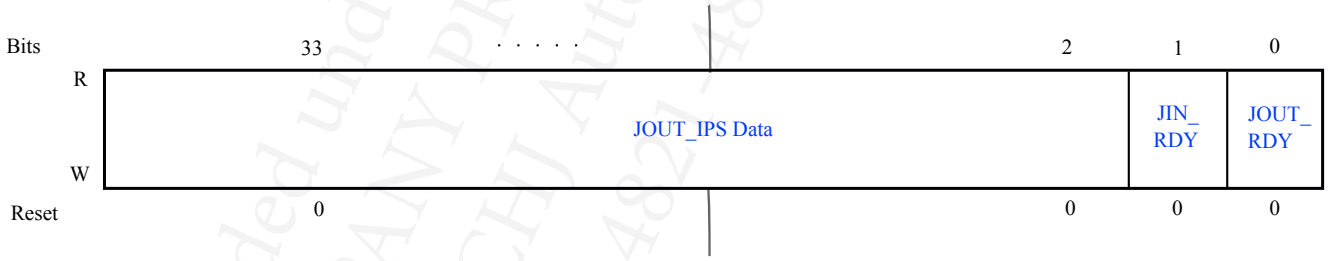


Figure 604. JOUT

The JOUT register is described in the following table.

Table 640. JOUT JTAG register field descriptions

Field	Function
JOUT_IPS Data	JOUT_IPS Data Data value from JOUT_IPS register

Table continues on the next page...

Table 640. JOUT JTAG register field descriptions (continued)

Field	Function
JIN_RDY	JIN_RDY State of JIN_RDY bit from MSR
JOUT_RDY	JOUT_RDY State of JOUT_RDY bit from MSR

77.5.2 JTAG input data register (JIN)

The external tool writes data to the JIN register via JTAG during execution of the JIN_WRITE JTAG instruction. The JDC later captures JIN data in the JIN_IPS register to be read via IPS. This register is reset by system power-on reset and JTAG reset. The reset value of the JIN register is 0. The following figure shows the format of the JIN register.

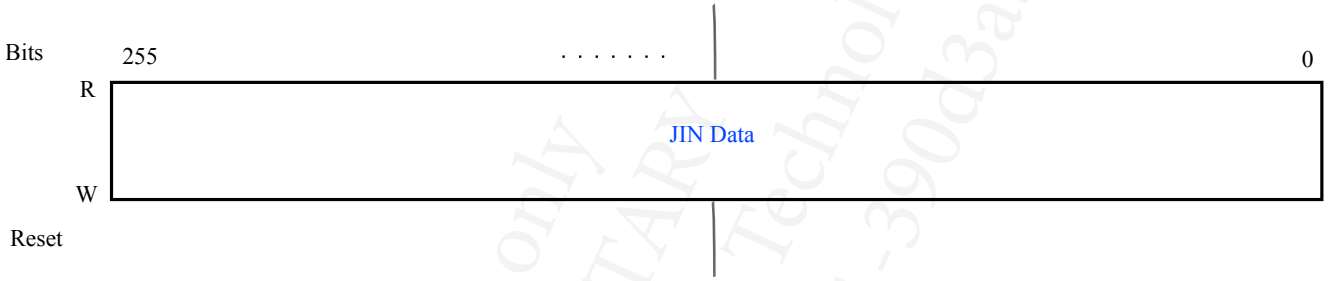


Figure 605. JIN

The JIN register is described in the following table.

Table 641. JIN JTAG register field descriptions

Field	Description
JIN Data	Contains data to be captured in JIN_IPS register upon exit of Update-DR state when executing WRITE_JIN JTAG instruction.

77.6 Functional description

The JDC module provides the ability to shift in data via the JTAG port and capture that data in memory-mapped register that can be accessed via IPS. It also provides the ability to capture data written to memory-mapped register into a JTAG shift register for output via the JTAG port. An overview of the module functionality is described here.

- Software write to the JOUT_IPS register:
  1. The JDC sets the JOUT_RDY flag bit, indicating new data is available to be read from the JOUT register via the JTAG port.
  2. The MSR reflects the state of the JOUT_RDY bit.
  3. The JDC also captures the state of the JOUT_RDY bit in the JOUT register.
  4. A JTAG read of the JOUT register via execution of the JOUT_READ instruction with a JOUT_RDY bit whose value is logic 1 indicates the register contains new data.
  5. The JDC clears the JOUT_RDY flag bit upon exit of the Capture-DR JTAG state during execution of the JOUT_READ instruction.

6. Clearing the JOUT_RDY bit indicates to software that a new data value can be written to the JOUT_IPS register.

- JTAG write to the JIN register via execution of the JIN_WRITE JTAG instruction:
  1. The JDC updates the contents of the JIN_IPS register upon exit of the Update-DR state.
  2. An update of the JIN_IPS register sets the JIN_RDY flag bit, indicating new data is available to be read via IPS.
  3. The MSR register reflects the state of the JIN_RDY bit.
  4. The JDC also captures the state of the JIN_RDY bit in the JOUT register.
  5. The JDC clears the JIN_RDY flag bit upon software read of the JIN_IPS register.
  6. A JTAG read of the JOUT register with a JIN_RDY value of logic 0 indicates new data can be written to the JIN register.

77.6.1 JTAG register access

See the JTAGC documentation for information on how to access the JTAG registers.

77.6.1.1 JDC block instructions

The JDC block implements the instructions listed in Table 642. This section gives an overview of each instruction. All undefined opcodes are reserved.

Table 642. JTAG instructions

Instruction	Code[4:0]	Instruction summary
Reserved	00001	Factory debug reserved.
JOUT_READ	00010	Selects JOUT data register. The JDC captures data from JOUT_IPS into JOUT data register upon entry to Capture-DR state when JOUT_READ is active.
JIN_WRITE	01110	Selects JIN data register. The JDC captures data from JIN into JIN_IPS upon exit of Update-DR state when JIN_WRITE is active.
BYPASS[1]	11111	Selects bypass register for data operations.
Reserved	All other opcodes	Decoded to select bypass register.

1. This is an IEEE 1149.1-2001 defined instruction. See the IEEE 1149.1-2001 standard for more details.

77.7 Secure challenge/response connection procedure

A summary of how the JDC can be used to interact with a security module to perform challenge/response password authorization is as follows:

1. After exit of reset, all registers are at their reset values. JIN_RDY bit value of 0 indicates the tool may write data to the JIN register. JOUT_RDY bit value of 0 indicates that software may write data to the JOUT_IPS register.
2. The MCR register is programmed to enable JOUT and JIN interrupts, or not.
3. Once the tool is ready to start the authorization process, it writes a value to the JIN register via execution of the JTAGC JIN_WRITE instruction.
4. Software monitors the state of the JIN_RDY bit or enables the JIN interrupt and uses the interrupt assertion as a trigger to start the authorization process.
5. Once the authorization process is started, software writes the first JOUT value to the JOUT_IPS register. This sets the JOUT_RDY bit value to 1.

6. Following the initial tool write to the JIN register, the tool polls the JOUT register to determine when software has provided a challenge word. Upon reading the JOUT register with JOUT_RDY bit set, the tool records the data value to be used to generate the appropriate response. The read of the JOUT register clears the JOUT_RDY bit. The clearing of the JOUT_RDY bit also sets the JOUT_INT bit and asserts the JOUT interrupt if the MCR[JOUT_IEN] is set.
7. With the JOUT_RDY bit cleared, software may now provide the next challenge word.
8. Once the tool has generated a response word, it uses the value of the JIN_RDY bit from the JOUT register read to determine when it can perform a write to the JIN register to provide the response.
9. The security module can provide additional challenge words and read back the corresponding response words as needed, repeating the process.

Simply providing a password without using the challenge/response protocol can be done in a similar way. The tool provides 32-bits of the password at a time with each write to the JIN register, and monitors the JIN_RDY bit value of the JOUT register to determine when the next JIN register value can be written. There is no need for software to write the JOUT_IPS register in this case.

For details related to the lifecycle stage at which this process is performed, see the "Hardware Security Engine (HSE_B)" chapter.

77.8 Glossary

IPS	Internal peripheral system
XBAR	Crossbar bus interface

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab02a11

Chapter 78
Aurora Trace Port (ATP) Subsystem

78.1 Introduction

The Aurora protocol is used to send debug information over a high-speed serial link. A high-speed serial link using the Aurora protocol is ideally suited to provide high speed trace debug support because it is a light-weight protocol with 80% efficiency based on an open standard developed by Xilinx Corporation. The Aurora protocol increases internal visibility and debug capability by providing higher trace throughput up to 20 Gbps at 5 GHz with four data lanes. With the Aurora interface, each data lane only takes two pins per differential output pair (positive and negative).

The Aurora Trace Port (ATP) subsystem is a part of the debug subsystem in Arm-based SOCs. ATP uses Xilinx Aurora 8B/10B protocol to send data off-chip. The control of the ATP subsystem is through the Debug Access Port (DAP) and Advanced Peripheral Bus (APB) interface. The data path of the ATP communicates with Arm cores through the Trace Port Interface Unit (TPIU) module. ATP has dedicated 4 KB address space and the address bus can be controlled and monitored. Access to ATP registers is through the Serial Wire JTAG Debug Port (SWJ-DP). Interaction of ATP with DAP and TPIU is shown in the following figure.

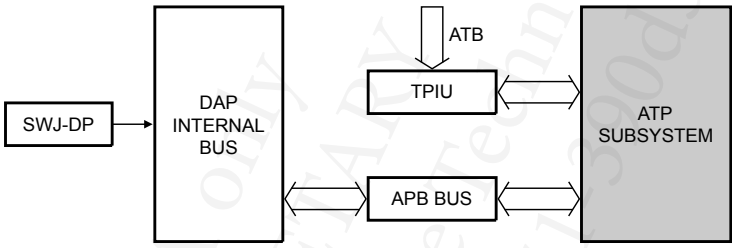


Figure 606. ATP subsystem overall debug interface

ATP subsystem has control and data paths. The control path consists of register configuration of all the internal modules, most of these registers are write and readable. The DAP sends the APB interface signals along with the select line to ATP. Users configure registers using dedicated 4 KB address space. The data path of ATP takes 32-bit data from TPIU module and ATP provides the gated clock of Aurora PHY output Clock (Aurora Clock, PLL output Clock / 10) to control the data flow of TPIU.

78.1.1 Features

ATP supports these features:

- Xilinx Aurora Protocol 8B/10B Specification V2.3 compliance
- Simplex mode with static (timer-based) channel training
- 0, 2 or 4 Aurora lane configuration
- Up to 5 Gbps bandwidth for each lane
- Arm compliance registers
- Configurable clock compensation
- Aurora PLL frequencies 2.5, 3, 3.4, 4, and 5 Gbps can be configured by using register configuration (PLLDIV).
- The Aurora PLL generates 5GHz clock using LVDS input reference clock of 100MHz.

The PLL does not detect loss of reference clock. If the clock stops after the PLL achieves lock, PLLDIG continues to indicate the lock status.

78.2 External signal description

The table below describes the port connections of ATP subsystem.

Direction	Width	Port name	Port details	Domain
output	1	AUR_TX0_PAD_N	Aurora TX data output - LVDS	-
output	1	AUR_TX0_PAD_P	Aurora TX data output - LVDS	-
output	1	AUR_TX1_PAD_N	Aurora TX data output - LVDS	-
output	1	AUR_TX1_PAD_P	Aurora TX data output - LVDS	-
output	1	AUR_TX2_PAD_N	Aurora TX data output - LVDS	-
output	1	AUR_TX2_PAD_P	Aurora TX data output - LVDS	-
output	1	AUR_TX3_PAD_N	Aurora TX data output - LVDS	-
output	1	AUR_TX3_PAD_P	Aurora TX data output - LVDS	-
input	1	AUR_RX_CLK_P	LVDS input clock - P	-
input	1	AUR_RX_CLK_N	LVDS input clock - N	-
inout	1	AUR_PLL_AVDD	PLL analog supply	1.8 V
inout	1	AUR_PLL_AVSS	PLL analog ground	0 V
inout	1	AUR_PLL_DVDD	PLL digital supply	0.8 V
inout	1	AUR_PLL_DVSS	PLL digital ground	0 V
inout	1	SOC_VDD_CORE	SOC core supply	0.8 V
inout	1	SOC_VSS_CORE	SOC core ground	0 V
inout	1	AUR_VDDIO	1.8V power supply	NA

78.3 Overview

The ATP subsystem supports high-speed trace output and transmits high bandwidth data off-chip from the SoC Arm debug infrastructure using an add-on to the TPIU parallel interface. This ensures complete reuse of existing Arm IP.

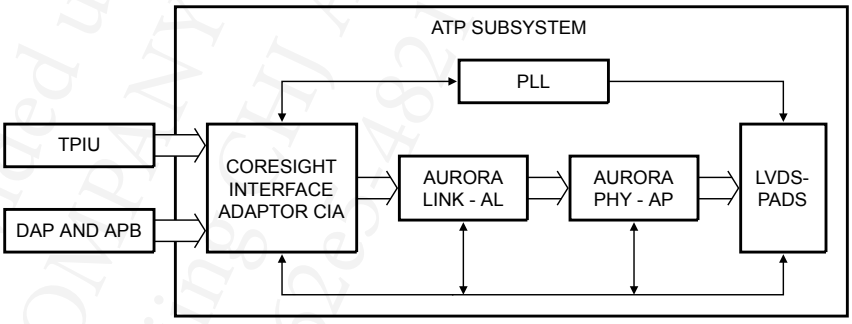


Figure 607. ATP subsystem modules and interface

The ATP subsystem consist of five modules:

Table 643. ATP subsystem modules

Module	Description
CoreSight Interface Adaptor (CIA)	CIA interacts with Trace Port Interface Unit (TPIU) and APB interface bus. CIA takes trace data from TPIU, converts it to the required format for Aurora Link (AL), and generates all the required intermediate signals. CIA interface with APB is connected for register configuration of ATP and it controls and reads the status of various registers of the ATP subsystem. CIA also controls the clock for TPIU. Therefore, there are no additional signals to control the data flow of TPIU and clock gating is used to throttle the data.
Aurora Link (AL)	AL communicates with the CIA and Aurora PHY (AP). AL receives 64-bit format data from CIA, implements Aurora 8B/10B protocol, and appends CRC based on user selection. Based on the number of lanes configured at AL, the data is sent to AP on the corresponding lanes.
Aurora PHY (AP)	AP interacts with AL and LVDS pads. AP receives the Aurora 8B/10B format segregated data from the AL and performs all the operations required to transfer the data off-chip. APs major function is to convert parallel data to serial data at 1:10 clock rate and to transfer data on to the LVDS pads.
Aurora PLL	Aurora PLL module generates the clock for the Aurora sub-system (AURORA_CLK) up to 5 GHz by taking the input reference clock (AURORA_EXT_CLK) of 100 MHz. The Aurora high-speed clock is used to send the data off-chip. The Aurora PLL is configurable using register settings.
Aurora LVDS pads	Aurora pads are high-speed supporting pads. They are LVDS differential pair pads and each supports up to 5 GHz per lane. These pads receive the data from AP and controlled by CIA.

78.4 ATP register descriptions

ATP subsystem registers configure the AL, AP, Aurora PLL, Aurora LVDS IO pads, and CIA.

All registers and fields which are marked reserved when written with values other than reset values could lead to unpredictable hardware behavior. Read values of reserved/bits registers can be ignored.

78.4.1 ATP memory map

ATP base address: 5103_3000h

Offset	Register	Width (In bits)	Access	Reset value
0h	PLL Control (PLLC)	32	RW	8000_0000h
4h	PLL Status (PLLS)	32	W1C	0000_0200h
8h	PLL Divider (PLLDIV)	32	RW	0C01_1019h
18h	PLL Calibration 2 (PLLCAL_2)	32	RW	0000_2000h
20h	PLL Clock MUX (PLLCKMUX)	32	RW	0000_0000h
300h	AL Status (ALS)	32	RO	0008_0000h
308h	AL General Control (ALGC)	32	RW	0000_0000h
30Ch	AL Training Control (ALTC)	32	RW	0000_0000h
440h	LVDS TX IO Configuration (LVDSTX)	32	RW	0000_C001h
444h	LVDS RX IO Configuration (LVDSRX)	32	RW	0000_0000h

Table continues on the next page...

Table continued from the previous page...

Offset	Register	Width (In bits)	Access	Reset value
448h	LVDS TX OBE Configuration (LVDSTXOBE)	32	RW	0000_0000h
480h	CIA Control (CIAC)	32	RW	0400_0000h
488h	Aurora Trace Port Enable (ATPE)	32	RW	0000_0000h
F00h	Integration Mode Control (ITCTRL)	32	RO	0000_0000h
FA0h	Claim Tag Set (CLAIMSET)	32	RO	0000_0000h
FA4h	Claim Tag Clear (CLAIMCLR)	32	RO	0000_0000h
FA8h	Device Affinity 0 (DEVAFF0)	32	RO	0000_0024h
FACh	Device Affinity 1 (DEVAFF1)	32	RO	0000_0024h
FB0h	Lock Access Register (LAR)	32	WO	0000_0000h
FB4h	Lock Status Register (LSR)	32	RO	0000_0003h
FB8h	Authentication Status (AUTHSTATUS)	32	RO	0000_0000h
FBCh	Device Architecture (DEVARCH)	32	RO	01D0_0024h
FC0h	Device Configuration 2 (DEVID2)	32	RO	0000_0000h
FC4h	Device Configuration 1 (DEVID1)	32	RO	0000_0000h
FC8h	Device Configuration (DEVID)	32	RO	0000_0000h
FCCh	Device Type Identifier (DEVTYPE)	32	RO	0000_0011h
FD0h	Peripheral Identification Register 4 (PIDR4)	32	RO	0000_0000h
FE0h	Peripheral Identification Register 0 (PIDR0)	32	RO	0000_0014h
FE4h	Peripheral Identification Register 1 (PIDR1)	32	RO	0000_00EAh
FE8h	Peripheral Identification Register 2 (PIDR2)	32	RO	0000_0008h
FECh	Peripheral Identification Register 3 (PIDR3)	32	RO	0000_0000h
FF0h	Component Identification Register 0 (CIDR0)	32	RO	0000_000Dh
FF4h	Component Identification Register 1 (CIDR1)	32	RO	0000_0009h
FF8h	Component Identification Register 2 (CIDR2)	32	RO	0000_0005h
FFCh	Component Identification Register 3 (CIDR3)	32	RO	0000_00B1h

78.4.2 PLL Control (PLLC)

Offset

Register	Offset
PLLC	0h

Preliminary Information
COMPANY CONFIDENTIAL
Beijing Chipsea Automotive Technology Co., Ltd.
c8c462e3-2521-48a8-b251-39f0d6ab6a1d

Function

This register controls the PLL Reset selection.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PLLPD	Reserved							Reserved	Reserved						
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31 PLLPD	PLL Reset This field controls the Analog PLL reset input. (PLLPD also used as enable/disable for PLL) 0b - Analog PLL reset is deasserted. 1b - Analog PLL reset is asserted.
30-24 —	Reserved
23 —	Reserved
22-0 —	Reserved

78.4.3 PLL Status (PLLS)

Offset

Register	Offset
PLLS	4h

Function

This register indicates the PLL status.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						Reserved	Reserved	Reserved				LOL	LOCK	Reserved	
W													W1C			
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-10 —	Reserved
9 —	Reserved
8 —	Reserved
7-4 —	Reserved
3 LOL	PLL Loss of Lock Status This field indicates if loss-of-lock has been detected. 0b - No loss-of-lock detected. 1b - Loss-of-lock detected.
2 LOCK	PLL Lock Status This field indicates if PLL has acquired lock. 0b - PLL unlocked. 1b - PLL locked.
1-0 —	Reserved

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab62...

78.4.4 PLL Divider (PLLDIV)

Offset

Register	Offset
PLLDIV	8h

Function

This register configures the PLL divider.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	Reserved						Reserved			ODIV1					
W																
Reset	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	1

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	RDIV			Reserved				MFID							
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	1

Fields

Field	Function
31 —	Reserved
30-25 —	Reserved
24-22 —	Reserved
21-16 ODIV1	PLL Output Division Factor 1 This field is the output frequency divider for raw AURORA_CLK PLL clock. 000000b - Divide by 1 000001b - Divide by 1 000010b - Divide by 2 000011b - Divide by 3

Table continues on the next page...

Table continued from the previous page...

Field	Function
	000100b - Divide by 4 111111b - Divide by 63
15 —	Reserved
14-12 RDIV	PLL Input Clock Predivider This field controls the value of the divider on the input clock. The output of the predivider circuit generates the reference clock to the PLL analog loop. 000b - Divide by 1 001b - Divide by 1 010b - Divide by 2 011b - Divide by 3 100b - Divide by 4 101b - Divide by 5 110b - Divide by 6 111b - Divide by 7
11-8 —	Reserved
7-0 MFID	PLL Multiplication Factor Integer Divider This field controls the value of the divider in the feedback loop. The value specified by MFID establishes the multiplication factor applied to the reference frequency. Divider value = MFID, where MFID has the range 10...150 (Ah...96h). All other values are reserved. AURORA_CLK = (MFID × 2 × RefClock) / (RDIV × ODIV1) where the RefClock is determined by PLLCKMUX[REFCLKSEL] setting.

78.4.5 PLL Calibration 2 (PLLCAL_2)

Offset

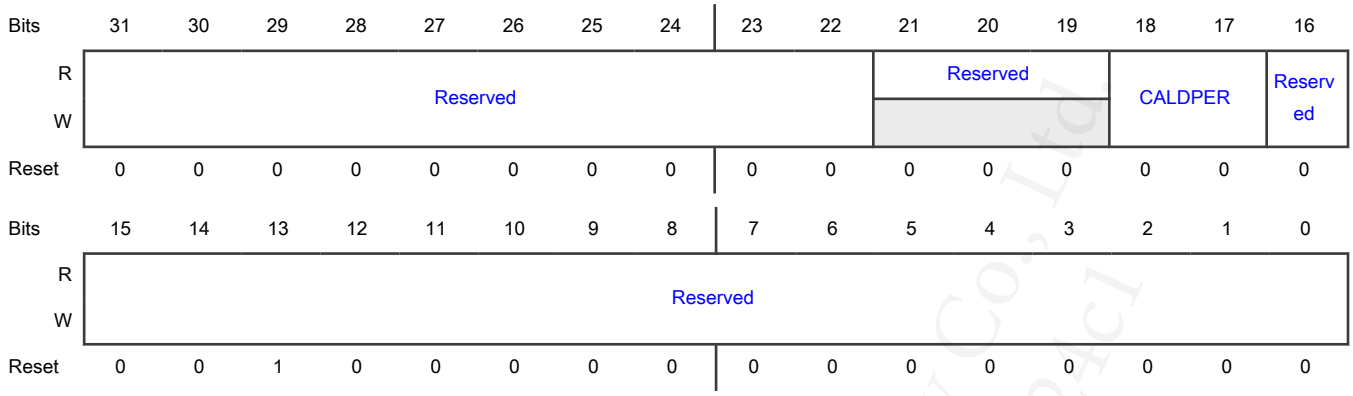
Register	Offset
PLLCAL_2	18h

Function

This offset register consists of PLL Calibration-2 Registers.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5-4821-48e-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-22 —	Reserved
21-19 —	Reserved
18-17 CALDPER	Calibration DAC Period This field is used to control the accuracy of the calibration logic to measure the VCO clock periods in given reference clock periods. 00b - Not allowed 01b - Not allowed 10b - 64 DAC settling time (in terms of number of PLL reference clock cycles) 11b - Not allowed
16-0 —	Reserved

78.4.6 PLL Clock MUX (PLLCKMUX)

Offset

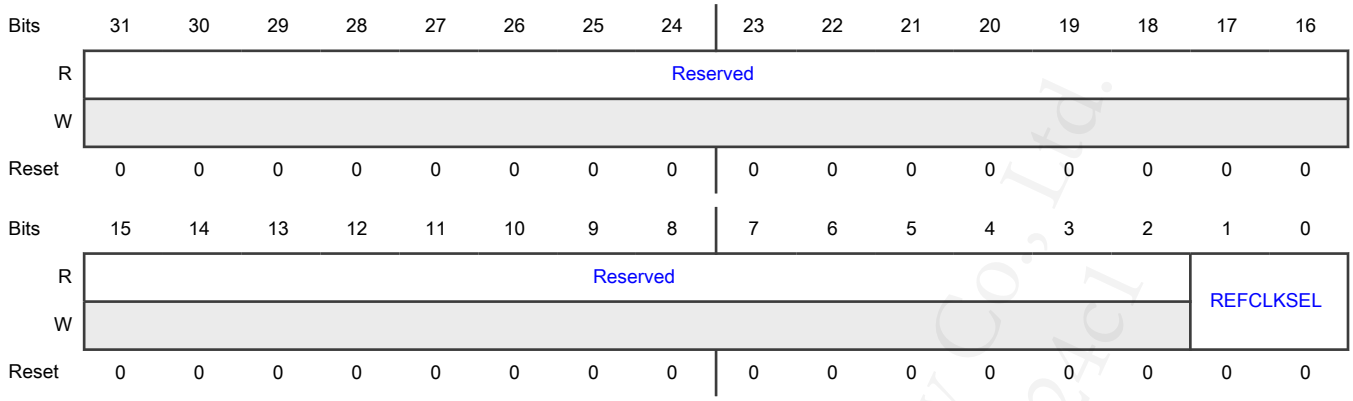
Register	Offset
PLLCKMUX	20h

Function

This register configures the PLL MUX to select Internal / External Clock source.

provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca462e5-4821-4828-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-2 —	Reserved
1-0 REFCLKSEL	Reference Clock Selection This field is used to select input reference clock for PLL. REFCLKSEL should never be written with 10b or 11b. 00b - 100 MHz LVDS reference clock (AURORA_EXT_CLK) 01b - 40 MHz crystal oscillator clock (FXOSC_CLK) 10b - Reserved 11b - Reserved

78.4.7 AL Status (ALS)

Offset

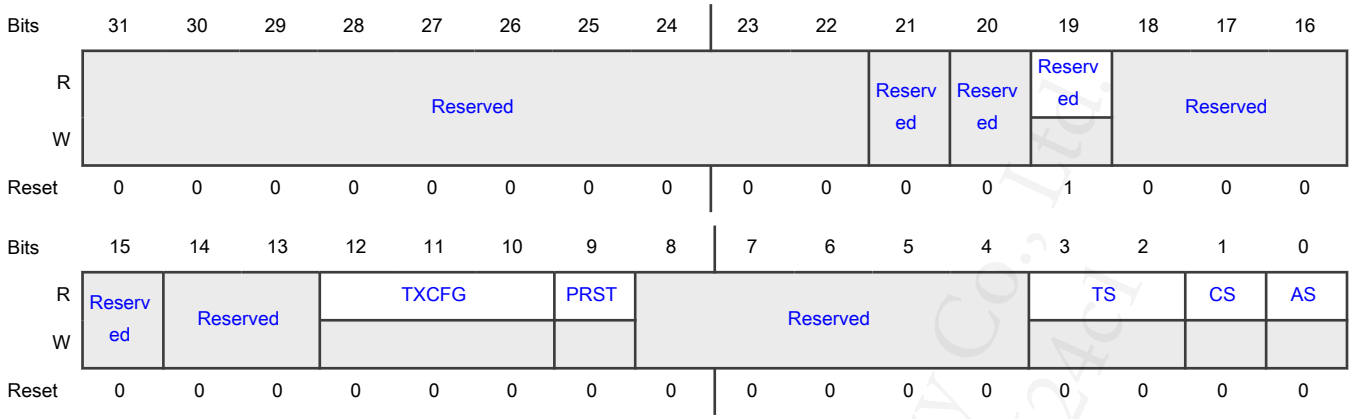
Register	Offset
ALS	300h

Function

This register indicates general status of the AL.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-22 —	Reserved
21 —	Reserved
20 —	Reserved
19 —	Reserved
18-15 —	Reserved
14-13 —	Reserved
12-10 TXCFG	TX Lane Configuration This field indicates the number of enabled TX lanes. 000b - Reserved. 001b - 2 TX lanes. 010b - Reserved. 011b - 4 TX lanes.
9 PRST	Channel Partner Reset This field indicates that the AL has detected a reset at the channel partner.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0b - No reset of channel partner. 1b - Channel partner has reset (causes AL to reset as well).
8-4 —	Reserved
3-2 TS	Training Status This field indicates that each stage of training has successfully completed. 00b - Training has not completed any stage. 01b - AL has successfully completed symbol lock and alignment. 10b - AL has successfully completed channel bonding. 11b - AL has successfully completed verification.
1 CS	Channel Status This field indicates the channel status. 0b - Aurora channel not ready. 1b - Aurora channel ready.
0 AS	Aurora Status This field indicates the Aurora status. 0b - Aurora not enabled. 1b - Aurora enabled.

78.4.8 AL General Control (ALGC)

Offset

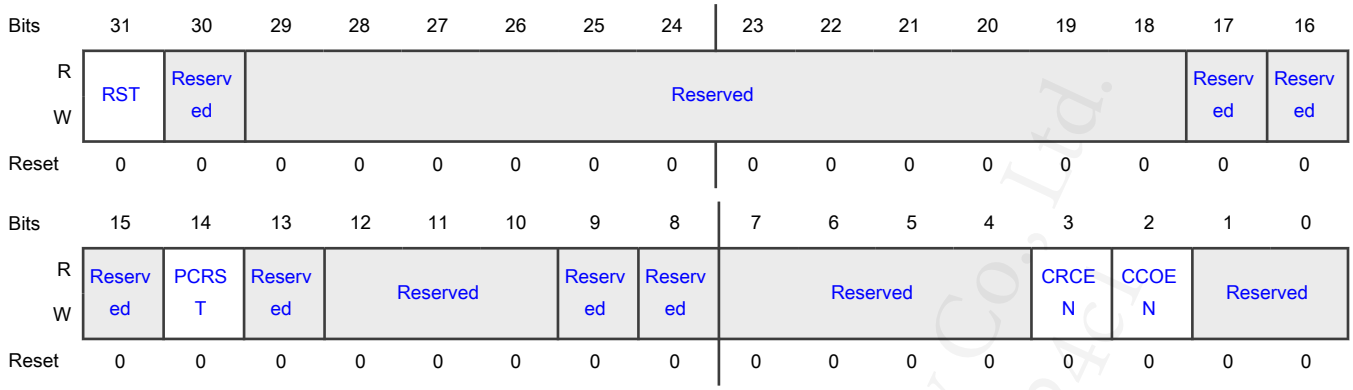
Register	Offset
ALGC	308h

Function

This register configures the behaviour of the AL.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31 RST	Aurora Channel Reset 0 to 1 transition if this field resets the channel and begins the training sequence.
30 —	Reserved
29-18 —	Reserved
17 —	Reserved
16 —	Reserved
15 —	Reserved
14 PCRST	Protocol Converter Reset This field resets the ALC when set.
13 —	Reserved
12-10 —	Reserved
9 —	Reserved

Table continues on the next page...

Table continued from the previous page...

Field	Function
8 —	Reserved
7-4 —	Reserved
3 CRCEN	CRC Enable This field enables the CRC. 0b - CRC generation is disabled for TX data. 1b - CRC is inserted for TX frames.
2 CCOEN	Clock Compensation Override Enable This field controls a debug feature which allows the duration between clock compensation sequences to be reduced to 1280 cycles. 0b - Clock compensation counter times out at 7427 cycles. 1b - Clock compensation counter times out at 1280 cycles.
1-0 —	Reserved

78.4.9 AL Training Control (ALTC)

Offset

Register	Offset
ALTC	30Ch

Function

This register configures static training and timeout controls of AL.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	STE	AHD	BHD	VHD	Reserved	Reserved	Reserved			ATC				Reserved		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Reserved		BTC				Reserved						VTC			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c84626e5-4821-48a8-b251-590a3ab624c1

Fields

Field	Function
31 STE	Static Training Enable This field enables static training. 0b - Reserved 1b - Static (timer based) training method is used for Aurora initialization. Training is established based on ATC/BTC/VTC timers.
30 AHD	Hold in Align 0b - Follows Aurora training sequence 1b - AL remains in align until this bit is cleared.
29 BHD	Hold in Bond 0b - Follows Aurora training sequence 1b - AL remains in bond until this bit is cleared.
28 VHD	Hold in Verify 0b - Follows Aurora training sequence 1b - AL remains in verify until this bit is cleared.
27 —	Reserved
26 —	Reserved
25-23 —	Reserved
22-19 ATC	Align Timer Count 0000b - Remain in align for 16 cycles. 0001b - Remain in align for 24 cycles. 0010b - Remain in align for 32 cycles. 0011b - Remain in align for 48 cycles. 0100b - Remain in align for 64 cycles. 0101b - Remain in align for 96 cycles. 0110b - Remain in align for 128 cycles. 0111b - Remain in align for 192 cycles. 1000b - Remain in align for 256 cycles. 1001b - Remain in align for 384 cycles. 1010b - Remain in align for 512 cycles.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	1011b - Remain in align for 768 cycles. 1100b - Remain in align for 1024 cycles. 1101b - Remain in align for 1536 cycles. 1110b - Remain in align for 2048 cycles. 1111b - Remain in align for 3072 cycles.
18-14 —	Reserved
13-10 BTC	Bond Timer Count 0000b - Remain in bond for 16 cycles. 0001b - Remain in bond for 24 cycles. 0010b - Remain in bond for 32 cycles. 0011b - Remain in bond for 48 cycles. 0100b - Remain in bond for 64 cycles. 0101b - Remain in bond for 96 cycles. 0110b - Remain in bond for 128 cycles. 0111b - Remain in bond for 192 cycles. 1000b - Remain in bond for 256 cycles. 1001b - Remain in bond for 384 cycles. 1010b - Remain in bond for 512 cycles. 1011b - Remain in bond for 768 cycles. 1100b - Remain in bond for 1024 cycles. 1101b - Remain in bond for 1536 cycles. 1110b - Remain in bond for 2048 cycles. 1111b - Remain in bond for 3072 cycles.
9-4 —	Reserved
3-0 VTC	Verify Timer Count 0000b - Remain in verify for 16 cycles. 0001b - Remain in verify for 24 cycles. 0010b - Remain in verify for 32 cycles. 0011b - Remain in verify for 48 cycles. 0100b - Remain in verify for 64 cycles.

Table continues on the next page...

Table continued from the previous page...

Field	Function
	0101b - Remain in verify for 96 cycles. 0110b - Remain in verify for 128 cycles. 0111b - Remain in verify for 192 cycles. 1000b - Remain in verify for 256 cycles. 1001b - Remain in verify for 384 cycles. 1010b - Remain in verify for 512 cycles. 1011b - Remain in verify for 768 cycles. 1100b - Remain in verify for 1024 cycles. 1101b - Remain in verify for 1536 cycles. 1110b - Remain in verify for 2048 cycles. 1111b - Remain in verify for 3072 cycles.

78.4.10 LVDS TX IO Configuration (LVDSTX)

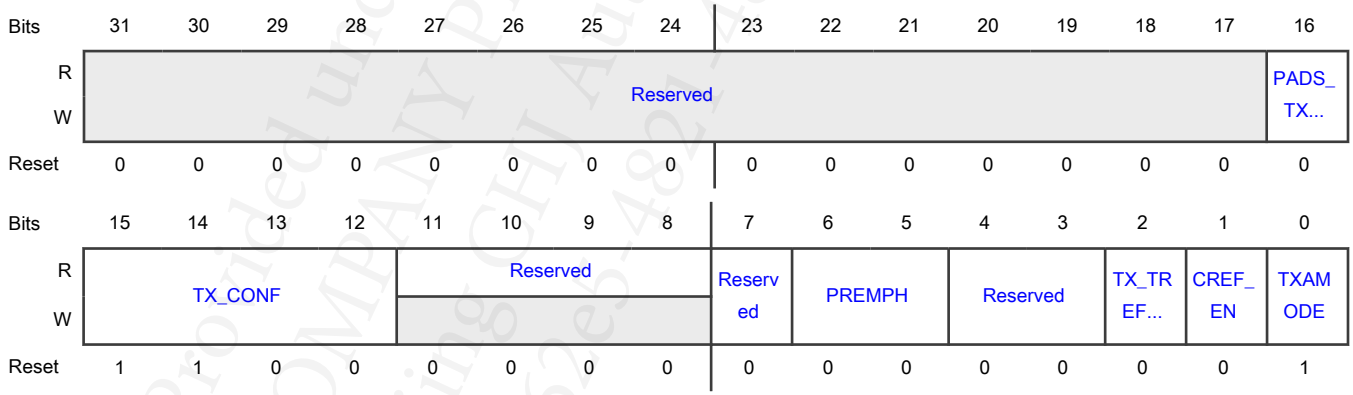
Offset

Register	Offset
LVDSTX	440h

Function

This register controls the Aurora LVDS TX pads.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c467e5-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-17 —	Reserved
16 PADS_TX_CONF_EN	PADS TX Configuration Enable Enables or disables the TX_CONF field to select the transmitter output stage current drive. The value of this field defaults to 0 that uses the transmit configuration from the fuse. This value is configured in the chip for optimum settings. You may require to override the internal default settings on early silicon, or if settings other than the default are needed in special circumstances. 0b - PADS TX configuration disabled 1b - PADS TX configuration enabled
15-12 TX_CONF	LVDS PADS TX Control Configuration Controls the drive current of transmitter output. Each enabled bit of this field adds approximately 16% drive. Typically, a setting of 1100 generates a 10mA drive, which is optimal for impedance matching. Setting the current to too high can lead to a higher inter-symbol interference (ISI) and a low setting can lead to impedance mismatch or reflections. See the following table for bit settings and their corresponding ballpark values.
11-8 —	Reserved
7 —	Reserved
6-5 PREMPH	Pre-Emphasis Configuration If enabled, adds up to 3 dB gain in differential swing, with 0b11 being the highest gain setting. However, in addition to increasing the differential signal height, the signal jitter could increase by approximately 5 to 10 ps.
4-3 —	Reserved

Bit setting	Drive current
0b0000	7 mA
0b1000	8.5 mA
0b1100	10 mA
0b1110	11.5 mA
0b1111	13 mA

Table continues on the next page...

Table continued from the previous page...

Field	Function
2 TX_TREF_EN	TX LVDS Termination Reference Enable This field enables the internal 100 Ω termination. 0b - Disable 1b - Enable
1 CREF_EN	Current Reference Enable This field enables the current reference control signal. 0b - Disable 1b - Enable
0 TXAMODE	TX Aurora pad mode enable 0b - Disable 1b - Enable

78.4.11 LVDS RX IO Configuration (LVDSRX)

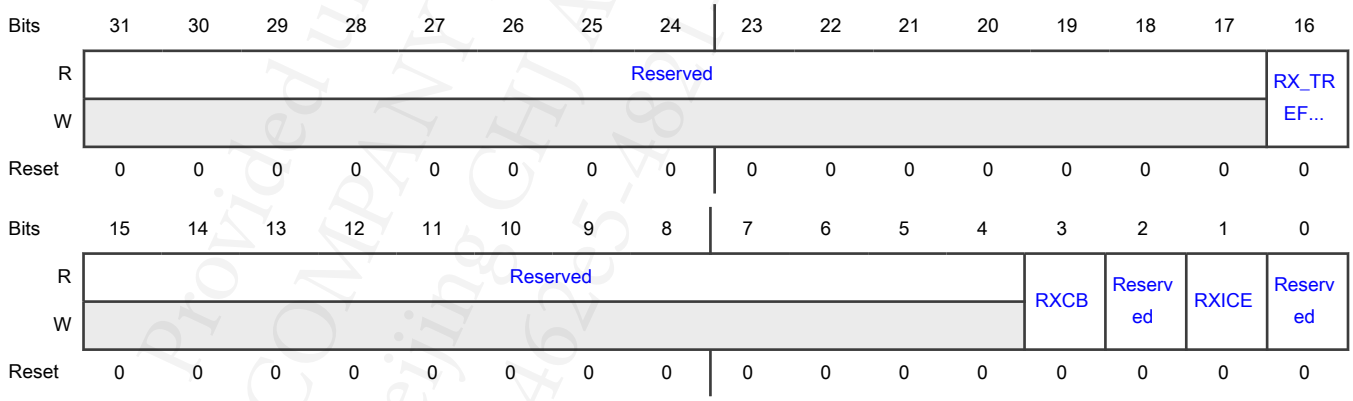
Offset

Register	Offset
LVDSRX	444h

Function

This register has configuration details of the LVDS Receiver to receive the Aurora PLL reference clock from an external source.

Diagram



Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462c5-48a2-48a3-b251-390d3ab624c1

Fields

Field	Function
31-17 —	Reserved
16 RX_TREF_EN	RX LVDS Termination Reference Enable This field enables the internal 100 Ω termination. 0b - Disable 1b - Enable
15-4 —	Reserved
3 RXCB	RX LVDS Current Boost This field is set to boost RX IO current. 0b - Current reference is 100 µA. 1b - Current reference is 1 mA.
2 —	Reserved
1 RXICE	RX Input Clock Enable 0b - Disable input buffer 1b - Enable input buffer
0 —	Reserved

78.4.12 LVDS TX OBE Configuration (LVDSTXOBE)

Offset

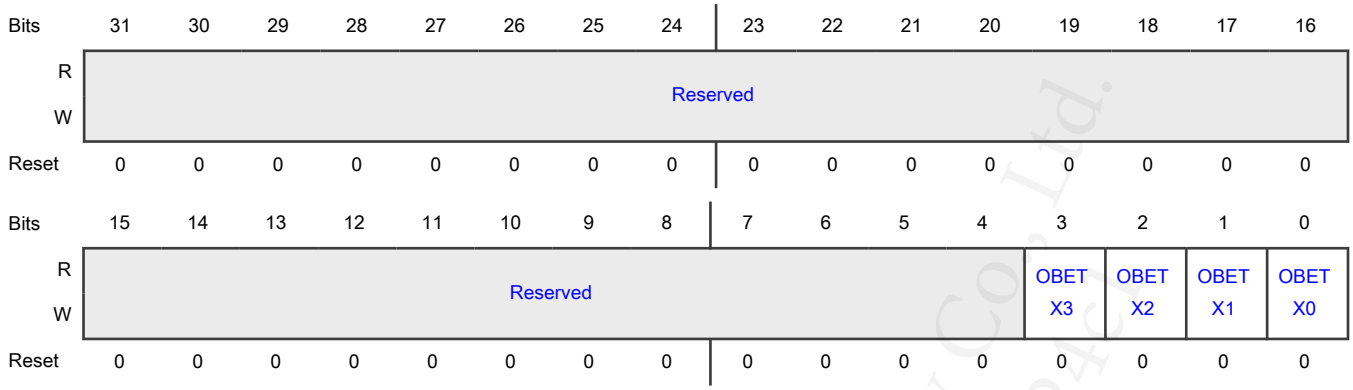
Register	Offset
LVDSTXOBE	448h

Function

This register controls the Aurora LVDS TX pads Output Buffer Enable.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Diagram



Fields

Field	Function
31-4 —	Reserved
3 OBETX3	TX-3 Aurora pad output buffer enable 0b - Disable 1b - Enable
2 OBETX2	TX-2 Aurora pad output buffer enable 0b - Disable 1b - Enable
1 OBETX1	TX-1 Aurora pad output buffer enable 0b - Disable 1b - Enable
0 OBETX0	TX-0 Aurora pad output buffer enable 0b - Disable 1b - Enable

78.4.13 CIA Control (CIAC)

Offset

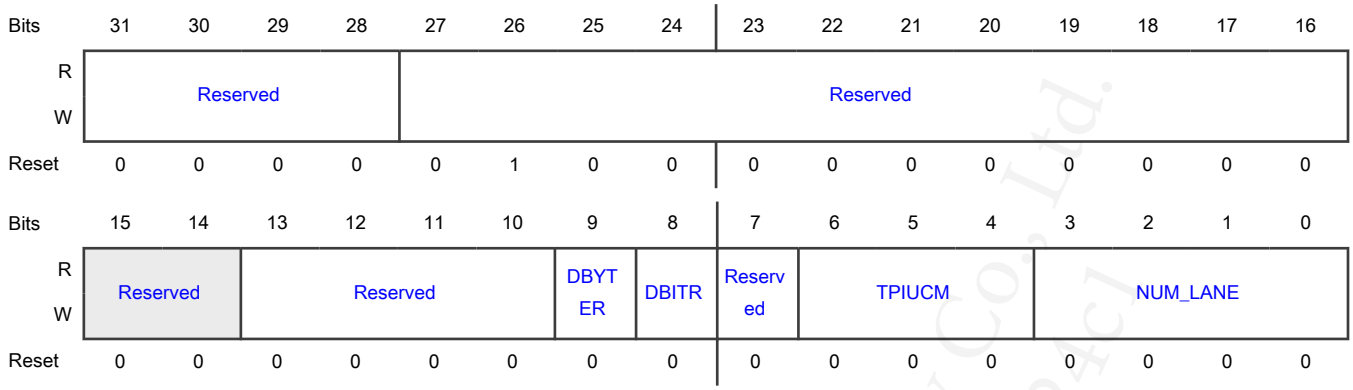
Register	Offset
CIAC	480h

Function

This register contains control and status fields for the CoreSight Interface Adaptor.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co.,Ltd.
8c462e5-4821-48a8-b251-390d3ab624c

Diagram



Fields

Field	Function
31-28 —	Reserved
27-16 —	Reserved
15-14 —	Reserved
13-10 —	Reserved
9 DBYTER	TPIU data bytes is reversed 0b - Disable byte reversal 1b - Enable byte reversal
8 DBITR	TPIU data bit reversal in a byte selection. 0b - Disable bit reversal 1b - Enable bit reversal
7 —	Reserved
6-4 TPIUCM	TPIU Control Mode 000b - Continuous and format mode. 001b - Normal mode. Only trace data is captured. 111b - Trace disable mode.
3-0	Number of Aurora Lanes

Table continues on the next page...

Table continued from the previous page...

Field	Function
NUM_LANE	Sets the number of Aurora lanes. 0000b - 0 Lanes. 0010b - 2 Lanes. 0100b - 4 Lanes.

78.4.14 Aurora Trace Port Enable (ATPE)

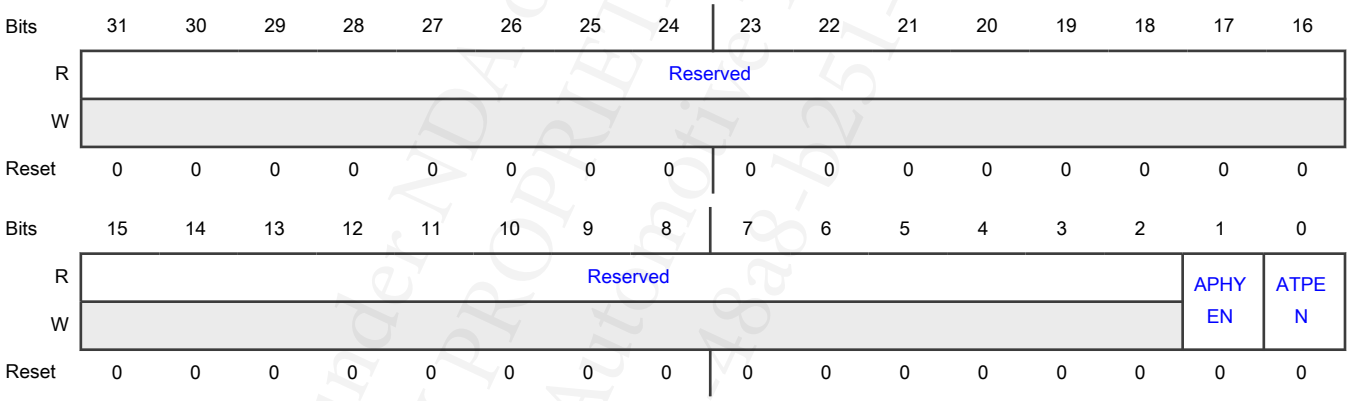
Offset

Register	Offset
ATPE	488h

Function

This register enables the ATP.

Diagram



Fields

Field	Function
31-2 —	Reserved
1 APHYEN	Aurora PHY Enable 0-AP Disable 1-AP Enable
0 ATPEN	Aurora Trace Port Enable 0b - Disable 1b - Enable

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c461e5-4821-48a8-b251-390d3ab624c1

78.4.15 Integration Mode Control (ITCTRL)

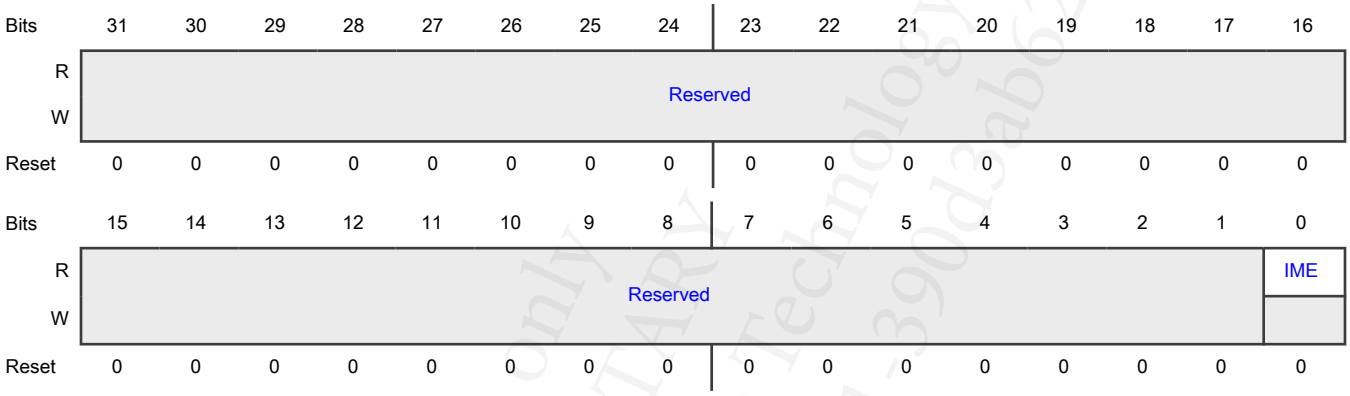
Offset

Register	Offset
ITCTRL	F00h

Function

This register indicates the status of the Arm Integration Mode Control Register.

Diagram



Fields

Field	Function
31-1 —	Reserved
0 IME	The bit field is always 0 indicating ATP only supports functional mode.

78.4.16 Claim Tag Set (CLAIMSET)

Offset

Register	Offset
CLAIMSET	FA0h

Function

This register indicates the status of the Arm Claim Tag Set Register.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462e5_4821_49a8_b251_390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SET															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SET															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 SET	This field is always 0 indicating no claim tag is implemented.

78.4.17 Claim Tag Clear (CLAIMCLR)

Offset

Register	Offset
CLAIMCLR	FA4h

Function

This register indicates the status of the Arm Claim Tag Clear Register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CLR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482f-48a8-b251-390d3ab624c1

Fields

Field	Function
31-0 CLR	This field is always 0 indicating no effect on claim tag.

78.4.18 Device Affinity 0 (DEVAFF0)

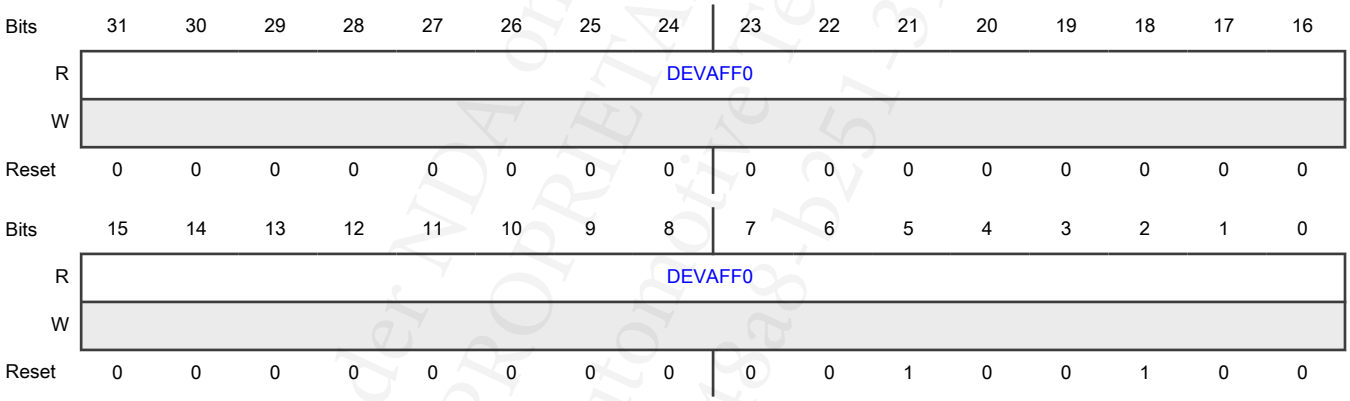
Offset

Register	Offset
DEVAFF0	FA8h

Function

This register indicates the status of the Arm Device Affinity Register 0. This register enables a debugger to determine if two components have an affinity with each other.

Diagram



Fields

Field	Function
31-0 DEVAFF0	Value of this field is set to 0x24.

78.4.19 Device Affinity 1 (DEVAFF1)

Offset

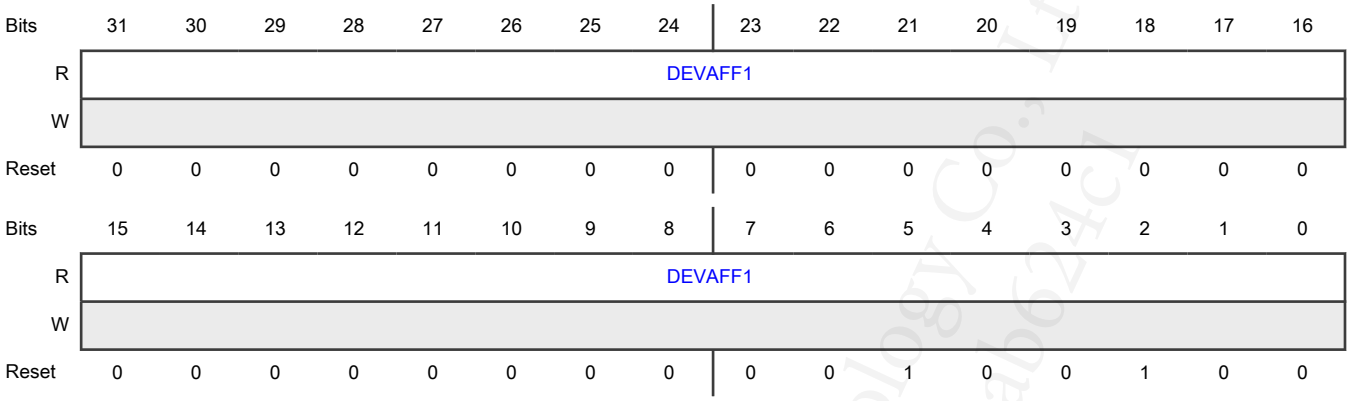
Register	Offset
DEVAFF1	FACh

Provided under NDA only
COMPANY PROPRIETARY
Beijing C-TJ Automotive Technology Co., Ltd.
c8c462e5-4821-43a8-b251-390a3ab624c1

Function

This register indicates the status of the Arm Device Affinity Register 1. This register enables a debugger to determine if two components have an affinity with each other.

Diagram



Fields

Field	Function
31-0 DEVAFF1	Value of this field is set to 0x24.

78.4.20 Lock Access Register (LAR)

Offset

Register	Offset
LAR	FB0h

Function

This register is used to enable write access to device registers. A write of 0xC5ACCE55 to this register enables further write access to this device. An invalid write has the affect of removing write access. This is applicable only for core access. For debug mode this is invalid. This is write only register.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-482f-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W	KEY															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	KEY															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 KEY	The value of this register must match with the expected key when the most significant bit of address is 0 (core access). Then writes are enabled for the other registers. In debug mode (most significant bit of address is 1), this key comparison is ignored and full write access is permitted.

78.4.21 Lock Status Register (LSR)

Offset

Register	Offset
LSR	FB4h

Function

This register indicates the status of the Arm Lock Status Register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved													nTT	SLK	SLI
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46e5-482-48a8-b251-390...8ab624c1

Fields

Field	Function
31-3 —	Reserved
2 nTT	Register size indicator. Always 0. Indicates that the LAR is implemented as 32-bit.
1 SLK	Software Lock Status. Returns the present lock status of the device. 0b - Lock matched and write operations are permitted. 1b - Lock not matched and write operations are not permitted.
0 SLI	This field is always 1 and indicates that Software Lock control mechanism exists for this device.

78.4.22 Authentication Status (AUTHSTATUS)

Offset

Register	Offset
AUTHSTATUS	FB8h

Function
This register indicates the status of the Arm Authentication Status Register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R W	Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								AUTHSTATUS							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8	Reserved

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-4ca8-b251-390d8a1624c1

Table continued from the previous page...

Field	Function
—	
7-0 AUTHSTATUS	This field is always 0, indicating that debug level authentication status is not supported.

78.4.23 Device Architecture (DEVARCH)

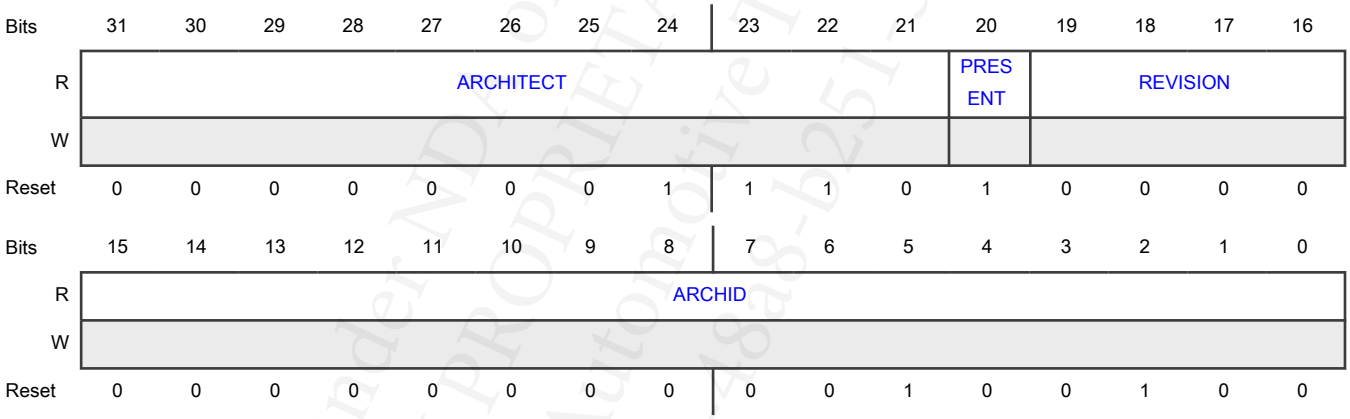
Offset

Register	Offset
DEVARCH	FBCh

Function

This register indicates the status of the Arm Device Architecture Register.

Diagram



Fields

Field	Function
31-21 ARCHITECT	Defines the architect of the component.
20 PRESENT	Indicates the presence of this register. This field always returns 1 which indicates that the DEVARCH register is present.
19-16 REVISION	Architecture revision Returns the revision of the architecture that the ARCHID field specifies.

Table continues on the next page...

Table continued from the previous page...

Field	Function
15-0 ARCHID	Architecture ID Returns a value that identifies the architecture of the component.

78.4.24 Device Configuration 2 (DEVID2)

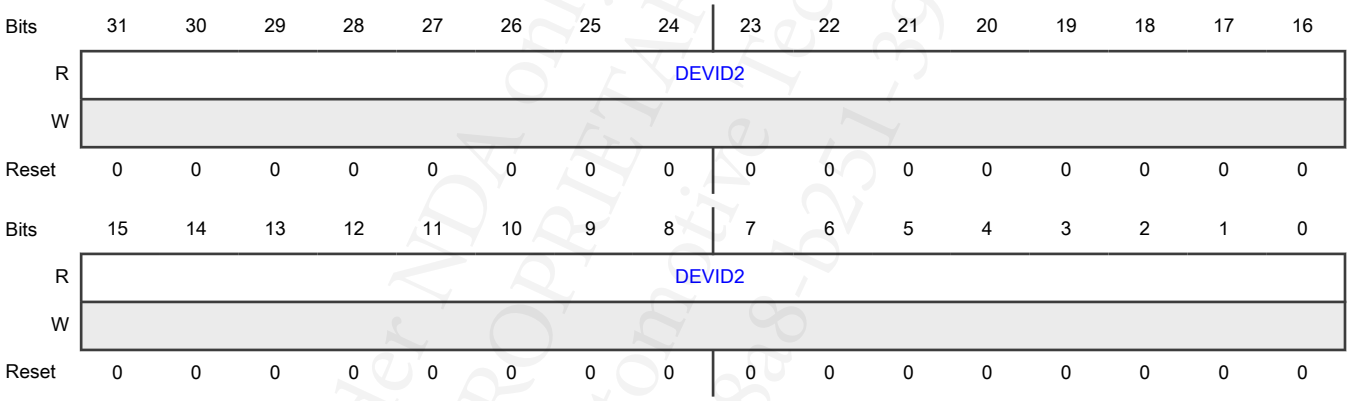
Offset

Register	Offset
DEVID2	FC0h

Function

This register indicates the status of the Arm Device Configuration Register 2.

Diagram



Fields

Field	Function
31-0 DEVID2	The value of the field is always 0.

78.4.25 Device Configuration 1 (DEVID1)

Offset

Register	Offset
DEVID1	FC4h

Provided under NDA only
COMPANY PROPRIETARY
Beijing CFI Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-590d3ab624c1

Function

This register indicates the status of the Arm Device Configuration Register 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DEVID1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DEVID1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-0 DEVID1	The value of the field is always 0.

78.4.26 Device Configuration (DEVID)

Offset

Register	Offset
DEVID	FC8h

Function

This register indicates the status of the Arm Device Configuration Register.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DEVID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DEVID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c46225-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-0 DEVID	The value of the field is always 0.

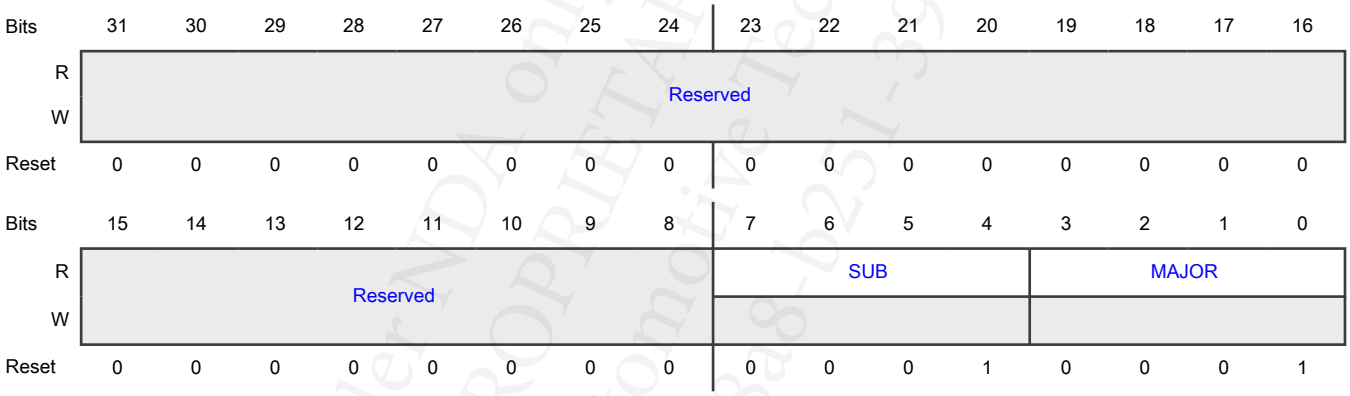
78.4.27 Device Type Identifier (DEVTYPE)

Offset

Register	Offset
DEVTYPE	FCCh

Function
This register indicates the status of the Arm Device Type Identifier Register.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-4 SUB	Sub Type This field is always 0001b which indicates the major type is Trace Port.
3-0 MAJOR	Major Type This field is always 0001b which indicates the major type is Trace Sink.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8a625-4821-4848-b251-590d3ab624c1

78.4.28 Peripheral Identification Register 4 (PIDR4)

Offset

Register	Offset
PIDR4	FD0h

Function

This register indicates the status of the Arm Peripheral Identification Register 4.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								SIZE				DES_2			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fields

Field	Function
31-8 —	Reserved
7-4 SIZE	4KB Count This field indicates the total contiguous size of the memory window used by this component in powers of 2 from the standard 4 KB. This field is always 0 which indicates the memory window is 4 KB.
3-0 DES_2	JEP106 Continuation Code This field is always 0.

78.4.29 Peripheral Identification Register 0 (PIDR0)

Offset

Register	Offset
PIDR0	FE0h

Function

This register indicates the status of the Arm Peripheral Identification Register 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								PART_0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

Fields

Field	Function
31-8 —	Reserved
7-0 PART_0	Part number, bits[7:0] This field is always 0x14.

78.4.30 Peripheral Identification Register 1 (PIDR1)

Offset

Register	Offset
PIDR1	FE4h

Function

This register indicates the status of the Arm Peripheral Identification Register 1.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								DES_0				PART_1			
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	0	1	0	1	0

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-8 —	Reserved
7-4 DES_0	JEP106 identification code, bits[3:0] This field is always 0xE.
3-0 PART_1	Part number, bits[11:8] This field is always 0xA.

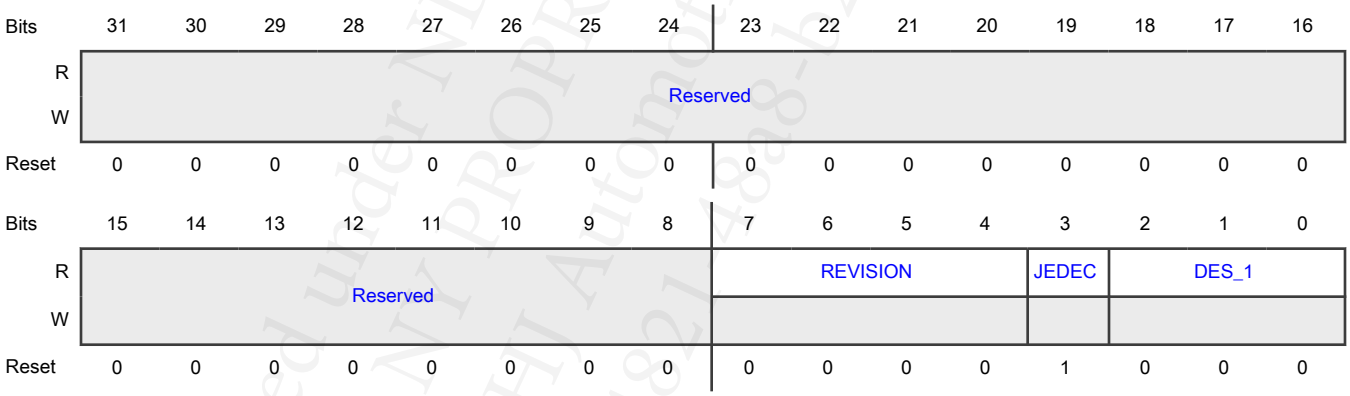
78.4.31 Peripheral Identification Register 2 (PIDR2)

Offset

Register	Offset
PIDR2	FE8h

Function
This register indicates the status of the Arm Peripheral Identification Register 2.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-4 REVISION	Revision This field is always 0.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Table continued from the previous page...

Field	Function
3 JEDEC	This bit is always set which indicates that a JEDEC assigned value is used.
2-0 DES_1	JEP106 identification code, bits[6:4] This field is always 0x0.

78.4.32 Peripheral Identification Register 3 (PIDR3)

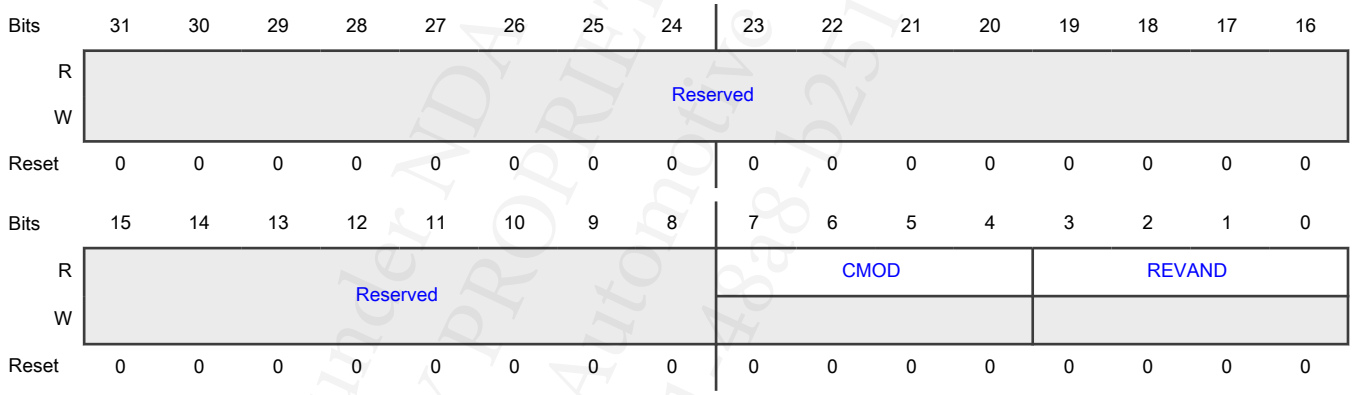
Offset

Register	Offset
PIDR3	FECh

Function

This register indicates the status of the Arm Peripheral Identification Register 3.

Diagram



Fields

Field	Function
31-8 —	Reserved
7-4 CMOD	Customer Modified This field is always 0.
3-0 REVAND	RevAnd This field is always 0.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

78.4.33 Component Identification Register 0 (CIDR0)

Offset

Register	Offset
CIDR0	FF0h

Function

This register indicates the status of the Arm Component Identification Register 0.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								PRMBL_0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1

Fields

Field	Function
31-8 —	Reserved
7-0 PRMBL_0	Preamble This field is always 0x0D.

78.4.34 Component Identification Register 1 (CIDR1)

Offset

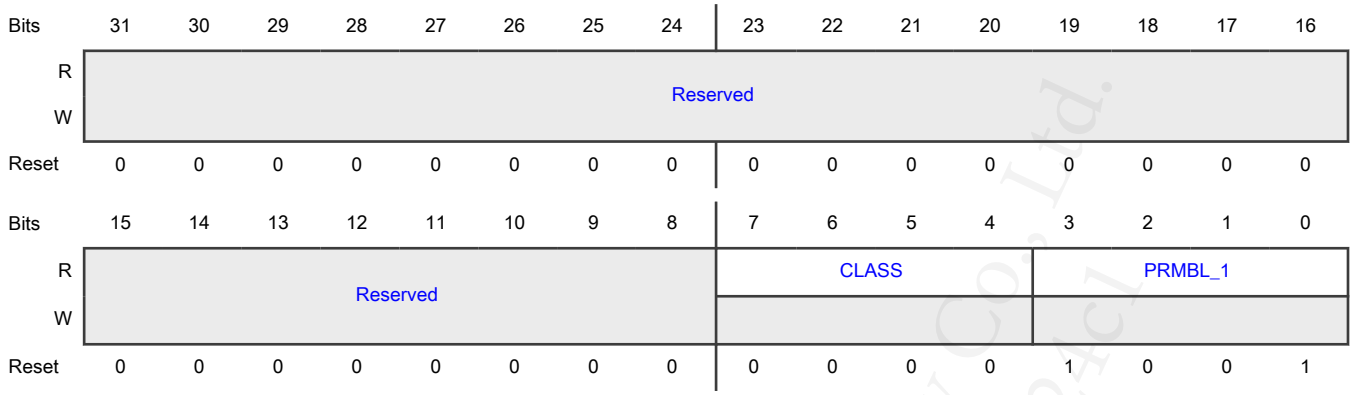
Register	Offset
CIDR1	FF4h

Function

This register indicates the status of the Arm Component Identification Register 1.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8462e5_4891_48a8_b251_390d3ab624c1

Diagram



Fields

Field	Function
31-8 —	Reserved
7-4 CLASS	Component class This field is always 0x0 which indicates a generic verification component.
3-0 PRMBL_1	Preamble This field is always 0x9.

78.4.35 Component Identification Register 2 (CIDR2)

Offset

Register	Offset
CIDR2	FF8h

Function

This register indicates the status of the Arm Component Identification Register 2.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CIU Automotive Technology Co., Ltd.
c8c462e5-4621-48a8-b251-390d3ab624c1

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								PRMBL_2							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Fields

Field	Function
31-8 —	Reserved
7-0 PRMBL_2	Preamble This field is always 0x05.

78.4.36 Component Identification Register 3 (CIDR3)

Offset

Register	Offset
CIDR3	FFCh

Function

This register indicates the status of the Arm Component Identification Register 3.

Diagram

Bits	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bits	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								PRMBL_3							
W																
Reset	0	0	0	0	0	0	0	0	1	0	1	1	0	0	0	1

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4625-4821-48a8-b251-390d3ab624c1

Fields

Field	Function
31-8 —	Reserved
7-0 PRMBL_3	Preamble This field is always 0xB1.

78.5 Functional description

78.5.1 CoreSight interface adaptor

The trace input interface to the ATP matches the output signals from the TPIU. The CoreSight Interface Adaptor (CIA) receives the data in TPIU standard format and stores it in an internal FIFO. The FIFO stored data is formatted as per the Aurora protocol and sent to AL.

The CIA consist of the logic to establish the connection between the TPIU and the AL. The CIA includes TPIU interface, buffer, AL formatter, APB to IPS conversion, memory mapping and configurations.

78.5.1.1 CIA functional blocks

The CIA consists of five functional modules:

- TPIU interface module
- Buffer management module
- AL interface module
- Register configuration module
- Clock gating module

The CIA block level overview and interface are shown in the following figure.

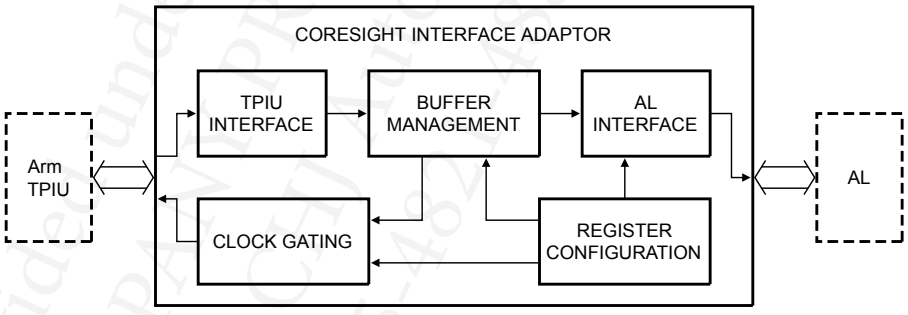


Figure 608. CIA functional blocks

The CIA block supports bit/byte reversal feature. CIAC[DBYTER] and CIAC[DBITR] are used in conjunction to implement the reversal feature as specified in the following table.

Table 644. Reversal feature

DBITR	DBYTER	output_DATA (31:0)
1	0	TPIU_DATA – (24:31)(16:23)(8:15)(0:7)

Table continues on the next page...

Table 644. Reversal feature (continued)

DBITR	DBYTER	output_DATA (31:0)
1	1	TPIU_DATA – (0:7)(8:15)(16:23)(24:31)
0	1	TPIU_DATA – (7:0)(15:8)(23:16)(31:24)
For rest of the combination		TPIU_DATA (31:0)

If TPIU_DATA (31:0) is input data from TPIU to ATP, then output_DATA (31:0) is output to the pads.

78.5.1.2 TPIU interface module

TPIU Interface Module (TIM) interfaces with Arm-TPIU, takes the data, and sends it to a buffer. TPIU trace clock input (TRACECLKIN) is provided by CIA clock gating module. Trace data is collected by considering gated clock and clock enable. As per TPIU configuration, each clock is provided by gated clock. TPIU gives 32-bit data on negative edge of the clock provide by CIA clock gating module. The data is sampled with the continuous 500 MHz clock. TIM takes 32-bit data and every alternate clock cycle of enable, it combines the data to make a 64-bit block of data. TIM generates write enable pulse of one clock period and gives the enable pulse to buffer management module. The buffer management module writes this data into the FIFO.

78.5.1.3 Buffer management module

Buffer Management Module (BMM) maintains the CIA FIFO and generates the signals related to AL internal FIFO. CIA FIFO receives the data and write enable signal from TPIU interface module and read enable and read data from AL interface module, these modules consider full and empty status flags before generating corresponding signals. AL FIFO status is maintained based on 8-credit flow, it increments with tx_valid generated by NIM and decrements with data_accept which is given by AL as received data acknowledgement.

78.5.1.4 AL interface module

AL Interface Module (AIM) interfaces with AL IP. AIM takes data from buffer management module and converts it into an AL compatible format and passes it to AL. AIM continuously polls the AL ready signal and whenever the AL is ready, AIM sends data and corresponding tx_valid signal. Before sending valid pulse, AIM reads data from FIFO and sends FIFO read pulse. To generate this AIM needs to satisfy these conditions:

- AL Aurora ready signal should be active.
- AL 8-credit FIFO should not be in almost full condition.
- CIA 8 depth internal FIFO should not be in almost empty condition.

These FIFO signals are managed by buffer management module. Ready and data accept signals are directly received from AL. It also generated start of the frame SOF and end of the frame EOF signals based on the Aurora frame length configured by user. It is programmable up to 8192.

78.5.1.5 Register configuration module

CIA module manages the register configuration required for ATP. It configure the registers of AL, AP, PLL and PADS. To configure these registers, CIA has APB interface. It takes the data over APB, then it convert the APB signals to IPS format. It decodes the corresponding register value from IPS interface and then it gives the value to the corresponding IP port.

78.5.1.6 Clock gating module

CIA interact with the Arm-TPIU in the SOC Arm debug infrastructure. TPIU uses 32-bit wide data path to send data to CIA. CIA provides 500 MHz gated clock to TPIU-TRACECLKIN port, same is used by TPIU to send the data. This clock is the output of clock gate (CG). This clock gating is required to back pressure TPIU as it does not have any control signals or mechanism to control the TPIU output data. To avoid the data loss of TPIU, the TRACECLKIN clock throttling implemented by non-continuous trace clock to Arm-TPIU.

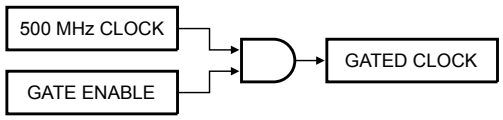


Figure 609. CIA clock gating

This clock gating is implemented when the CG override and CG by-pass are not active. If CG override is active, then the 500 MHz clock in not gated, and is continuously given to TPIU.

78.5.2 Aurora link

The Aurora Link (AL) consists of the logic required for Aurora protocol 8B/10B and to implement the connection up to the physical-layer SerDes and transceivers. The AL includes Aurora control and data multiplexing, data-encoding and striping, and the Physical Coding Sublayer (PCS) portion of the protocol. The following figure details the AL flow:

- Debug data in the frame format from multiple clients is collected by the TPIU.
- Debug data is framed by CIA as per AL requirement.
- Messages are packetized and sent across an asynchronous clock boundary-crossing gasket to the Aurora Protocol Engine (APE), which performs all the logical layer functions and flow control.
- Data from the APE is sent on to the Aurora Lane Control (ALC) block, which performs data-striping and sizing, based on the number of lanes selected for debug. The number of lanes for the AL is configurable as 0, 2 or 4 lanes.
- ALC also performs the PCS functions of the link. The PCS drives the Aurora Physical (AP) interface. The AP, in conjunction with the AL, supports one-way simplex high-speed serial communication with an external debugger.

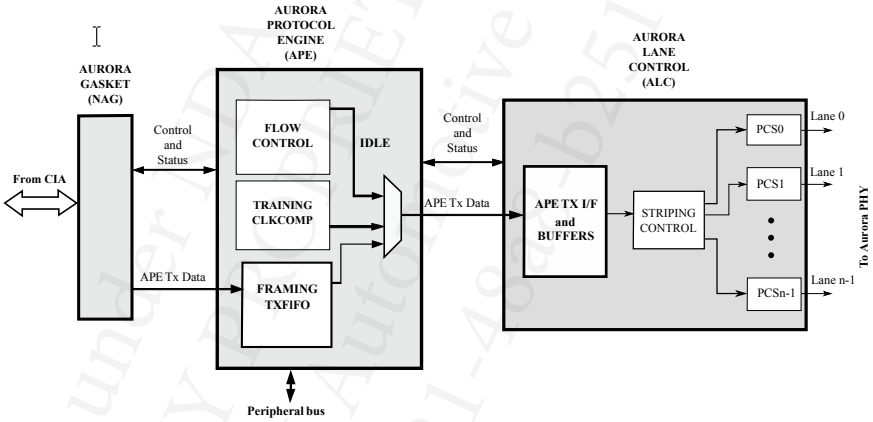


Figure 610. AL block diagram

78.5.2.1 AL functional blocks

During normal operation, the CIA reads data from TPIU and pushes data into the AL at a sufficiently high rate to ensure that the link does not become data-starved even in its highest-bandwidth configuration (4 lanes operating at 5 GHz). The AL three main blocks then perform the buffering, domain-crossing, data-packaging, and protocol translations to prepare the data stream for transmission through the AP. The work is divided as follows:

- CIA Interface (AG): The CIA and Aurora Gasket use a credit/debit interface to send data over a (possibly) multiple-stage connection, with the CIA pushing data out whenever possible and the gasket informing the CIA whenever it makes space available.
- Platform/AP Domain Crossing (AG): The logic on the AP side of the link is synchronous with the AP block and clocked by the Aurora PLL transmit/receive clocks, when the logic on the CIA side is on the platform clock domain. The AG performs this domain crossing in both directions.

- Link Training (APE): The core contains the state machines and timers that control the link training procedure.
- Aurora Protocol Engine (APE): As described in the Aurora Protocol specification, native flow-control (NFC), clock-compensation, and data framing are performed by the Aurora core.
- Cyclic Redundancy Check (APE): The core performs the CRC encoding/decoding on each transmitted/received frame.
- Data Striping (ALC): The ALC stripes the data stream from the Aurora core across all available lanes in two-byte chunks. This requires resizing and repackaging the data based on the debug port configuration.
- Channel Bonding (ALC): As part of the training procedure, the ALC aligns the data streams from different lanes to ensure that the received data stream is correctly aligned across the lanes.
- 8B/10B encoding/decoding (ALC): The protocol converter contains the 8B/10B encode and decode engines for the PCS layer of the three protocols.
- Receive data elastic buffer (ALC): The protocol converter buffers the raw incoming data stream.
- Data Alignment (ALC): As part of the training procedure, the incoming serial data stream must be partitioned into discrete characters. The protocol converter determines where the character boundary is and aligns each lane accordingly. The AL supports 2 and 4 TX lane configurations.

78.5.2.2 AL link training

The Aurora link training procedure is divided into three stages:

Table 645. Aurora link training stages

Stage	Description
Lane alignment	Each transmitter sends a sequence consisting of data and comma characters. The receive side must analyze the incoming serial data stream and determine the character boundaries by attempting to detect commas.
Channel bonding	Each transmitter simultaneously sends a series of idle sequences, which include the /A/ character. The receive side checks to see if it sees the /A/ on every lane on the same cycle, skewing the lanes as necessary to bring them into alignment.
Channel verification	Each transmitter simultaneously sends the verification sequence (/V/) on all lanes, imbedded in a series of idles. If the receive side detects the sequence on all lanes at the same cycle, the channel is brought up.

78.5.2.3 AL static training

The AL only supports static training. The AL performs static training using programmable timers to determine how long to stay in each stage of the training process. There is no communication between the channel partners as to training status. Rather each partner sends the align/bond/verify sequence for a set amount of time that it is assumes is sufficient for the receive side to complete the training stage.

To enable static training, program the counters in the AL Training Control (ALTC)) and then set the STE bit. This resets the channel and initiates the training routine. The time spent in each stage can range from 16 to 3072 cycles. When a counter times out, a counter timeout signal is asserted to the APE to prompt it to move to the next stage.

Additionally, it is possible to hold the training procedure in any of the stages indefinitely, by setting one of the three hold bits (AHD/BHD/VHD). When set, these bits cause the channel to stay in align, bond, or verify until unset. This allows a user to interactively step through the training procedure by monitoring the status at the debugger side and then stepping the SOC side to the next stage.

During the Aurora training sequence, the AL generates a specific set of character sequences per the Aurora protocol specification which are to be transmitted simultaneously on all lanes.

78.5.2.4 AL transmission procedure

The CIA generates user protocol data units (PDU) to form a frame of data. The boundaries of the frame are indicated by assertion of SOF and EOF signals. The frame of data is transmitted on the Aurora channel as a channel PDU according to the following procedure:

- CRC calculation
- Padding
- Encapsulation with channel PDU delimiters
- Clock compensation

The AL calculates CRC for each frame of data according to the CRC-16 polynomial:

$$G(x) = x^{16} + x^{15} + x^2 + 1$$

When enabled, two octets of CRC information are inserted into each frame.

The Aurora channel requires that all transmissions contain an even number of symbols. In this implementation, the data is generated 8 bytes at a time and the CRC information is encoded into two bytes, therefore padding is unnecessary. Padding is therefore not supported by the APE for transmission. Received data, however, may contain pad characters and the APE processes them according to the protocol specification.

After the CRC octets are added, the frame is encapsulated with ordered sets which delimit the beginning and end of the channel PDU for identification by the channel partner. The SCP and ECP ordered sets are used to indicate start-of-channel and end-of-channel, respectively. If more than 7424 symbol pairs have been sent after the last clock compensation sequence, a new clock compensation sequence is immediately inserted into the data flow. A clock compensation sequence consists of 12 pairs of the /CC/ ordered set, and its periodic insertion allows the AL to tolerate small clock rate-differences between the transmitter and receiver.

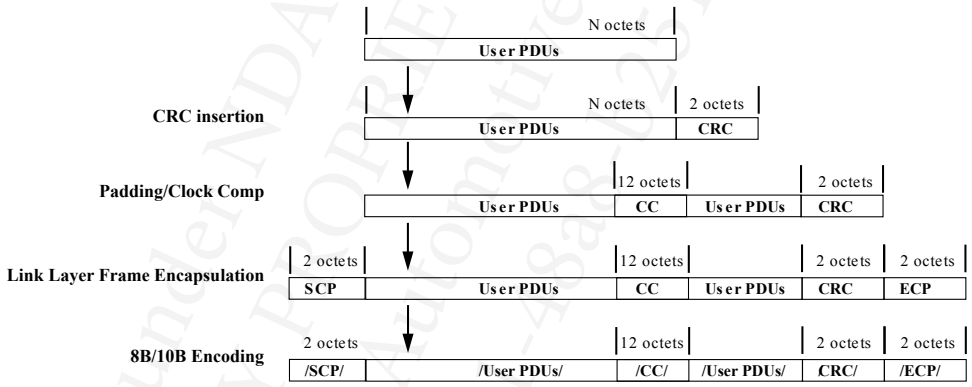


Figure 611. AL transmission procedure

78.5.2.5 AL clock compensation considerations

The Aurora Protocol specification describes clock compensation as a means for the link to compensate for clock rate differences between the transmitter and receiver. The specification requires that a clock compensation be inserted at a minimum every 10,000 symbol pairs. This allows the link to accommodate up to a 200 ppm difference in clock rates. The ALGC[CCOEN] field sets the default clock compensation time-out at 7424 cycles. However, due to the fact that the AL is restricted to only inserting clock compensation sequences after an end-of-frame, it may not be possible to always meet the Aurora specification requirement.

Because the CIA always operates in framed mode, there is one scenario that can prevent an adequate number of clock compensation sequences from being generated: the sparse data case. In the sparse data case, there are long periods of time when no data is flowing through the link; no data means no frames. The AL could end up transmitting idles for an undetermined number of cycles when waiting for the CIA to provide new data. However, a user can get around this problem by insuring that the CIA always has at least some minimal amount of data to send. In extreme sparse data cases where clock compensation

is required, the receiver could reset the CIA or use a periodic watchpoint to force data transmission, and therefore force and end-of-frame to complete clock compensation.

78.5.3 Aurora PHY

The Aurora PHY (AP), in conjunction with the AL, supports one way simplex high-speed serial communication with an external debugger. Configuration settings in the ATP configuration register specify the AP and AL lane configuration. The AL performs all logical layer functions, flow control, data-striping and sizing based on the number of data lanes available for use, and physical coding sublayer functions on the data being sent to the AP for transmission. The AP receives 8B/10B encoded data from the AL then serializes and transmits this data through a given PHY lane's Low Voltage Differential Signaling (LVDS) buffers to an external debugger.

Each AP lane consists of a data symbol character accumulator which registers the 10-bit character for that lane from the AL on the rising edge of the AL clock (AL clock = AURORA_CLK / 10). Next, a serializer takes the 10-bit character and serializes it into the individual data bits that are then transmitted at the PHY clock rate via the LVDS output drivers to the external debugger receive channel.

On the receive side of the external debugger, it is the responsibility of the external debugger to perform elastic buffering, symbol detection, symbol locking (called lane alignment in Aurora terminology), 10b/8b decoding, decode and disparity error tracking in the received data stream.

78.5.3.1 AP modes of operation

The operating mode of the AP is determined by the device control signals. Functional operation is enabled by the CIA.

Table 646. AP modes of operation

Mode	Description
Reset	During Reset all internal AP resources are reset, all output ports are disabled, and all inputs are ignored. Because there is no device clock domain logic in the AP itself, the AP is unaffected by device system reset. Depending on the AP configuration settings, the AP may activate upon exiting Reset if any Aurora lanes are enabled during this reset mode, else it remains disabled and inactive.
Link training	Before beginning trace output mode operation, the AL must go through a training process to ensure that an external debugger can correctly reconstruct the transmitted data stream. Training proceeds after the link powers up, and must complete successfully before data transmission can occur. The AL initializes and trains the Aurora link in a simplex configuration using a timer based static-training method. The initialization and training sequence described in Aurora Protocol Specification progresses from one step to the next via the use of programmable timers contained in the AL. A timer is programmed for each stage of training and as each timer expires, the training sequence proceeds to the next stage until complete. The AP is not directly aware of the link training operation, because it merely transmits the data presented to it by the AL.
Transmit	During trace output operation, it is the responsibility of the CIA to push data into the AL at a sufficiently high rate to ensure that the link does not become data-starved. The AL and AP support up to four transmit lanes. The Aurora encapsulated data is modified by the AL to ensure the appropriate number of symbols are transmitted through the AP.

78.5.3.2 AP functional blocks

High-level descriptions of the operation of the AP and its main functional blocks are given in the following table.

Table 647. AP functional blocks

Block	Description
Data symbol character accumulator	The data symbol character accumulator portion of the AP for each lane is made up of a symbol character register whose inputs are connected to the 10-bit AL character bus. The data symbol character accumulator essentially registers a 10-bit character for that lane simultaneous with the rising edge of the AL clock and presents the 10-bit quantity to the serializer at the appropriate times.
Serializer	The serializer portion of the AP for each lane takes the data captured by that lane's data symbol character accumulator and demuxes the data to provide a single serial stream of symbol data to be output by the LVDS LO buffer for that lane at the PHY clock rate. Figure 612 shows a high level timing diagram that demonstrates the operation of the serializer for a AP lane, where S0 - S9 represents a 10-bit symbol, S10 - S19 represent the next symbol, and so on. The 5 GHz pll_clk is the AURORA_CLK.

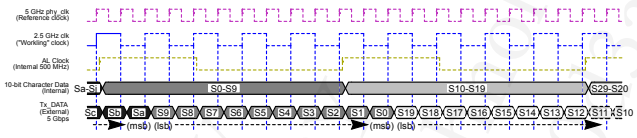


Figure 612. Serializer timing diagram

78.5.4 Aurora PLL

The Aurora PLL module provides the clocks required for ATP subsystem. The overall clock flow of ATP using Aurora PLL is shown in the following figure.

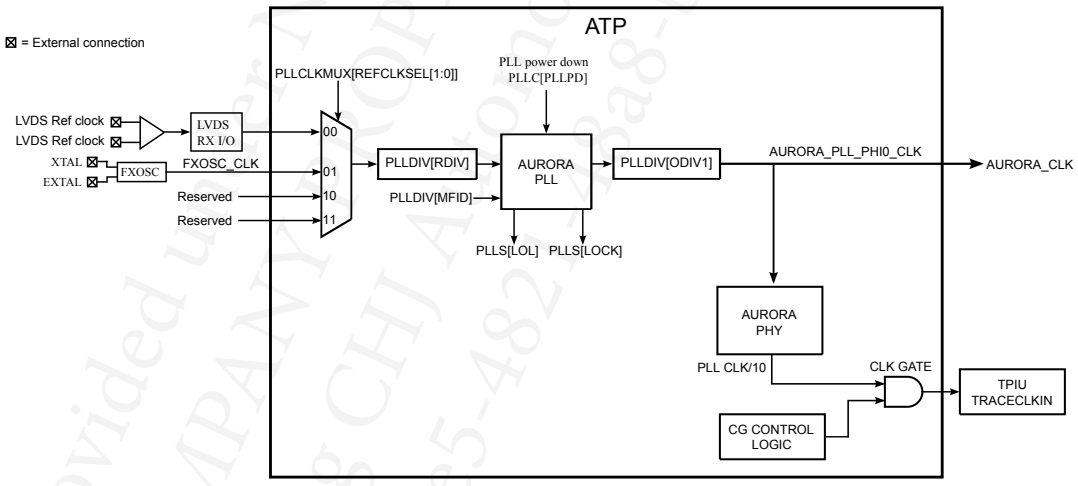


Figure 613. ATP clock flow functional block diagram

78.5.4.1 PLL clock configuration

See the following figures and the corresponding register configuration that determine the relationship between VCO frequency (f_{VCO}) and PLL reference frequency.

- When PLLDIV[RDIV] is 0:

$$f_{pll\_VCO} = f_{pll\_ref} \times (PLLDIV[MFI] \times 2)$$

Figure 614. PLL VCO frequency when PLLDIV[RDIV] is 0

- When PLLDV[RDIV] is not 0:

$$f_{pll\_VCO} = \frac{f_{pll\_ref}}{PLLDIV[RDIV]} \times (PLLDIV[MFI] \times 2)$$

Figure 615. PLL VCO frequency when PLLDIV[RDIV] is not 0

See the following figure and the corresponding register configuration that determine the relationship between reference and PLL_PHI*n* output frequencies.

$$f_{pll\_phi} = \frac{f_{VCO}}{PLLDIV[ODIV1]}$$

Figure 616. PLL PHI output frequency

When configuring the PLL, user software must not violate the maximum system clock frequency or max/min VCO frequency specification of the PLL (see this chip's data sheet for frequency limits).

NOTE

The PLL must be disabled by writing PLLC[PLLPD] = 1 before any PLL configuration or input clock are modified.

NOTE

The PLL must be disabled by writing PLLC[PLLPD] = 1 for at least 5 µs before enabling the PLL (writing PLLC[PLLPD] = 0).

The recommended procedure to program the PLL and engage normal mode is shown in Initialization sequence.

78.6 Initialization sequence

The ATP initialization sequence is described as follows:

1. Aurora IO configuration:
    a. Configure the ATP lock key: Write C5AC_CE55h to LAR[KEY] in core access mode only.
    b. Write 1 to LVDSRX[RXICE], LVDSRX[RXCB], and LVDSRX[RX_TREF_EN].

       You must ensure that there is no termination on the PCB because Aurora is AC coupled.
    c. Write 1 to LVDSTX[TXAMODE] and LVDSTX[CREF_EN].
    d. Wait for 1 µs.
    e. Write 1 to LVDSTXOBE[OBETX1], LVDSTXOBE[OBETX2], LVDSTXOBE[OBETX3], and LVDSTX[TX_TREF_EN].

       You must:
       - Ensure that the aforementioned fields are enabled together.
       - Avoid writing 1 to LVDSTX[TX_TREF_EN] and 0 to any of the OBETX fields of LVDS TX OBE Configuration (LVDSTXOBE).

       This enables the source or dual termination scheme for LVDS Tx pads for a better jitter performance.

       If the board design has losses and you have a smaller eye height, you can keep LVDSTX[TX_TREF_EN] = 0 to improve the eye height.

In case of high board loss, you can write 11b to LVDSTX[PREMPH] to get faster slew rates and increased eye height at the cost of additional power consumption.

2. Clock output to TPIU had been muxed with FIRC (after reset de-assertion) to complete auto-discovery. Before configuring PLL, switch from default FIRC clock to PLL output (PLL CLK/10). Steps to switch:
    a. Set CIA Control register bit[29] to 1b1.
    b. Set CIA Control register bit[28] to 1b1.
    c. Set CIA Control register bit[29] to 1b0.
3. Aurora PLL configuration:
    a. Before configuring PLL, connect LVDS RX IO to 100MHz clock.
    b. Wait 6 µs to reset the analog PLL.
    c. Configure PLL divider
        - Configure PLL Divider (PLLDIV) Register: Address = 0x008, Configure MFID, RDIV and ODIV1.
        - Configure PLL Calibration 2 (PLLCAL_2) Register: Address = 0x018, Configure CALDPER.
    d. Configure PLL reference clock
        - Configure PLL Clock MUX (PLLCKMUX): Address = 0x020, Configure reference clock for PLL(REFCLKSEL).
    e. Configure PLL power down
        - Configure PLL Control (PLLC) Register: Address = 0x000, Configure PLLPD.
    f. Wait for PLL to lock
        - Wait 150 µs or
        - Wait for PLL Status (PLLS) register: Address = 0x004, LOCK Status bit set to 1b1.
4. ATP CIA configuration:
    a. Configure CIA Control Registers
        - Configure CIA Control (CIAC): Address = 0x480, Configure number of lanes (NUM_LANE) and TPIU Control mode (TPIUCM).
5. Aurora PHY configuration:
    a. Configure APHY enable
        - Aurora Trace Port Enable (ATPE): Address = 0x488 : Configure Aurora PHY (APHYEN).
        - Wait 1 µs to complete training.

NOTE

MISR calculations start four cycles after the APMISR[AP_MISR_START] time, and MISR calculations stop four cycles after the APMISR[AP_MISR_STOP] time.

6. Aurora Link configuration:
    a. Configure Aurora Link Control register
        - Configure AL General Control (ALGC): Address = 0x308 , CRC Enable (CRCEN) and Clock Compensation Selection (CCOEN).
        - In case of training failure, configure AL Training Control (ALTC): Address = 0x30C , AHD (Align), BHD (Bond) and VHD (Verify).
7. Aurora enable configuration:
    a. Configure ATP enable

- Aurora Trace Port Enable (ATPE): Address = 0x488 : Configure Aurora Trace Port Enable (ATPEN).

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c462e5-4821-48a8-b251-390d3ab624c1

Appendix A
Revision History

A.1 About_This_Manual module changes

• Made editorial changes.
• In Conventions, changed "Notes, cautions, and warnings" to Notes and cautions. • In Notes and cautions: — Deleted the definition for warnings (which are now deprecated). — Provided new definitions for notes and cautions.
• Added Editorial changes.
• Description of 'MOD.REG' is added in Typographic notation

A.2 Introduction changes

• In the Feature comparison table, revised the PCI express (PCIe) and SERDES rows for S32G234M. • In the Feature comparison table, removed DDR4 reference and added DDR3L reference.
• Revised the Overview section.
• In the Block Diagram, removed the DDR4 reference and added DDR3L reference.
• S32G274A is replaced with S32G2. This is a global change and implemented throughout this document and the excels embedded in this document.

A.3 Arm modules changes

• Added a point, "The MPU contains support for up to 16 regions." in Cortex-M7 core introduction
• Fixed the formatting issues of the image, "Figure 7" to ensure that it appears in the doc

A.4 FlexNoC module changes

No substantial content changes.

A.5 Chip-specific Ncore information changes

No substantial content changes.

A.6 Ncore module changes

• Added a statement about full-word access for all the registers in Ncore memory space, in Memory map and register definition.

A.7 MCM module changes

• Added reference for Arm Cortex-M7 technical reference manual in Interrupt Status and Control (ISCR) and Core Platform Control (CPCR).
• In Introduction added note "MCM is accessible using the Private Peripheral Bus of Cortex-M7".

A.8 A53_CLUSTER_GPR module changes

• In GPR08 (GPR08), for Warm Reset Status Of Core* Of Cluster* bits, updated the bit field value for 0 and 1 from "0b - Reset request completed and 1b - Reset in progress" to "0 - Reset in progress and 1 - Reset request completed".

A.9 CM7_CLUSTER_GPR module changes

No substantial content changes.

A.10 Chip-specific eDMA information changes

No substantial content changes.

A.11 eDMA3 module changes

• Added glossary. • Removed 'otherwise, a configuration error is reported' from TCDn_BITER_ELINKYES[BITER] and TCDn_BITER_ELINKNO[BITER]. • Fixed xref issue in Lockstep. • In Figure 22 replaced ACTIVE and DONE with EEOP and ESDA.
• In Testing for minor loop completion splitted the table heading for CHn_CSR and TCDn_CSR fields.
• In TCDn_NBYTES_MLOFFNO replaced TCDn word 2 with correct register (TCDn_BYTES_MLOFFNO/ TCDn_BYTES_MLOFFYES). • In TCDn_NBYTES_MLOFFYES replaced TCDn_NBYTES_™* with correct register (TCDn_BYTES_MLOFFNO/ TCDn_BYTES_MLOFFYES).

A.12 DMA_CRC module changes

No substantial content changes.

A.13 Chip-specific DMAMUX information changes

No substantial content changes.

A.14 DMAMUX module changes

• Updated the section Enabling and configuring sources, and added comments to the code lines.

- Removed the word "for example" from the phrase "for example PIT" in the sections DMA channels with periodic triggering capability, and Modes of operation.
- Minor editorial changes throughout.

A.15 Chip-specific XRDC information changes

- In Table 48, updated from Input from master to Input from master(0):
  - XRDC_MDAC8, XRDC_MDAC9, XRDC_MDAC10
- In Table 49, updated from DRAM0_COHERENT(0) to DRAM0_A53(0).
- In Table 49, updated from M7_TCM_0(0), M7_TCM_1(1), and M7_TCM_2(2) to M7_TCM_0(0), M7_TCM_1(0), and M7_TCM_2(0) for MRC11.
- In MDAC configurations, added a note, "Input DID is 0 for all masters except DMA MDAC which takes input from DMA."

A.16 XRDC module changes

- Added Glossary
- Added 'It also clears the corresponding field in DERRLOC*n* register.' in DERR_W3_*n* register description.

A.17 Application Extension-Specific SIUL2 Changes

No substantial content changes.

A.18 SIUL2 module changes

- In the MSCR and IMCR register descriptions, modified figure titles.
- In the DISR0[EIFn] field description descriptions, removed reference to "DMA request" for inapplicable field instances.
- In IREER0[IREE*n*] and IFEER0[IFEE*n*] field descriptions, corrected references to "DISR0[EIFRn]" to "DISR0[EIFn]".

A.19 SEMA42 module changes

- Added glossary.

A.20 SRC module changes

- Removed the references of cjtag from DEBUG_CONTROL[DEBUG_CLK_DISABLE]

Provided under NDA ONLY
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c42e54-8218-4848-b251-90d3a6f04c1

A.21 S32G_MAIN_GPR module changes

• Enhanced the description of the PCIE_TOGGLE register.
• Removed the CLKOUT_PAD_CTRL_REG register.
• In the the Generic Control 1 register, added a note, "In addition to programming the MC_CGM_2 to select the correct SerDes clocks, setting this bit to 1 is mandatory for SGMII mode."

A.22 S32G_STDBY_GPR module changes

No substantial content changes.

A.23 REG_PROT module changes

• Updated the #unique_4595 section — REG_PROT is a protection module that is located between the module under protection and the peripheral interface. — Writes to these protected address locations are locked using a Soft Lock Bit Register (SLBRn). Any access to address locations in the module under protection is restricted if their corresponding Soft lock fields in a Soft Lock Bit Register (SLBRn) are 1. — The configured soft-lock fields can also be write-restricted by using GCR[HLB]. When GCR[HLB]=1, you cannot write to the softlock fields. • Updated the #unique_4596 section — Write accesses for the module under protection can be restricted to the supervisor mode only. — Modules with their register slot size occupying 1 KB to 32 KB of memory-mapped address space, depending on the PROT_MEM parameter, can be dynamically locked. See Memory map and register definition. — Multiple ways are present to set the lock bits. • Updated the #unique_4597 section — This section provides a detailed description of the memory map of a module using register protection with instantiating a REG_PROT wrapper. — The resulting memory space is divided into four areas. Memory locations within Area 1 and Area 2 are a part of the module under protection. Area 3 and Area 4 are a part of REG_PROT. — See the Offset column to know the exact location of the GCR register. • Updated the #unique_4598 section — The register protection size can be. — The addresses that are protected and the register protection size depends on the chip and/or module. — If an address is unprotected, the corresponding SLBRn[SLBm] bit is always 0, regardless of what the software writes to that field. • Updated the #unique_4599 section. • Updated the #unique_4600 section. • Updated the #unique_4601 section.

Table continues on the next page...

- Updated the chapter for the following points:
  - SLBn to SLB*n*
  - SLBR*n*(SLB*m*) to SLBR*n*[SLB*m*]
  - Area to Area#
  - SLBR*n*[SLBR*m*] to SLBR*n*[SLB*m*]
  - See the REG_PROT sheet attached to this document to See the REG_PROT details file attached to this document.
- Updated the definition of PROT_MEM in #unique_4602 from "Memory protection size" to "Integration parameter that specifies the size of the module register slot protected".
- Removed the phrase, "Only a write to the mirror-module space can change a lock field to 1. Reading the mirror-module space does not change the lock fields." from the end of #unique_4599 section.
- Added a phrase, "Note that reading area #2 does not affect soft lock bits." in #unique_4599 section.
- Added the #unique_4603 back to the map and removed the definition of Module register (MR) from the #unique_4602 section.

- Switched the order of bullet points in #unique_4604 section.
- Added the phrase, "The following shows an example with PROT_MEM = 8 KB where SLBs are modified using area #2." and updated from "0Ch and 0Dh" to "000Ch and 000Dh" in #unique_4599 section.
- Added a phrase, "The PROT_MEM size varies per chips." in #unique_4597 section.

A.24 Chip-specific [S32G] information changes

- Added a topic Enabling wakeup pads
- Updated the note in Pull control of wake-up pads
- Added a note in Asynchronous wake-up sources for various modes.
- In the Wakeup/Interrupt Pull Enable Register (WIPUER_WIPDER) section, added a note " Enabling pull-ups on the wake-up pins during the Standby mode will increase the overall standby power consumption."

A.25 WKPU module changes

No substantial content changes.

A.26 Clocking changes

- In section Input clock pins corrected table alignment. No content change.
- In section SIRC corrected "POR_WDG" to "POR_WDOG".
- In section PLL SSCG Setup modified sentence before bullet for clarification.

- In section PLL corrected "HSE" to "HSE_H".

- In section PFE_MAC_0 clocking overview moved CMU_FC_46 to monitor PFE_MAC_0_TX_DIV_CLK in the diagram.

Table continues on the next page...

• In section Clock input sources Table 105 changed the CMU_FC_46 monitored clock to PFE_MAC_0_TX_DIV_CLK.
• Removed section "Built-In Self-Test configuration".
• In sections PFE_MAC_0 clocking overview and Application clock source options added SERDES_1_LANE_0_TX_CLK clock to MC_CGM2 Clock mux1 Selector.
• In section Peripheral-related clock overview corrected diagram by adding a DIV at mux 15 output between the GMAC_0_REF_CLK and GMAC_0_REF_DIV_CLK connections.

A.27 MC_CGM module changes

• Updated the description of MUX_n_DIV_UPD_STAT[DIV_STAT] field. • Updated the description of the "PCFS Step Duration" register. • Added section: Glossary.
• In Software-controlled clock multiplexer, updated the first sentence of the second bullet in the note. • Added following note at the end of the sections: Hardware-controlled clock multiplexer and Software-controlled clock multiplexer: The above flowchart steps can be preceded by points 1 and 3 below ...

A.28 DFS module changes

No substantial content changes.

A.29 PLLDIG module changes

• In section Modes of operation, added title to figure.
• In section Block diagram added description that number of output dividers can vary with the module instance.
• In PLLCAL2 register section corrected ULKCTL field description.

A.30 FXOSC module changes

• In section Initialization information subsection Initializing FXOSC: — Modified first bullet in first NOTE to clarify it is referring to the default mode when the module is enabled. — Added step to set EOCV and moved the content of the "Calculating EOCV" subsection to be part of this procedure. — Added step to confirm that the clock is stable.
• In section Initialization information: — Removed step "Configure inputs needed for FXOSC". — Removed first bullet in first NOTE on the default mode when OSC_BYP and COMP_EN fields are not set.
• In CTRL[GM_SEL] field description NOTE:

Preliminary Information
COMPANY CONFIDENTIAL

— Modified bullet on GM_SEL = 0 to apply to Crystal mode and Differential Bypass mode. — Replaced description on configuration of this field with a reference to the Initialization information section. • In section Initialization information: — Modified GM_SEL configuration description to provide guidance for each mode. — Modified NOTE bullet on GM_SEL = 0 to apply to Crystal mode and Differential Bypass mode.

A.31 Reset changes

• In Reset changes added a bullet "Boot configuration pins" and a note "To allow the boot configuration pins to be properly sampled, you must ensure that the corresponding I/O power supplies are within their specified ranges. If this is not the case, the reset sequence is not prevented from completing, but could lead to an incorrect boot".
• Removed the reference of cJTAG from the chapter.

A.32 Chip-specific MC_RGM information changes

No substantial content changes.

A.33 MC_RGM module changes

• Removed the section: PAD tristate interface. • Added section: RESET_B pin assertion and pin safe state control. • Added note in ERCTRL[ERASSERT].
• Changed the "Warning" to "Note" in Functional Event Reset Disable Register (FERD).

A.34 Boot changes

• In BootROM impact on module registers' reset values — Added a note.
• Updated Table 163.
• Changed the IVT offset for SD/MMC/eMMC entry from "0h" to "1000h" in Table 127. • In Functional description removed the note "By default, the boot ROM is configured for a flash-memory size of 1 MB. If you need to prepare an image greater than 1 MB, you must reconfigure the flash-memory size using a QuadSPI configuration parameter".
• In PHY identification and configuration: — Removed three from the text "BootROM maintains a list of RGMII PHYs against which these PHYIDn registers are compared". — Changed 2.5 seconds to 5 seconds in the texts "BootROM expects PHYs to start Autonegotiation after every reset and wait for autonegotiation to complete within 5 seconds" and in the first bullet "Autonegotiation fails within 5 seconds".

Table continues on the next page...

- Thoroughly updated Ethernet boot.

- In Functional description:
  - In Table 170 added the text "BootROM uses BOOT_CFG/RCON based values, table 170, for configuring QSPI controller for applying successful initial configuration" in the QuadSPI initial configuration description.
  - In Table 170 changed the changed the settings of RCON/BOOT_CFG1[30:29] for 00 and 10 to Reserved.
- In Table 173 updated the description of DLL_AUTO_UPD_EN, DLL_SLV_UPD_EN and DLL_BYPASS_EN.

- In Table 173, changed DLL_SLV_UPD_EN to Reserved.

A.35 Power management changes

- In NCSPD configuration changed the text from "Not mapped to any monitor for S32G274A" to "Not mapped to any monitor for the chip".
- In Use case: Interfacing to external PMIC changed the text from "For interfacing to the PMIC, see S32G274A Hardware Design Guidelines" to "For interfacing to the PMIC, see the chip Hardware Design Guidelines".
- In Standby mode exit changed the text from "For more information on Standby mode exit sequence, refer to section 10 of S32G274A data sheet" to "For more information on Standby mode exit sequence, refer to the chip data sheet".

- Safety context for PMIC WDOG :
  - In "sequence of steps" list, revised "After power supplies reach correct normal operation", step for clarity.
  - Updated Deep Fail-Safe mode description for clarity.

- In the Wake-up pin pullup/pulldown control during Run and Standby modes section, added a note "It is not valid to configure the wake-up pad from SIUL2 and write zero to WKUP_PUS[WKUP_PU_OVERRIDE] of the S32G_STBY_GPR before entering the Standby mode."

- In NCSPD configuration, changed NCSPD6 to Reserved.

- In Static Voltage Scaling (SVS), changed the bullet text to:
  - DIE_PROCESS[1:0] = 01b, SVS mode 1
  - DIE_PROCESS[1:0] = 10b, SVS mode 2
  - DIE_PROCESS[1:0] = 11b, SVS mode 3
- In Static Voltage Scaling (SVS), added text "For more information on power domain SVS voltage levels, see the “Operating Conditions" section of the chip data sheet".

- In NCSPD configuration, for NCSPD8 changed the supply name from VDD_HV_PLL_DDR0 to VDD_LV_PLL_DDR0.

A.36 Chip-specific MC_ME information changes

- In MC_ME partition 1 mapping:
  - For MC_ME_PRTN1_CORE0_STAT[CCS], mentioned "Used for" column as "CA53 cluster0 core0/1".
  - For MC_ME_PRTN1_CORE0_ADDR[ADDR], mentioned "Used for" column as "Performance mode - CA53 cluster0 core0 Lockstep mode - shared control for core0 of CA53 cluster0 and cluster1"

- For MC_ME_PRTN1_CORE1_ADDR[ADDR], mentioned "Used for" column as "Performance mode - CA53 cluster0 core1 Lockstep mode - shared control for core1 of CA53 cluster0 and cluster1"
- For MC_ME_PRTN1_CORE2_STAT[CCS], mentioned "Used for" column as "CA53 cluster1 core0/1"
- For MC_ME_PRTN1_CORE2_STAT[WFI], mentioned "Used for" column as "CA53 cluster1 core0"
- For MC_ME_PRTN1_CORE3_STAT[WFI], mentioned "Used for" column as "CA53 cluster1 core1"

A.37 MC_ME module changes

- Updated the section: Glossary.

A.38 MC_PCU changes

- Updated the first sentence of the third paragraph in Introduction.

A.39 Chip-specific PMC information changes

No substantial content changes.

A.40 PMC module changes

No substantial content changes.

A.41 Chip-specific [S32G] information changes

- Created a new chip-specific section SRAMC registers topic.

A.42 SRAMC module changes

- Updated "SRAMC register descriptions" section. Added information regarding generation of transfer errors.
- Removed 'hardware initialization' related content in Features.
- updated Latency of read and write accesses to remove content related to "2-cycle Resp".

A.43 DDR module changes

- Added a new section as Performance signals/counters
- Updated the description of CSV
- Updated the description of reg_par_poison_en

Printed under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8c4626-480-18a8-b251-390d3ab624c1

A.44 DDR_GPR module changes

• raq_split_0, raq_split_1, raq_split_2, raqr_push_0, raqr_push_1, raqr_push_2, raqr_pop_0, raqr_pop_1, and raqr_pop_2 are hidden in Configuration 5 (DDR_Config_5)
• From: 000b RD_N (RD data for non-ECC region), 001b RD_E (RD data for ECC region), 000b WD_N (WR data for non-ECC region), and 001b WD_E (WR Data for ECC region) To: 000b RD_N (RD data for non protected region), 001b RD_E (RD data for protected region), 000b WD_N (WR data for non protected region), and 001b WD_E (WR data for protected region) in Configuration 3 (DDR_Config_3)

A.45 Chip-specific QuadSPI changes

• Added a new limitation, "It is required that the application must rely on the FCCU fault for the ECC error handling by writing '0' to the QuadSPI RSER[CRCBIE] and RSER[CRCAIE]. In cases where FCCU is not configured (for example, in the customer bootloader), these fields must be enabled" in QuadSPI limitations
• In the Chip-specific feature configuration section, added a note.

A.46 QuadSPI module changes

• Glossary table added.
• Editorial update in "Internally generated DQS" section. Changed "QuadSPI chip-specific information" to "Chip-specific QuadSPI information". • Updated TX buffer operation. Updated the statement that talks about the interrupt generated by TBUFF field.
• Updated Driving external signals - Updated description of "Dummy".
• Updated decimal notation format in Programmable sequence engine and Address scheme.
• Removed the section "Internally generated DQS". • Added a new section Pad loopback, and removed the existing section "Pad loopback DQS".
• Updates in DLLCRA and DLLCRB register — Added DLLRES field — SLV_DLY_FINE field at position 5 is now removed • Updated programming sequence for DLL auto update mode in DLL and delay chain usage.

A.47 uSDHC module changes

• Changed the size of bit TUNING_CTRL[TUNING_START_TAP] from 7 to 6. • Added a new bit, TUNING_CTRL[DIS_CMD_CHK_FOR_STD_TUNING] .
• Added Command Queue Register Map from 100h to 15Ch in uSDHC register descriptions
• Corrected the reset value of Command Queuing Capabilities (CQCAP) and Command Queuing Send Status Configuration 1 (CQSSC1) as 0x0000_310A and 0x0001_1000

Table continues on the next page...

• Corrected from "<cmd_index> & 0x3f) >> 24;" to "<cmd_index> & 0x3f) << 24;" in Command send & response receive basic operation
• In DLL_CTRL[DLL_CTRL_SLV_DLY_TARGET0], added a note, "For the restrictions of delay cell implementation, the delay target must be set between REF_CLOCK/16 and REF_CLOCK*2 when REF_CLOCK is running at 200 MHz. When REF_CLOCK frequency is slower than 100 MHz, the maximum delay target might not reach REF_CLOCK*2.". • In STROBE_DLL_CTRL[STROBE_DLL_CTRL_SLV_DLY_TARGET], added a note, "For the restrictions of delay cell implementation, the minimum delay target that can be set is REF_CLOCK/16.".

A.48 Chip-specific STM information changes

No substantial content changes.

A.49 STM module changes

No substantial content changes.

A.50 Chip-specific SWT information changes

No substantial content changes.

A.51 SWT module changes

• Added "soft" to the title of Avoid accidental soft unlock. • In that topic, changed "generator can include the unlock keys for soft lock" to "generator includes the unlock keys for soft lock".

A.52 Chip-specific PIT information changes

No substantial content changes.

A.53 PIT_RTI module changes

• In Example configuration for the Lifetime Timer, removed the sentence "The timer is a downcounter".

A.54 Chip-specific FTM information changes

• In FTM instances, updated Quadrature decoder filter to Quadrature decoder.

A.55 FTM module changes

No substantial content changes.

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c8462548-1481-48a8-b251-390d3ab624c1

A.56 RTC module changes

No substantial content changes.

A.57 I²C module changes

• Revised Divider and hold values.

A.58 Chip-specific GMAC changes

No substantial content changes.

A.59 PFE module changes

No substantial content changes.

A.60 Chip-specific FlexRay information changes

No substantial content changes.

A.61 FlexRay module changes

• Made editorial changes.
• Made editorial changes.
• Made editorial changes.
• Made editorial changes.

A.62 LINFlexD module changes

No substantial content changes.

A.63 Chip-specific SPI information changes

No substantial content changes.

A.64 Chip-specific FlexCAN information changes

No substantial content changes.

A.65 FlexCAN module changes

• Added new instruction to initialization sequence in FlexCAN initialization sequence.

Table continues on the next page...

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
e8c46265-4821-48a8-b251-390d3ab624c1

• Fixed problem in note explaining maximum range value in Detection and correction of memory errors. • Created Glossary and added links to it from multiple topics.
• In FlexCAN initialization sequence changed "The FlexCAN module may be reset in three ways" to "You can reset FlexCAN in the following ways."
• In Error and Status 1 register (ESR1), added the following text to all read-only error-flag fields: "After a read operation, the field's value clears to 0." • Made editorial changes.
• Made editorial changes.

A.66 S32G CAN Timestamp changes

No substantial content changes.

A.67 Chip-specific SerDes information changes

• Added Unexpected behavior when enabling error injection in DMA Channel context memories of SerDes
• In SerDes subsystem working modes, a note about S32G234M.
• Added Converting nonsecure accesses to secure accesses.

A.68 SERDES_GPR module changes

No substantial content changes.

A.69 LLCE module

• Changed CAN instances to BCAN. • Added high-level block diagram. • Added FIFO interrupts • Added Core-to-Core module • Added Interrupts and status

A.70 Chip-specific USBOTG changes

• Added a new chip-specific section SBUSCFG[AHBBRST] field reset value topic.

A.71 USBOTG module changes

• Editorial changes • In Host mode events, corrected "PORTSC1[WKDC]" to be "PORTSC1[WKDS]"

Table continues on the next page...

• Editorial changes
• Added a new field - CR[UCMALLBE] • Updated "PID Code" description in qTD token. • Removed mention of TTCTRL register from Asynchronous transaction scheduling and buffer management and Operational registers. • Removed mention of CTRLDSSEGMENT register from FSTN back path link pointer . • Changed occurrences of EPCR register throughout the chapter to ENDPTCTRL register. • Updated field descriptions for the following. Updates are only to refer to the correct fields from the text: — OTGSC[DPIS] — USBMODE[SDIS] — ENDPTSTAT[ETBR] — USBSTS[UEI] — ENDPTCOMPLETE[ETCE] — ENDPTCOMPLETE[ERCE] — ENDPTCTRLn • Updated QH. • Updated Halting a QH.
• Updated field description for CR[PWRFLT] and CR[PWRFLTDF].

A.72 Chip-specific CTU changes

• Added, "Throughout the CTU chapter, ADC_0 is referred to as ADC A (ADC_A). Throughout the CTU chapter, ADC_1 is referred to as ADC B (ADC_B)" in CTU connections.

A.73 CTU module changes

No substantial content changes.

A.74 Chip-specific ADC information changes

No substantial content changes.

A.75 SAR_ADC module changes

• In STAW0R[THRH], added bandgap value information. • In STAW0R[THRL], added bandgap value information.
• In Introduction, corrected the formatting of "12-bit resolution".

Preliminary NDA only
COMPANY PROPRIETARY
Beijing CHT Automotive Technology Co., Ltd.
c8c42e5-4821-48a8-b251-390d3ab6240

A.76 Chip-specific TMU information changes

No substantial content changes.

A.77 TMU changes

• In Initialization information, changed from "For the last four calibration points (12-15), there are per-part trims that you need to apply. You must read these fields in the appropriate Shadow register of OCOTP—SHADOW38[EFUSES4:0], SHADOW38[EFUSES10:6],SHADOW38[EFUSES16:12], and SHADOW38[EFUSES 22:18]." changed to "For the last four calibration points (12-15), there are per-part trims that you need to apply. You must read these OCOTP fields (in the appropriate shadow register): CFG_DAC_TRIM_0, CFG_DAC_TRIM_1, CFG_DAC_TRIM_2, and CFG_DAC_TRIM_3."
• Added Table 534 to show how the eight flags are mapped to the two TMU outputs. • In Figure 543, added the single "interrupts" arrow to two signals (for example. alarm / critical alarm). • In Figure 543, removed the ADC output bitwidth "9".
• Updated the description of TCMCFG[DEMA] and TCMCFG[OCM]: — Added a note, "This field should be set to 1 for the normal TMU monitor operation. For details on initialization, see Initialization information." — Removed the description of bit field values for 0 and 1
• For the TCMCFG register: — Added a note, "The values for OCS, OCM, DEMA, DFDRCTC, and CLKDIV are chip-specific. See the "Chip-specific TMU section" for the required settings." — Updated the DAC_OFFSET reset value as undefined.

A.78 Security changes

No substantial content changes.

A.79 Chip-specific MU information changes

No substantial content changes.

A.80 MU changes

No substantial content changes.

A.81 OCOTP Controller module changes

• Added a new statement to Fuse ECC as "The OCOTP implements 7-bit ECC for every 32-bit ECC-protected eFuse word."
• A new table is added to show the OCOTP memory address space assignments - Table 546.

Posted under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
ca4605-4821-48a8-b251-390c3ab624

A.82 FBXC module changes

No substantial content changes.

A.83 Safety changes

No substantial content changes.

A.84 Chip-specific ERM information changes

• In Memory error event sources and captured information, for SRAMC 0 and SRAMC1 under ERM PER, added a footnote, "The status must be cleared from SRAM Controller (SRAMC) before clearing the ERM status for SRAMC".

A.85 ERM module changes

• Added section: Glossary. • Updated the register names to show actual error "number" instead of showing generic "n".
• Updated the Overview section. • Changed the ordering bullets in Features. • From Features, replaced the term "ECC" with "error" in points 1 and 3. • In Initialization, updated the first sentence as follows: For each ERM channel supporting memory with ECC, prepare the corresponding memory array before enabling ERM interrupts about errors for that memory. • In Features, updated the last bullet as follows: Recording the count value of the number of corrected error events • Removed the following sentence from the "ERM register descriptions": The ERM provides an IPS programming model mapped to an on-platform peripheral slot. • Removed the term "IPS" from the following statement in "ERM register descriptions": Any of the following attempted references to the programming model generates an error termination

A.86 FCCU chip-specific changes

No substantial content changes.

A.87 FCCU module changes

• Glossary table added.
• In Introduction, replaced the term "hardware channel" with "hardware interface", moved all features from this section to Features section. • In FCCU submodules, for the HNSHK blocks submodule, clarified the clock names as "CLKPRIM(module clock) and CLKSAFE(RC oscillator clock)" • In Signals, replaced column heading "Active level" with "Sensitivity", added "NCF" as "Programmable". • In Definitions, removed the term "edge-triggered".

Table continues on the next page...

- In FSM description, removed a paragraph, "Consider the case... has IRQ enabled.
- In FCCU state diagram, replaced the term "Any fault pending" with "pending fault (unmasked AND timeout enabled)".
- In EOUT interface, removed the table, "FCCU error pin behavior on state transitions" and text associated with it.
- In , Bi-Stable protocol , updated the terms "Normal phase" with "Non-faulty" and "Error phase" with "Faulty phase".
- Added footnote in the reset value of EINOUT[EINn]".
- In CFG register bit value at different events, replaced the table with a sentence, "In this chip, there are no events that affect the bits in the configuration register."

- In Features, re-arranged the "Lockable configuration" bullet, added sub-bullets in it.
- Removed all occurrences of "Dual-Rail" and "Time-Switching" modes from the chapter.
  - Updated CFG[SM] field as “reserved”.
- In CFG[FOM], removed the note, "In the Test modes, ...input/outputs with CLKSAFE."

- Removed the section, "Fault signal flow".

A.88 Chip-specific CMU_FC information changes

No substantial content changes.

A.89 Chip-specific CMU_FM information changes

No substantial content changes.

A.90 SELFTEST_GPR module changes

No substantial content changes.

A.91 SELFTEST_GPR_TOP module changes

No substantial content changes.

A.92 Chip-specific STCU2 information changes

- In Indicating fault state during the main reset domain self-test, Replaced, "When MC_RGM's ERCTRL[ERASSERT] field is programmed to 1 during the main reset domain self-test, along with RESET_B assertion, it also results into FCCU EOUT pins indicating the fault state" with "The Software must program the MC_RGM's ERCTRL[ERASSERT] field to 1b prior to initiating the main reset domain self-test. This results in the RESET_B pin being asserted, the FCCU EOUT pins indicating the fault state, and all other pins being put into their default state as indicated by the "Pad State During Main Reset Domain Selftest" column of the "Misc With Pin Controls" tab of IOMUX sheet".

- In STCU2 BIST start (BSTART) register description, added the field description of the register
- In Self-test programming sequence, added, "instructions for the configurations when the selftest is executed on the software resettable domain followed by the selftest run on the main reset domain, or software resettable domain."

- For LBIST28, updated the value from Reserved to NOC_Internal in STCU2 LBIST mapping

A.93 STCU2 module changes

• Added section: Glossary.
• In Main features, replaced "IPS" with "System", clarified BIST related feature, removed the term "ITF" and removed the feature, "LB_MISRELSWn[MISRESWx].... software". • Removed the term "TCK" throughout the content. • In Figure 564, removed "MC Reset" and removed extraneous terms "Kern 1 and Kern 2" from the "Fault interfaces". • In Peripheral bus interface, replaced the term, "Any other operation (such as free byte enables) has to be avoided" with "Any other operation (such as byte enable) is not supported." • In CFG[PTR], LB_CTRL[PTR], MB_CTRL[PTR] field maps, replaced "LBIST" with "NLBIST" and "MBIST" with "NMCUT" actual parameter values. • In CFG[CLK_CFG], corrected field map description: "sys_clk1" with "sys_clk/1", "sys_clk2" with "sys_clk/ 2" and so on • In field map, clarified PLL unlock as PLL unlock (loss-of-clock) • In MBSSW[MBSSWn], — updated the term "Online status of MBISTs" with "Online status of MBISTn" — updated the term "NMCUT BIST with "MBIST" • In ALGOSEL and BSTART register description, added a sentence to see "Chip-specific STCU2 information" for details. • In BIST scheduling, updated MBIST mapping in hex format and removed the reference to "Use cases and limitations" section.
In Glossary: • Rearranged the entries in alphabetical order. • Changed definitions to sentence case. • Changed "self test" to "self-test".
• Removed extraneous bracket from 3Ch and updated 06Eh value to all caps in CFG[PTR], LB_CTRL[PTR], MB_CTRL[PTR].
• In STCU2 LBIST Control (LB_CTRL0 - LB_CTRL59), updated access of field [7] to "RW" (was "RO"). • Made clarifications in the Peripheral bus interface about the write access. • In the CFG[PTR], added "others: invalid pointer...STCU2 Error (ERR_STAT)." • In BIST scheduling, added a sentence, "A BIST descriptor identifies...control register (STCU2 LBIST Control (LB_CTRL0 - LB_CTRL59) or STCU2 MBIST Control (MB_CTRL0 - MB_CTRL109))".

A.94 SBSW module changes

• Updated the description for SAFETY_IP_CLK in Signal descriptions. • Added title to the table in Signal descriptions. • Replaced all references of SBSW clock with SAFETY_IP_CLK in the chapter. • Added information in "SBSW register descriptions" about the offset address (1Ch) in the SBSW memory space on which accesses does not result in transfer error.

A.95 Chip-specific EIM information changes

- In the EIM instances, updated SerDes_1 to PCIe_1.

A.96 EIM changes

- Added section: Glossary
- In Introduction, updated the second sentence.
- Removed the term "IPS" from the first sentence of "EIM register descriptions".

- Updated the register names to show actual Error Injection Channel Descriptor "number" instead of showing generic "n".
- Removed the term "IPS" from the chapter.
- Updated the Introduction.

- Added following content in the Overview.

A.97 CRC module changes

No substantial content changes.

A.98 Chip-specific JTAGC information changes

No substantial content changes.

A.99 JTAGC module changes

- Fixed shared review rejected comments and glossary added.
- Made editorial changes.

- Shared review comments incorporated.

A.100 CJTAG module changes

No substantial content changes.

A.101 ATP module changes

No substantial content changes.

A.102 JDC module changes

- In Secure challenge/response connection procedure added statement "For details related to the lifecycle stage at which this process is performed, see the "Hardware Security Engine (HSE_B)" chapter."

Provided under NDA only
COMPANY PROPRIETARY
Beijing CHJ Automotive Technology Co., Ltd.
c0462e5-4821-48a8-b251-390d3ab621c1

How To Reach Us

Home Page:

nxp.com

Web Support:

nxp.com/support

Information in this document is provided solely to enable system and software implementers to use NXP products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document. NXP reserves the right to make changes without further notice to any products herein.

NXP makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does NXP assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters that may be provided in NXP data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including “typicals,” must be validated for each customer application by customer's technical experts. NXP does not convey any license under its patent rights nor the rights of others. NXP sells products pursuant to standard terms and conditions of sale, which can be found at the following address: nxp.com/SalesTermsandConditions.

While NXP has implemented advanced security features, all products may be subject to unidentified vulnerabilities. Customers are responsible for the design and operation of their applications and products to reduce the effect of these vulnerabilities on customer’s applications and products, and NXP accepts no liability for any vulnerability that is discovered. Customers should implement appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP, the NXP logo, NXP SECURE CONNECTIONS FOR A SMARTER WORLD, COOLFLUX, EMBRACE, GREENCHIP, HITAG, I2C BUS, ICODE, JCOP, LIFE VIBES, MIFARE, MIFARE CLASSIC, MIFARE DESFire, MIFARE PLUS, MIFARE FLEX, MANTIS, MIFARE ULTRALIGHT, MIFARE4MOBILE, MIGLO, NTAG, ROADLINK, SMARTLX, SMARTMX, STARPLUG, TOPFET, TRENCHMOS, UCODE, Freescale, the Freescale logo, AltiVec, C-5, CodeTEST, CodeWarrior, ColdFire, ColdFire+, C-Ware, the Energy Efficient Solutions logo, Kinetis, Layerscape, MagniV, mobileGT, PEG, PowerQUICC, Processor Expert, QorIQ, QorIQ Qonverge, Ready Play, SafeAssure, the SafeAssure logo, StarCore, Symphony, VortiQa, Vybrid, Airfast, BeeKit, BeeStack, CoreNet, Flexis, MXC, Platform in a Package, QUICC Engine, SMARTMOS, Tower, TurboLink, and UMEMS are trademarks of NXP B.V. All other product or service names are the property of their respective owners. AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile are trademarks or registered trademarks of Arm Limited (or its subsidiaries) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. The Power Architecture and Power.org word marks and the Power and Power.org logos and related marks are trademarks and service marks licensed by Power.org. Synopsys & DesignWare are registered trademarks of Synopsys, Inc.

Portions © 2017, 2019 Synopsys, Inc. Used with permission. All rights reserved.

Provided under NDA
COMPANY PROPRIETARY
Beijing C*** Technology Co., Ltd.
c8c462e5-***-***-***-************

© NXP B.V. 2020. **All rights reserved.**

For more information, please visit: http://www.nxp.com
For sales office addresses, please send an email to: salesaddresses@nxp.com

Date of release: Sep 2020
Document identifier: S32G2RM